НАЦІОНАЛЬНИЙ ПЕДАГОГІЧНИЙ УНІВЕРСИТЕТ
імені М.П.ДРАГОМАНОВА

на правах рукопису

Лукаш Ірина Миколаївна

УДК 681.3 (07)

ФОРМУВАННЯ ІНТЕЛЕКТУАЛЬНИХ УМІНЬ СТАРШОКЛАСНИКІВ У ПРОЦЕСІ НАВЧАННЯ ІНФОРМАТИКИ

13.00.02 – теорія та методика навчання інформатики

ДИСЕРТАЦІЯ

на здобуття наукового ступеня

кандидата педагогічних наук

Науковий керівник

кандидат фізико-математичних

наук, професор

Рамський Юрій Савіянович

Київ - 2003

ЗМІСТ

ВСТУП

Актуальність дослідження. Зростання темпів науково-технічного прогресу і розвиток нових форм економічної і соціальної діяльності призводять до посилення взаємозв'язків між людьми у процесі здійснення сумісних проектів, розумного і мирного розв'язування неминучих конфліктів. У зв'язку з цим актуальною проблемою сьогодення є перегляд задач освіти з акцентом на соціалізацію особистості. Саме соціалізації належить провідна роль у розвитку здібностей спілкуватися, зіставляти позиції, визнавати відмінності, критику, враховувати думки інших. Сьогодні ми підійшли до того, що в сучасному світі загальний культурний рівень є фундаментальним для людини, а поєднання широкої загальнокультурної освіти з глибоким освоєнням вузькоспеціалізованих знань є важливим завданням гуманістичної освітньої парадигми.

Полем продуктивної реалізації системи формування особистісної культури учнів (знань та інтересів, переконань, вмінь і розвинених на їх основі здібностей, індивідуальних норм поведінки і освоєних методів діяльності, соціальних почуттів) є індивідуальна діяльність, зокрема, це – аксіологічна, інтелектуальна, комунікативна, естетична тощо. Грунтовні основи культурологічного розвитку безумовно закладаються в загальноосвітній школі. У зв'язку з цим змінюються вимоги до шкільної освіти. Якщо раніше навчання в основному було спрямоване на формування знань, умінь, навичок, то зараз пріоритетним критерієм вважається рівень сформованості відповідних якостей мислення (глибини, гнучкості, критичності, стійкості, усвідомленості розумової діяльності тощо). Сучасному суспільству необхідний випускник, який вміє самостійно і критично мислити, бачити і творчо розв'язувати проблеми, що виникають. На перший план висувається задача розвитку особистості учня на основі його внутрішнього потенціалу, зокрема, інтелектуального. Реалізація інтелектуального потенціалу не тільки адаптує особистість до оточуючого світу, а через творчу діяльність адаптує

оточуючий світ до потреб особистості. Закон України “Про освіту” і Державна національна програма “Україна: Освіта XXI століття” [248] принципово змінили основне завдання сучасної школи – від засвоєння учнями знань до всебічного розвитку кожної дитини як особистості і найбільшої цінності суспільства.

З огляду на вищесказане, відбувається переосмислення ролі інформатики як навчального предмету у формуванні наукового світогляду, розвитку інтелекту учнів. Питанням підтримки засобами сучасних інформаційних технологій інтелектуальної, пізнавальної діяльності учнів присвячені роботи Н.В.Апатової [9], Н.Р.Балик [17], В.Ю.Бикова [21], А.Ф.Верланя [44], О.В.Вітюка [49], М.С.Голованя [64], Ю.В.Горошка [67], А.М.Гуржія [73, 74], М.І.Жалдака [85], О.Б.Жильцова [88], Ю.О.Жука [90], І.С.Іваськіва [99], А.В.Пенькова [185], С.А.Ракова [101], Ю.С.Рамського [205], В.Д.Руденка [214], Є.М.Смірнової [230], І.О.Теплицького [242], Ю.В.Триуса [246], Г.Ю.Цибко [266], Т.І.Чепрасової [272], М.І.Шкіля [102], А.М.Ясінського [286] та ін. Програмування як засіб інтелектуального розвитку учнів висвітлюється в роботах І.М.Антіпова [10], М.З.Грузмана [70], А.П.Єршова, Г.А.Звенигородського [83], В.М.Монахова [165], Ю.А.Первіна та ін.

Одночасно з розвитком комп’ютерної техніки відбувається удосконалення інформаційних технологій. Одними з сучасних технологій проектування і створення комп’ютерних систем обробки даних є технології об’єктно-орієнтованого та візуального програмування. Проблемам розробки і використання технології об’єктно-орієнтованого програмування присвячені роботи Т.Бадда [16], Г.Буча [35], Д. Рамбо, А.Джекобсона [36], С.Шлеєра, С.Меллора [278]. В дослідженнях А.Б.Кузнєцова [128], С.О. Семерікова [222] відображені питання методики використання окремих положень технології об’єктно-орієнтованого програмування. Методичні аспекти вивчення систем візуального програмування висвітлюються в роботах Л.І.Білоусової [23],

С.Бобровського [25], А.Жукова [89], Н.В.Морзе, О.В.Копаєва [119], О.Ю.Ніколаєнко [169], В.В.Соколова [232] та ін.

Аналіз літератури показав, що на основі досліджень Д.Пойа [190, 191], психологів Д.Б.Богоявленської [26], Л.С.Виготського [51], П.Я.Гальперина [54-57], В.В.Давидова [75], Л.В.Занкова [95], О.М.Кабанової-Меллер [103, 104], З.І.Калмикової [105-108], Г.С.Костюка [120], О.М.Леонтьєва [133-136], Ю.І.Машбиця [155], Н.О.Менчинської [157, 158], В.О.Моляко [164], М.Л.Смульсон [231], Н.Ф.Тализіної [239-241], С.Л.Рубінштейна [211-213] та ін., педагогів Ю.К.Бабанського [12-14], І.Я.Лернера [137], М.І.Махмутова [154], В.Ф.Паламарчук [179-183], Т.І. Шамової [276], Г.І. Щукіної [280] та ін., методистів М.І.Бурди [34], М.Я.Ігнатенка [100], В.І.Клочка [113], О.І.Ляшенка [147], М.Т.Мартинюка [149], О.В.Сергєєва [223], З.І.Слєпкань [228] та ін. розробляються різні аспекти впливу інформаційних технологій на розвиток мислення, пізнавальної активності учнів. Проте дані констатуючого експерименту свідчать, що у старшокласників недостатньо розвинена гнучкість мислення, сформовані уміння виконувати такі загальні інтелектуальні операції, як: аналогію, узагальнення, класифікацію, абстрагування, виділення істотних ознак, знаходження закономірностей, які тісно пов’язані з аналізом, синтезом, порівнянням тощо. Однією з причин такого становища можна вважати неповномірне застосування вчителями спеціальних методик формування загальних інтелектуальних умінь учнів на уроках з різних навчальних дисциплін, зокрема з інформатики.

Одна з таких методик формування інтелектуальних умінь учнів на уроках інформатики може бути побудована на основі використання технології об’єктно-орієнтованого програмування, вивченню якої в базовому курсі інформатики ще не приділяється достатньої уваги, хоч для цього є вагомі підстави. Створення комп’ютерних моделей різних реальних систем на основі об’єктно-орієнтованого підходу передбачає виконання послідовностей загальних інтелектуальних умінь (умінь виконувати розумові операції: аналіз,

синтез, порівняння, узагальнення, класифікацію, абстрагування тощо), які є складовими спеціальних умінь (виділення об’єктів з предметної галузі, створення абстракції об’єкта, побудова ієрархії класів тощо). Імовірно, що оволодіння цими спеціальними для технології об’єктно-орієнтованого програмування уміннями сприятиме формуванню їх складових – загальних інтелектуальних умінь.

Окрім цього, орієнтація педагогічної парадигми на особистісно-орієнтоване навчання потребує створення умов для індивідуально-творчого розвитку кожної особистості, залучення до цього процесу самої дитини. Ще недостатньо досліджене використання для створення таких умов задач-проектів моделювання функціонування реальних систем та застосування засобів проектування і візуального програмування для їх розв’язування. Подібні задачі належать до типу задач, умова яких містить недостатню кількість або взагалі не містить необхідних даних для їх розв’язування. В дисертаційному дослідженні Т.А.Завади [92] обгрунтовується, що розв’язування таких задач сприяє інтелектуальному розвитку учнів. Створюючи програмний проект учні повинні виконувати дослідження предметної галузі з метою здобуття необхідних даних, а тим самим і нових знань, що сприяє формуванню їх інтелектуальних умінь та набуттю знань не тільки з інформатики, але й з інших навчальних предметів.

Таким чином, існує протиріччя між об’єктивною необхідністю формування інтелектуальних умінь учнів у процесі навчання інформатики в школі, потенціалом засобів технологій об’єктно-орієнтованого та візуального програмування для формування інтелектуальних умінь учнів і не розробленістю відповідного наукового і методичного забезпечення, що породжує актуальну соціально значущу проблему, на вирішення якої спрямоване наше дослідження.

Тема дослідження “Формування інтелектуальних умінь старшокласників у процесі навчання інформатики” входить до плану науково-

дослідних робіт Національного педагогічного університету імені М.П.Драгоманова як складова колективної теми кафедри основ інформатики і обчислювальної техніки (номер державної реєстрації 0198U001678).

Об'єктом дослідження є процес навчання інформатики в старших класах загальноосвітньої школи.

Предметом дослідження є методика формування інтелектуальних умінь старшокласників у процесі навчання інформатики на основі технологій об'єктно-орієнтованого та візуального програмування.

Метою дослідження є розробка науково обгрунтованої методики формування інтелектуальних умінь учнів у процесі навчання інформатики в старшій школі на основі технологій об'єктно-орієнтованого та візуального програмування.

У процесі дослідження була висунута **гіпотеза**: методично обгрунтоване цілеспрямоване використання в навчальному процесі технологій об'єктно-орієнтованого та візуального програмування підвищує як теоретичну, так і прикладну спрямованість курсу інформатики, є ефективним засобом формування інтелектуальних умінь учнів у процесі навчання інформатики, що сприяє підвищенню рівня їх інтелектуальної активності, формуванню інтересу до пошукової, навчально-дослідницької, творчої діяльності.

У відповідності з проблемою і метою дослідження розв'язуються наступні **завдання**:

1. Виявити стан проблеми формування інтелектуальних умінь учнів в практиці шкільного навчання інформатики та ступінь її розробки в психолого-педагогічній і методичній літературі.
2. Визначити понятійно-методичний апарат, структуру, вихідні принципи та методи діагностики сформованості інтелектуальних умінь учнів у процесі навчання інформатики.

3. Визначити типи програмних засобів технологій об’єктно-орієнтованого та візуального програмування з урахуванням вимоги до них - застосування для формування інтелектуальних умінь учнів та розв’язування задач-проектів.
4. Розробити компоненти методичної системи формування інтелектуальних умінь учнів на основі технологій об’єктно-орієнтованого і візуального програмування з використанням активних методів навчання та експериментально перевірити її ефективність.

Для розв’язання поставлених завдань використовувалися такі **методи дослідження**:

теоретичні: системний аналіз наукової психолого-педагогічної та навчально-методичної літератури з проблеми дослідження (уточнення понятійного апарату, змісту інтелектуальних умінь і закономірностей їх формування); аналіз програм, навчальних посібників і методичних рекомендацій, технічної літератури, існуючих програмних засобів; обробка педагогічного експерименту методами математичної статистики (підтвердження ефективності експериментальної методики);

емпіричні: діагностичні (психолого-діагностичне анкетування, бесіди з учителями і учнями); обсерваційні (спостереження за навчальним процесом в школі, аналіз уроків інформатики у 10-11 класах, систематизація та узагальнення педагогічного досвіду); експериментальні (констатуючий, пошуковий, формуючий експерименти) з метою апробації запропонованої методики та експериментального впровадження у шкільну практику основних положень дослідження.

Методологічною основою дослідження є Закон про освіту, Державна національна програма “Освіта” (Україна XXI століття); положення теорії пізнання про взаємозв’язок теорії та практики, про пізнання як активну перетворюючу діяльність людини; дидактичні ідеї розвивального (В.В.Давидов [75], Д.Б. Ельконін [282], А.З.Зак [94], З.І.Слепкань [228],

І.С.Якіманська [284]), проблемного (В.Т.Кудрявцев [126], І.Я.Лернер [137], А.М.Матюшкін [153], М.І.Махмутов [154], М.М.Скаткін [226], Г.І.Щукіна [280]), формуючого (В.П.Беспалько [19], П.Я.Гальперін [57], Н.Ф.Тализіна [239], С.І.Шапіро [274]), гуманістичного, особистісно-зорієнтованого (Ш.А.Амонашвілі [6], Л.В.Занков [95], М.В.Зверєва [97], В.Ф.Паламарчук [182], В.О.Сухомлинський [236]) навчання; принцип психології про єдність свідомості і діяльності (Л.С.Виготський [51], О.М.Лєонтьєв [133], С.Л.Рубінштейн [212]); теорія діяльнісного та поетапного підходу до формування прийомів розумової діяльності (П.Я.Гальперин [56], О.М.Кабанова-Меллер [104], В.І.Крутецький [125], Н.О.Менчинська [158]); підходи пояснення природи інтелекту: соціально-культурний (Л.С.Виготський [51]), генетичний (Ж.Піаже [186]), процесуально-діяльнісний (Л.В.Венгер [43], С.Л.Рубінштейн [212], Н.Ф.Тализіна [241]), навчальний (Г.А.Белурава [20], З.І.Калмикова [105], Н.О.Менчинська [157], Р.Фейерштейн [288]), феноменологічний (В.Келер [112], Р.Мейлі [290]), функціонально-рівневий (Б.Г.Ананьєв [8]); загально-дидактичні положення; концепція інформатизації освіти.

Наукова новизна дослідження полягає в тому, що запропоновано та теоретико-експериментально обгрунтовано новий підхід (на основі технологій об’єктно-орієнтованого та візуального програмування) до розв’язання проблеми формування інтелектуальних умінь учнів; розроблені компоненти методичної системи формування інтелектуальних умінь учнів на основі технологій об’єктно-орієнтованого та візуального програмування у процесі навчання інформатики; запропоновано новий зміст навчального матеріалу з інформатики, який сприяє формуванню інтелектуальних умінь учнів та підвищенню рівня їх інтелектуальної активності.

Теоретичне значення дослідження полягає у теоретичному та експериментальному обгрунтуванні особистісно-зорієнтованої методичної системи формування інтелектуальних умінь учнів на основі технологій

об’єктно-орієнтованого та візуального програмування; визначено ефективність моделювання предметних галузей у процесі розв’язування задач-проектів засобами технологій об’єктно-орієнтованого та візуального програмування як непрямого методу формування загальних інтелектуальних умінь учнів; визначено структуру і зміст тем: “Основи об’єктно-орієнтованого програмування”, “Основи об’єктно-орієнтованого аналізу та проектування”, “Основи візуального програмування”, які можуть бути використані в курсі інформатики старшої школи.

Практичне значення дослідження визначається тим, що розроблено компоненти методичної системи формування інтелектуальних умінь учнів на основі технологій об’єктно-орієнтованого та візуального програмування у процесі навчання інформатики в старшій школі. Розроблені теоретичні положення реалізовані у вигляді публікацій. Основні положення дисертації можна використовувати у процесі створення та вдосконалення чинних підручників. Матеріали і висновки дослідження можуть бути використані методистами інститутів підвищення кваліфікації учителів, викладачами вузів, учителями і студентами.

Особистий внесок здобувача полягає у науковому обгрунтуванні теоретичної моделі формування інтелектуальних умінь учнів на основі технологій об’єктно-орієнтованого та візуального програмування; у впровадженні програмних засобів на основі технологій об’єктно-орієнтованого та візуального програмування; у визначенні структури і змісту нових тем “Основи об’єктно-орієнтованого програмування”, “Основи об’єктно-орієнтованого аналізу та проектування”, “Основи візуального програмування”; у розробці програмного середовища DESS (Diagnostic Expert Systems Shell) як засобу формування інтелектуальних умінь учнів у процесі навчання інформатики.

Обгрунтованість і вірогідність одержаних наукових результатів і висновків дисертації забезпечена методологічною обгрунтованістю вихідних

теоретичних положень дослідження; застосуванням комплексу методів педагогічного дослідження, адекватних його меті та завданням; репрезентативністю вибірок об'єктів дослідження; результатами обробки даних експериментального дослідження.

Апробація і впровадження результатів дослідження. Основні положення та результати дослідження доповідались і отримали схвалення на Всеукраїнській конференції "Актуальні проблеми вивчення природничо-математичних дисциплін у загальноосвітніх навчальних закладах України" (Київ, Київський національний університет імені Тараса Шевченка, 1999 р.), Всеукраїнській науково-методичній конференції "Профорієнтація та довузівська підготовка майбутніх спеціалістів: проблеми, досвід, перспективи" (Чернігів, 1999 р.), IV Всеукраїнських читаннях, присвячених пам'яті М.В. Остроградського "Педагогіка математики і природознавства" (Полтава, 2000 р.), Всеукраїнській науково-практичній конференції "Інформатика та комп'ютерно-орієнтовані технології навчання" (Хмельницький, 2001 р.), Всеукраїнському науково-методичному семінарі з питань використання засобів сучасних інформаційних технологій в навчальному процесі (Київ, НПУ імені М.П. Драгоманова, 2001р.), Всеукраїнській науково-практичній конференції "Комп'ютери в навчальному процесі" (Умань, 2002 р.). Результати дослідження обговорювались на засіданнях кафедри основ інформатики і обчислювальної техніки, звітних наукових конференціях НПУ імені М.П. Драгоманова (1999-2001 р.р.) та викладені шляхом публікацій.

Результати дослідження впроваджено у навчальний процес загальноосвітніх шкіл м.Чернігова № 8, 9, 12, гімназії гуманітарно-естетичного профілю №31, колегіуму №11 м.Чернігова, Чернігівського обласного педагогічного ліцею, фізико-математичних факультетів Чернігівського державного педагогічного університету імені Т.Г. Шевченка та

Національного педагогічного університету імені М.П.Драгоманова, Чернігівського інституту післядипломної педагогічної освіти.

Публікації. За темою дослідження опубліковано 19 наукових праць, із них 14 статей – у провідних фахових виданнях, 5 робіт – у збірниках матеріалів і тез конференцій.

РОЗДІЛ 1. ТЕОРЕТИЧНІ ОСНОВИ ФОРМУВАННЯ ІНТЕЛЕКТУАЛЬНИХ УМІНЬ СТАРШОКЛАСНИКІВ У НАЧАЛЬНО-ПІЗНАВАЛЬНІЙ ДІЯЛЬНОСТІ

1.1. Проблема інтелектуального розвитку учнів в психолого-педагогічній літературі

1.1.1. Необхідність інтелектуального розвитку учнів

Проблема інтелектуального розвитку людини розглядається в багатьох психолого-педагогічних дослідженнях. Актуальність цієї проблеми пов'язана з постійно зростаючими вимогами суспільства щодо його членів. Визначаються причини необхідності інтелектуального розвитку людини як на фізіологічному та побутовому, так і на державному рівнях.

Розглядаючи інтелектуальні уміння як змістовно-операційну компоненту пізнавальної активності [64, с.22; 210, с.23], зазначається, що пошукова активність є одним з механізмів удосконалення і розвитку центральної нервової системи. Щоб особистість змогла реалізуватись в напруженому світі, її мозок повинен мати великі і різнобічні можливості у засвоєнні, реорганізації та вибиранні інформації із пам'яті, у регуляції поведінки, яка весь час ускладнюється [210, с.23].

Таким чином, пізнавальна активність, виконання відповідних їй розумових дій, сприяє адаптивному розвитку нервової системи, а остання в свою чергу стимулює самовдосконалення кожної людини, розвиток її духовності, творчості. Інтелектуальна творчість є соціальним механізмом, що протистоїть регресивним течіям в розвиткові суспільства. В.О.Сухомлинський стверджував, що неук може бути дуже небезпечним для суспільства. Він не може бути щасливим сам і тому шкодить іншим. Випускнику школи дозволяється чогось не знати, але він обов'язково повинен бути розумною людиною [236, с.3]. Розумна людина здатна висувати ідеї, які вважаються продуктом інтелектуальної творчості. Сфера ідей в суспільній атмосфері схожа на озонову у звичайній земній атмосфері. Чим менше в суспільстві розумних людей, тим більш тонкою стає інтелектуальна сфера,

тим, відповідно, більше "озонових дір" і тим більш виражені деструктивні тенденції в суспільстві [258, с.10], які можуть викликати погіршення економічної ситуації в державі.

“Прояв розуму і спрямована ним праця людини”, “наукова думка соціального людства” розглядалися В.І. Вернадським “як нова небувала на планеті геологічна сила” (ноосфера), яка обумовлює науковий і технічний прогрес [45, с.67]. Початок розвитку ноосфери пов’язується з періодом формування потужних античних держав, з відносно розвинутим землеробством і великими містами. Ця економічна основа сприяла швидкому збільшенню населення і зростанню “культурної біохімічної енергії людства”, а відповідно, зростанню і прояву наукової думки в суспільному житті.

В історії останнього століття відомі такі факти, що в найбільшій мірі завдячуючи саме інтелектуальному виробництву країни досягали значного економічного зростання. Невипадково японські фахівці класифікують всі країни за ознакою рівня розвитку інтелектуальної власності (ідеї і нові технології – країни першої групи) [182, с.3]. Переважання інтелектуального виробництва і інтелектуальної власності надає право деяким аналітикам говорити про глобальний інтелектуальний переділ світу з його жорсткою конкурентною боротьбою окремих держав за переважне володіння інтелектуально обдарованими людьми – потенціальними носіями нового знання [258, с.10]. Тому кожна цивілізована країна або та, яка хоче бути цивілізованою, повинна дбати про інтелектуальний потенціал суспільства взагалі і кожної людини зокрема. Шлях до демократичного, цивілізованого суспільства зумовлений не стільки економічними пріоритетами, скільки загальним розвитком кожної особистості, її свідомості та самосвідомості, творчого потенціалу [196, с.3], реалізація якого залежить від особистої свободи людини.

Саме розвинутий інтелект є гарантією особистої свободи людини та самодостатності її індивідуальної долі. Чим більше людина використовує свій

інтелект при аналізі і оцінюванні того, що відбувається, тим менше вона піддається спробам маніпулювати нею зовні. Філософська формула “свобода є пізнана необхідність” стосується і психології: людина може вести себе незалежно від ситуації, тільки, якщо вона має повне і адекватне уявлення про цю ситуацію [258, с.10-11].

Маючи уявлення про загострення глобальних (передусім екологічних) проблем і їх наслідки, розумна людина буде підпорядковувати всі види своєї діяльності меті їх вирішення і запобігання [145, с.21].

Але незважаючи на загальне підвищення інтелектуального розвитку суспільства глобальні проблеми продовжують висіти над людством, загрожуючи йому загибеллю. Одна з найважливіших причин, яка заважає розробленню єдиної концепції вирішення глобальних проблем, обумовлена так званим “інформаційним вибухом”. В більшості галузей людської діяльності кількість інформації подвоюється приблизно кожні 10 років. У зв’язку з цим швидкими темпами відбувається звуження різних спеціалізацій, поява багатьох принципово нових технологій виробництва і взагалі нових спеціалізацій у різних видах людської діяльності. Знання розпадаються на неосяжну кількість тих своїх елементів, які погано пов’язані один з одним. Це негативно позначається на розв’язанні комплексних проблем, в тому числі і глобальних, яке потребує взаємозв’язку різних наук [145, с.25].

У зв’язку з тим, що зараз важко передбачити, які саме спеціалізовані знання для кадрів будь-якої галузі будуть потрібні суспільству через декілька років, а тим більше десятиріч, завдання освіти вбачають не в передачі молодому поколінню тих знань і вмінь, які людство набуло у своєму попередньому досвіді, а у підготовці такої особистості, яка може творчо вирішувати будь-які проблеми, в тому числі і ті, що будуть виникати у майбутньому і про які ми зараз нічого не знаємо [145, с.24].

Необхідність формування творчої особистості, здатної самостійно добувати нові знання, призводить до перегляду співвідношення інтелекту та

знань у процесі навчання. Переведення уваги на інтелектуальний розвиток учнів, потребує внесення певних змін в традиційні форми, методи та зміст навчально-пізнавальної діяльності учнів.

1.1.2. Вирішення проблеми інтелектуального розвитку учнів в різних педагогічних концепціях

Проблема інтелектуального розвитку учнів, що набуває більшого загострення в наш час, не є новою. Кілька останніх десятиліть в педагогіці розроблялися різні підходи щодо її вирішення.

Найбільш поширеною в педагогіці стала концепція розвивального навчання, основні теоретичні положення і методичні рекомендації якої висвітлені в науково-дослідницьких роботах В.В.Давидова [75], Д.Б.Ельконіна [282], А.З.Зака [94], З.І.Слєпкань [227, 228], І.С.Якіманської [284] та інших. Розвивальним вважається “навчання, яке забезпечує повноцінне засвоєння знань, формує навчальну діяльність і тим самим безпосередньо впливає на розумовий розвиток” (І.С.Якіманська [284, с.5]). Сутність розвивального навчання полягає в тому, що вчитель, використовуючи різні методи і форми роботи з учнями на уроці, систематично навчає їх способам виконання тих чи інших видів навчально-пізнавальної діяльності [227, с.6], забезпечуючи їх розумовий розвиток, тобто якісні та кількісні зміни в психічній діяльності. До якісних змін належить набуття самостійності, критичності, конструктивності, гнучкості, проблемності мислення тощо та відповідних їм інтелектуальних умінь. Про кількісні зміни в психічній діяльності учня буде свідчити швидкість виконання тих чи інших розумових операцій. Наявність перерахованих якостей мислення є передумовою перетворювання активної пізнавальної діяльності учня на його самоактивність, яка є джерелом подальшого розумового розвитку [284, с.138]. Основними компонентами розумового розвитку, за якими визначають його якість, вважаються фонд дійових знань і здатність до навчання, тобто здатність до набуття знань, умінь

та навиків, під якою розуміють систему інтелектуальних властивостей особистості, особливостей розуму [107]. Цінність різних моделей розвивального навчання залежить від того, наскільки вони впливають на формування зазначених вище компонент розумового розвитку учнів.

В роботах В.Т.Кудрявцева [126], А.М.Матюшкіна [153], М.І.Махмутова [154], М.Н.Скаткіна [226], Г.І.Щукіної [280] та інших визначаються теоретичні основи однієї з моделей розвивального навчання – проблемного навчання. В процесі навчальної діяльності в межах зазначеної моделі учень повторює логіку пройденого шляху в науці для вирішення деякої проблеми, як продукт розв'язання якої були отримані відповідні знання. Таким чином знання подаються не в "готовому" вигляді як догми, а передбачається активна пізнавальна творча діяльність учнів з метою їх здобуття. В психологічну основу проблемного навчання покладена теорія мислення як продуктивного процесу, висунута С.Л.Рубінштейном, згідно з якою мислення розглядається як відшукування, відкриття нового [212]. Схема проблемного навчання полягає в наступному: постановка вчителем навчально-проблемної задачі, що створює в учнів проблемну ситуацію; усвідомлення, прийняття і розв'язування проблеми, що виникла, у процесі чого вони оволодівають узагальненими способами набуття нових знань; застосування даних способів для розв'язання конкретних систем задач [126, с.49]. Наведена схема моделює творчий процес пошуку нових знань, в результаті якого учень повинен виконувати ряд теоретичних процедур: проведення елементарної абстракції і узагальнення, виявлення нових суттєвих, закономірних і необхідних властивостей об'єкта, побудова загальних способів їх перетворювання, чому передує розгорнутий аналіз ситуації, здійснення пошуку, висування проміжних гіпотез. Реалізація систематичного виконання учнями зазначених дій безперечно сприятиме їх інтелектуальному розвитку. Якщо при традиційному (інформативному) навчанні, коли звертається увага на досягнення найближчих результатів

(засвоєння сукупності конкретних знань, вмінь, навичок, тренування наявних способів діяльності), найменша зміна в умовах діяльності викликає збій в діяльності, то при проблемному навчанні досягається зона дальніх результатів навчання – формується система узагальнених способів розв'язування проблемних задач, що закономірно призводить до універсалізації творчих можливостей учнів, а тим самим до їх розвитку.

Незважаючи на доволі сильний потенціал щодо розумового розвитку учнів проблемне навчання не набуло широкого розповсюдження в шкільній практиці масового навчання, хоча відомо, що ще Сократ не викладав своїм учням готове знання, а приводив їх до розуміння цих знань за допомогою евристичної бесіди, тобто системи навідних питань, відповідаючи на які учні набували нових знань. Проблемне навчання забезпечує позитивний ефект при наявності початкового мінімуму знань заданого рівня узагальненості та при відповідності складності задачі рівню здатності учня до навчання. При колективному розв'язуванні проблем не всі учні мають однаковий рівень здатності до навчання і відповідний фонд знань. У більш слабких учнів не виникає проблемної ситуації, вони залишаються пасивними учасниками уроку. Найбільший розвивальний ефект проблемне навчання може дати тільки в умовах самостійного розв'язування задач [107, с.37] та індивідуальної роботи.

Більшої підтримки педагогів набула формуюча модель розвивального навчання, розробкою якої займалися В.П.Беспалько [19], П.Я.Гальперін [54, 57], О.М.Кабанова-Меллер [104], Н.Ф.Тализіна [239, 240], С.І.Шапіро [274] та інші. Для підвищення розвивального ефекту навчання в межах цієї моделі використовується принцип спеціального формування прийомів розумової діяльності шляхом розв'язування і навчання алгоритмів розв'язування різних типів задач. "Жорстке", цілеспрямоване управління розумовою діяльністю учнів, що передбачає проходження ними ряду етапів (мотивації дії, складання схеми орієнтовної основи дії, формування дії в матеріалізованій формі,

перенесення) гарантує сформованість знань і умінь з визначеними якостями, забезпечує більш швидке оволодіння раціональними прийомами мислення, запобігання від помилок, які, на думку вчених, заважають формуванню правильного мислення і тим самим уповільнюють розвиток. Слід зазначити, що, як підтверджують дослідження З.І.Калмикової [108] та інших, навчання алгоритмів розв'язування задач, тобто розв'язування задач за зразком, більше сприяє розвиткові репродуктивного мислення, ніж творчого. Але, переслідуючи мету формування інтелектуальних умінь учнів, не доречно відмовлятися від цієї моделі навчання. Репродуктивне мислення сприяє формуванню необхідного фонду міцних знань і умінь учнів, від чого залежить розуміння, відкриття, свідоме засвоєння нового в навчально-пізнавальній діяльності.

Існуючі моделі навчання відрізняються не тільки формами та методами інтелектуального розвитку учнів, а і характером взаємодії учителя і учня. В межах авторитарної педагогічної парадигми, більше орієнтованої на набуття учнями знань, до якої відносяться розглянуті вище моделі розвивального навчання, така взаємодія приймає форму прямого впливу вчителя на учня, в результаті якого учень повинен засвоїти визначені суспільством і державою способи діяльності, норми поведінки, системи цінностей, створені попередніми поколіннями. Реалізуючи сформульовані цілі виховання і навчання, вчитель конструює шляхи і способи їх досягнення, відбирає відповідні засоби і методи впливу на учня, тим самим встановлює межі його самостійної активності в навчальному процесі, нівелює індивідуальність окремої дитини.

Прихильники гуманістичної педагогічної парадигми (Ш.А.Амонашвілі [6], Л.В.Занков [95], Н.М.Зверєва [97], В.Ф.Паламарчук [182], В.О.Сухомлинський [236] та інші) вважають, що провідну роль у розвитку дитини, у формуванні її як особистості відіграє активна її участь у процесі організації власної освіти. В межах маніпулятивної моделі (М.Монтессорі)

гуманістичної педагогічної парадигми вчитель конструює розвивально-навчальне середовище, в яке включається учень шляхом застосування відповідних дидактичних матеріалів. Уникаючи прямого формуючого впливу, опосередковано взаємодіючи з дитиною, вчитель збуджує в нього наміри, що спрямовують його активність в напрямку, який не завжди збігається з бажанням учня, але при цьому у нього залишається ілюзія самостійності і незалежності.

На відміну від маніпулятивної, в особистісно-орієнтованій моделі учню надається можливість вільно вибрати вид навчальної діяльності з позиції своїх потреб, внутрішніх сил і інтересів, будується "індивідуальна траєкторія його розвитку" [130, с.48]. "Процес оволодіння знаннями здійснюється в атмосфері інтелектуальних, моральних та естетичних переживань, зіткнення думок, поглядів, позицій, наукових підходів, проектування різних розв'язків задач, творчості вчителів і учнів" [29, с.14]. Організація особистісно-орієнтованого навчання вимагає використання педагогічних технологій, метою яких є не накопичення знань та умінь, а постійне збагачення їх досвідом творчості і формування механізму самоорганізації особистості кожного учня. Реалізація такої задачі вимагає виділення і моделювання таких видів діяльності в межах предметного навчання, які б допомагали здійснювати активне відношення учнів до оточуючого світу [182, с.5].

Застосування зазначених педагогічних моделей для організації навчально-виховної роботи в школі призводить до ускладнення змісту навчання і структури пізнавальної діяльності в його засвоєнні, що робить необхідним опанування учнями різноманітними прийомами мислення, інтелектуальними уміннями, завдяки чому відбувається зміна стилю їх розумової діяльності, а тим самим здійснюється їх інтелектуальний розвиток.

При розробці компонент методичної системи формування інтелектуальних умінь у процесі навчання інформатики ми намагалися комбіновано використати розглянуті вище різнопланові моделі навчання.

Орієнтуючись на особистісно-орієнтоване навчання, застосування основних ідей діяльнісного підходу до формування прийомів розумової діяльності П.Я.Гальперіна, який надає високий рівень розвитку репродуктивного мислення, ми поєднували з творчою, самостійною пізнавальною діяльністю учнів під час роботи у спеціально змодельованому вчителем навчальному предметному середовищі, виконання проблемних завдань, творчих робіт, власних проектів, в процесі створення яких протиріччя між існуючими знаннями і вимогами завдання є рушійною силою для набуття нових знань, формування і застосування загальних інтелектуальних умінь. Педагогічний досвід показує, що учні з високою потребою в пошуку без великої охоти виконують завдання за зразком (репродуктивного рівня). Їм не цікаво те, що вже добре відомо іншим. Тільки протиріччя стимулюють їх уяву, викликаючи потребу в пошуку і самостійність мислення.

1.1.3. Психолого-педагогічні методи діагностики інтелектуального розвитку учнів

В залежності від підходу до визначення інтелекту існують різні методи його діагностики, та оцінювання рівня сформованості інтелектуальних умінь. Загалом методи діагностики інтелекту поділяються на дві основні групи: тестологічні (“зрізові”) і генетичні (формуючі).

Теоретичні положення, які становлять основу тестологічних методів діагностики інтелекту, в більшості грунтуються на думці, що інтелектуальні можливості обумовлюються біологічною природою людини (Фр. Гальтон [58]), особливостями функціонування нервової системи (Х. Айзенк [4]), від чого залежить точність передачі інформації, закодованої у вигляді послідовності нервових імпульсів в корі головного мозку. При визначенні “коефіцієнту інтелекту” IQ (Intelligence Quotient) враховуються швидкість розв’язування тестового завдання, наполегливість в пошуку розв’язку та помилки виконання. “Ментальна швидкість” вважається психологічною

базою, джерелом розвитку інтелекту, а за критерії сформованості інтелектуальних умінь беруться швидкісні характеристики процесу опрацювання інформації. Не виключається також і вплив оточуючого середовища на розвиток інтелектуальних здібностей людини (А.Біне, К.Спірмен, Л.Терстоун [293]). При цьому інтелект визначається як досягнутий рівень психологічного розвитку, який виявляється в показниках сформованості пізнавальних функцій (запам'ятовування, уява, просторове розрізнення тощо) і показниках засвоєння соціального досвіду (значення слів, моральні оцінки тощо), знань, навиків і умінь. Результативні характеристики інтелектуальної діяльності вимірюються за допомогою вербальних та невербальних тестів інтелекту і оброблюються процедурами кореляційного та факторного аналізу незалежно від теорій пояснення природи інтелекту: наявності загального фактору інтелекту, який є основою кореляційних зв'язків між результатами виконання різних інтелектуальних тестів (К.Спірмен), або множини незалежних інтелектуальних здібностей (оперувати в уяві просторовими відношеннями, деталізувати зорові образи, виконувати основні арифметичні дії, розкривати значення слів, швидко підбирати слово по заданому критерію, запам'ятовувати і відтворювати інформацію, виявляти закономірності у рядку букв, цифр, фігур тощо), які відповідають за відповідні інтелектуальні операції і визначаються за допомогою окремих тестів (Л. Терстоун [293], Дж. Гилфорд [61]).

Тестологічні методи психодіагностики інтелекту дуже зручні для вчителів і психологів при роботі з великою кількістю учнів, але вони не завжди будуть давати об'єктивну оцінку рівня їх інтелектуального розвитку. Причини неточності результатів полягають в наступному:

- відмінності в інтелектуальних здібностях різних людей зводяться до кількісних показників, використання тестів не дає можливості отримати ніяких змістовних відомостей про якісну своєрідність здібностей людини, визначається тільки її місце на кривій Гауса і наскільки її показники

відхиляються від середніх значень, інтелектуальні тести надають можливість виявити піддослідних з низькими результатами, але за їх допомогою не можна відрізнити менш здібних від більш здібних;

- проводиться тестовий контроль тільки кінцевого результату діяльності учня (розв'язав – не розв'язав), особливості інтелектуальної діяльності при виконанні завдання не діагностуються і не враховуються;
- оцінювання відповіді (правильно – неправильно) здійснюється у відповідності з існуючими соціальними еталонами, тобто інтелектуальні дії, виконання яких діагностується, є адаптованими до деяких соціально-типових нормативів поведінки, вимог домінуючої культури заданих зовні, і можуть не відповідати внутрішньому стану людини;
- не завжди швидке виконання тестових завдань буде свідчити про високий рівень інтелектуального розвитку. Рефлекторні піддослідні схильні до уповільненого типу реагування та більшої точності відповідей, а імпульсивні піддослідні схильні до швидкого реагування, допускаючи при цьому значну кількість помилок [257]. Згідно з думкою О.М.Леонтьєва [135], інтелект виявляється у переміщенні орієнтації в ситуації з фази виконання до фази підготовки дії, тобто у внутрішній план, що супроводжується деякою "паузою";
- інтелектуальні тести створюються на основі штучних ситуацій, процедури і зміст завдань мають умовне відношення до функціонування інтелекту в реальній життєдіяльності;
- інтелектуальні тести є фрагментальними і не дозволяють вимірювати інтелект як ціле.

Для визначення якісного стану інтелектуальних дій рекомендується використовувати генетичний метод (Ю.В. Карпов, Н.Ф. Тализіна [110]), тобто виявляти рівень інтелектуального розвитку учня у процесі формування в нього нових дій. Доцільність використання генетичного (формуючого) підходу для діагностики сформованості інтелектуальних умінь

обгрунтовується багатьма теоретичними підходами до пояснення природи інтелекту, в основі яких лежить ідея розвитку інтелекту людини під впливом зовнішніх факторів. До таких підходів належать наступні: соціально-культурний (інтелект розглядається як результат процесу соціалізації, а також впливу культури в цілому), генетичний (інтелект розглядається як наслідок складної адаптації до вимог оточуючого середовища в умовах взаємодії людини з зовнішнім світом), процесуально-діяльнісний (інтелект як особлива форма людської діяльності), навчальний (інтелект як продукт цілеспрямованого навчання), феноменологічний (інтелект як особлива форма змісту свідомості), функціонально-рівневий (інтелект як система різнорівневих пізнавальних процесів) тощо.

З точки зору **соціально-культурного** підходу людина формується як культурно-історична істота, яка у процесі своєї життєвої діяльності засвоює матеріальні та духовні цінності, створені іншими людьми. Згідно з культурно-історичною теорією вищих психічних функцій Л.С.Виготського [51] інтелект розуміється як мислення в поняттях, розглядається понятійне мислення як свідома, категоріально-логічна форма інтелектуальної діяльності. Якістю розвитку інтелекту вважається ступінь узагальнення поняття (характеристика поняття як з точки зору узагальненості його змісту, так і з точки зору його включення в систему зв'язків з іншими поняттями). Основна формула розвитку інтелекту за положеннями Л.С.Виготського: вербалізація → категорізація → раціоналізація.

Згідно з його теорією мислення розвивається в три етапи:

1) мислення в синкретичних образах, коли предмети об'єднуються на основі суб'єктивних уявлень, випадкового враження про них;
2) мислення в комплексах, коли предмети об'єднуються на основі об'єктивних, дійсно існуючих зв'язків між ними, виявлених в ході виконання операцій порівняння, виділення часткових і загальних ознак;

3) мислення в поняттях, коли легко виділяються, абстрагуються, комбінуються окремі ознаки предметів. Думка може рухатись від окремого до загального та від загального до окремого. Окреме поняття знаходиться в системі зв'язків з іншими поняттями, тому аналіз одного і того ж самого предмету може здійснюватись різними шляхами в пошуках найбільш раціонального підходу до визначення.

До **генетичного** підходу пояснення інтелекту належить операційна теорія Ж. Піаже [186]. За цією теорією інтелектуальний розвиток – це розвиток операційних структур інтелекту, в ході якого мислительні операції набувають якісно нових властивостей: координованості (взаємозв'язок та узгодженість множини операцій), оберненості (можливість у будь-яку мить повернутись до початкової точки своїх розмірковувань, перейти до розгляду об'єкту прямо з протилежної точки зору), автоматизованості (невимушене використання), скороченості (згортання окремих ланок, "миттєвість" актуалізації), "мислення стає вільним по відношенню до реального світу".

Як критерії розвитку інтелекту беруться: ступінь інтегрованості операційних структур (поступове набуття мислительними операціями всіх необхідних якостей) та здатність під час міркування визволятись від концентрації уваги на власній точці зору, не ігноруючи інші.

Згідно з **процесуально-діяльнісним** підходом (Л.В.Венгер [43], С.Л.Рубінштейн [212], Н.Ф.Тализіна [241] та інші) інтелект людини визначається через розкриття процесу мислення, структури розумових здібностей. В структурі розумових здібностей виділяються дві компоненти: перша головна компонента – якість процесів аналізу, синтезу і узагальнення, друга – більш менш підлагоджена або відпрацьована сукупність операцій (розумових дій, за допомогою яких здійснюється відповідна діяльність) [212]. Зміст інтелектуального виховання особистості за цією теорією полягає у формуванні культури тих внутрішніх процесів, які містяться у здатності до постійного пропонування нових думок, що і вважається критерієм

інтелектуального розвитку. Л.В.Венгер рівнем інтелектуального розвитку вважав ступінь оволодіння людиною основними видами перцептивних (ідентифікація з еталоном, моделювання), мислительних (наглядно-образного, логічного мислення: виділення суттєвих ознак об'єктів) і мнемонічних дій [43].

В теоріях **навчального** підходу (А.Стаатс [287], Р.Фейерштейн [288] та ін.) природа інтелекту розкривається через процедури його набуття. Інтелект розглядається як сукупність когнітивних навичок (тобто таких, які мають відношення до психологічних механізмів опрацювання інформації на різних рівнях пізнавального відображення: сприйняття, зберігання, дедуктивні умовиводи), засвоєння яких є необхідною умовою інтелектуального розвитку [287]. На думку Р.Фейерштейна інтелект - це динамічний процес взаємодії людини зі світом, тому критерієм розвитку інтелекту вважається мобільність (гнучкість, пластичність) індивідуальної поведінки. Джерелом мобільності виступає "опосередкований досвід навчання" [288]. В результаті накопичення досвіду людина відносно ефективно адаптується до свого оточення, здатна зберігати свою власну ідентичність (тобто залишається сама собою, не зважаючи на умови та обставини, що змінюються). Для неї характерна різноманітність у поведінці, оцінках того, що відбувається, орієнтація, варіативність власних дій.

Г.А.Берулава [20], З.І.Калмикова [105], Н.О.Менчинська [158] та інші визначають природу інтелекту через "продуктивне мислення", сутність якого полягає у здатності до набуття нових знань (здатності до навчання). Показниками здатності до навчання є рівень узагальненості знань, широта їх застосування, швидкість засвоєння, темп просування у навчанні. Відповідно, "ядро" індивідуального інтелекту складають можливості людини до самостійного відкриття нових знань та їх використання у нестандартних проблемних ситуаціях. Саме характеристики здатності до навчання

визначають успішність шкільного навчання, тому вони розглядаються як критерії інтелектуального розвитку дитини [105].

При **феноменологічному** підході М.Вертгеймер [46], В.Келер [112], Р.Мейлі [290] та інші розглядають інтелект як особливу форму свідомості людини. На думку К.Келера критерієм наявності інтелектуальної поведінки вважається ефект структурності, тобто виникнення розв'язку пов'язується з набуттям поля сприйняття нової структури, в якій усвідомлюються відповідності між елементами проблемної ситуації, що мають велике значення для її розв'язування. При цьому сам розв'язок виникає раптово, на базі миттєвого переструктурування образу вихідної ситуації ("інсайту"). Таким чином, здатність до інсайту (тобто здатність до швидкої перебудови пізнавального образу в напрямку виявлення основного проблемного протиріччя) і визначає критерій розвитку інтелекту [112]. Р.Мейлі, теж дотримуючись феноменологічного підходу, у своїй роботі [290] виділяє та інтерпретує чотири наступні фактори інтелекту: складність (здатність диференціювати та пов'язувати елементи тестової ситуації); пластичність (здатність швидко та гнучко перебудовувати образи); глобальність (здатність із деякого набору елементів будувати цілісний осмислений образ); швидкість (здатність до швидкого виникнення множини різноманітних ідей відносно вихідної ситуації).

В межах **функціонально-рівневого** підходу інтелект розглядається як складна розумова діяльність, що поєднує в собі пізнавальні функції різного рівня. За цією теорією загальний напрямок інтелектуального розвитку характеризується єдністю процесів когнітивної диференціації (зростанням виразності властивостей окремих пізнавальних функцій) і процесів когнітивної інтегрованості (посиленням міжфункціональних зв'язків між пізнавальними функціями різного рівня), які і задають "архітектуру" цілісної структури інтелекту. За словами Б.Г.Ананьєва інтелект знаходиться за основними пізнавальними процесами, які розглядаються як його "робочі

органи". Інтелект забезпечує актуалізацію, координацію та контроль пізнавальних процесів в умовах побудови пізнавального образу на будь-якому рівні пізнавального відображення [8].

Активна участь у пізнавальній діяльності, набуття характерних для інтелекту за наведеними теоріями якостей розуму (гнучкості, глибини, стійкості, самостійності, критичності тощо) та якостей мислительних операцій (оберненості, скороченості, автоматизованості тощо) передбачає виконання учнями відповідних розумових дій, які поступово накопичуються у "фонді добре відпрацьованих та закріплених розумових прийомів" [124], інтелектуальних умінь. Процес "відпрацювання та закріплення", визначення рівня сформованості інтелектуальних умінь вимагає від вчителя розробки і застосування спеціальних компонент формуючих методик.

Розробка генетичних (формуючих) методик передбачає:

1) пошук інтелектуальних дій, які незнайомі учню, але доступні йому, тобто тих дій, що знаходяться в зоні його найближчого розвитку (нові дії, які учень може засвоїти за допомогою дорослого);

2) визначення характеристик, за якими необхідно проводити діагностування рівнів сформованості інтелектуальних дій;

3) формування нової дії починаючи з найвищого рівня з поступовим переходом до нижчих форм;

4) встановлення, в якій формі (на якому рівні) учень може зрозуміти пояснення вчителя, що необхідні для виконання нової дії;

5) встановлення, в якій формі учень може виконати нову дію після пояснення вчителя;

6) перевірка на кількох нових діях рівнів інтелектуальної діяльності, доступних учневі [241, с.11].

Якщо формування інтелектуальних умінь і інтелектуального розвитку учнів проводиться у процесі навчальної діяльності на основі навчального предмету, який містить в собі значну кількість наукових знань, для

встановлення рівнів розуміння учнями пояснень вчителя в [19, с.69-71] рекомендується спиратися на наступні чотири ступеня абстрагування:

Ступінь А (феноменологічний) передбачає описання фактів і явищ природною мовою, каталогізацію об'єктів та констатацію їх властивостей і якостей.

Ступінь Б (аналітико–синтетичний) містить пояснення природи, властивостей об'єктів та закономірностей явищ. На цьому рівні використовується мова науки з властивими їй поняттями і виразами, символами і позначеннями.

Ступінь В (прогностичний) передбачає пояснення явищ заданої області з створенням їх кількісної теорії, моделювання основних процесів, аналітичним представленням законів і властивостей.

Ступінь Г (аксіоматичний) передбачає пояснення явищ з високим рівнем узагальненості описання при використанні міждисциплінарної мови науки.

Для встановлення, в якій формі учень може виконати нову дію після пояснення вчителя на основі розв'язування задач, в [19, с.55-56] пропонується використовувати наступні рівні засвоєння дії:

- на першому рівні учні лише впізнають правильність використання раніше засвоєної інформації при повторному її поданні у вигляді готових розв'язків відповідних питань і задач;
- другий рівень передбачає наявність в учнів уміння відновлювати інформацію без підказки у процесі розв'язування типових задач за раніше вивченою схемою;
- третій рівень потребує від учня попереднього перетворювання засвоєних методик і їх пристосування до ситуації в нетиповій задачі;
- на четвертому рівні виявляються творчі уміння учня, його дослідницькі можливості з отримання нових даних.

Якість сформованості уміння визначається на відповідному рівні засвоєння дії і ступені абстрагування її описання. Чим вище рівень, тим якісніше сформоване уміння.

В роботах М.С.Голованя [64], Л.В.Мар'яненко [150] для отримання точніших оцінок розвиненості багатьох якостей особистості та сформованості інтелектуальних умінь даються наступні психологічні визначення:

1 бал – якість не розвинена або вміння не сформоване;

2 бали – якість або вміння проявляються рідко, що неефективно позначається на діяльності;

3 бали – властивість або вміння помітні в діяльності учня, проявляються, однак не завжди якісно;

4 бали – вміння ефективне, однак його ще можна вдосконалити;

5 балів – вміння сформоване, ефективне в навчальній діяльності, рефлексується учнем.

Оцінюючи якість розвивальних, формуючих методичних компонент навчання в своєму дисертаційному дослідженні А.В.Фурман [256] спирається на ідею Л.С.Виготського про два рівні розумового розвитку дитини: рівень актуального розвитку (РАР) і зону найближчого розвитку (ЗНР). Динаміка розвитку дитини та успішність розумової діяльності характеризується можливістю її переходу від того, що вона вміє робити самостійно до того, що робить у співробітництві з дорослим. Тому методична компонента вважається якіснішею, якщо вона сприяє максимально швидкому переходу ЗНР на РАР й виникненню нової ЗНР з широкими можливостями учня наслідування, співробітництва і самостійності. У зв'язку з цим діагностику інтелектуального розвитку учнів пропонується проводити за наступними рівнями:

1 рівень – мислительно-практичні дії лежать за межами ЗНР учня;

2 рівень – мислительно-практичні дії учня на ЗНР, але виконані у співробітництві з вчителем;

3 рівень - мислительно-практичні дії учня на ЗНР, виконані у спільній діяльності з товаришем;

4 рівень - мислительно-практичні дії учня на РАР.

Досліджуючи вплив компонент генетичної (формуючої) методики на розвиток окремих розумових якостей учнів і формування відповідних їм інтелектуальних умінь, доцільно поряд з формуючими методами діагностики використовувати тестологічні. Тестологічні методи діагностики в більшій мірі дозволяють виявити ті інтелектуальні риси, які притаманні людині з народження, тобто пов'язані з особливостями функціонування її нервової системи, її темпераментом, і оцінити їх якісні зміни після використання формуючих методик. Так, наприклад, з психологічних досліджень [253, с.25; 59, с.16-41] відомо, що в плані співвідношення пластичності і ригідності мислення в залежності від темпераменту людини тільки сангвініки мають високий рівень пластичності, а у холериків, флегматиків, меланхоліків виявлено відповідний рівень ригідності. Сангвініки легко входять у роботу, без особливих труднощів переключаються з одного виду занять на інший, швидко виконують розумові дії. Серед холериків трапляються учні, яких відзначають глибина думки, допитливість, велика працелюбність, але більшості властива прямолінійність, неприйняття, недооцінка думки інших, імпульсивність у судженнях про чиїсь помилки. Меланхолік також допитливий, здатний до розмірковувань, але йому бракує упевненості для подолання труднощів, відрізняється порівняно низькою працездатністю, домінуванням зниженого тонусу тощо. Флегматику притаманна велика працездатність, наполегливість у досягненні поставлених цілей. Але поряд з цим йому можуть бути властиві інертність думки, повільне входження в роботу, трудність переключення з однієї розумової дії на іншу, сповільнена адаптація до нових умов праці тощо.

Психологи стверджують [59, с.16-41], що на темпераменти, на їх негативні риси можна певною мірою впливати, змінюючи їх у потрібному

напрямі, тобто коригувати. Це повністю поширюється, зокрема, і на інертність нервово-психічних процесів. Тому вважається, що одним з способів коригування і розвитку інтелектуальних рис, притаманних особистості, є використання в процесі навчальної діяльності учнів відповідних методик формування інтелектуальних умінь.

1.2. Структура і теоретичні основи формування інтелектуальних умінь

1.2.1. Визначення інтелектуальних умінь в психолого-педагогічній літературі

Незважаючи на те, що порівняно зі знаннями інтелектуальні уміння розглядаються як головний показник інтелектуального розвитку учнів, оскільки знання можна здобувати і шляхом механічного заучування [157], сьогодні не існує єдиного, загальноприйнятого визначення понять: “уміння” та “інтелектуальні уміння”. Однією з причин відсутності єдиного підходу до цього питання може бути складність зазначених понять та багатоплановість взаємозв’язку між знаннями, уміннями та навичками. Рекомендується розглядати уміння з мотиваційного, змістовного, операційного боків. В мотиваційну компоненту уміння включається спонукаюча сила діяльності. До змістовної компоненти входять знання учнів на рівні фактів, понять, визначень, а в операційну – прийоми розумової діяльності та окремі дії, що входять до прийому [184, с.6; 64, с.17-19].

Загальним в різних підходах є поєднання уміння з діяльністю, тобто уміння розглядається як: здатність належно виконувати певні дії, готовність до практичних дій, засновані на доцільному використанні людиною набутих знань та навичок (С.У.Гончаренко [66]); знання в дії [197]; елементарні дії, наступні за знаннями [200]; вид діяльності, що виконується після деякого продумування, аналізу умов, шляхів розв’язування завдання, відбір потрібних знань, навичок та застосування їх в оптимальній комбінації (М.І.Єрецкий, Е.С.Пороцкий, Г.О.Люблінська [81, 146]); здатність виконувати деяку роботу в нових умовах (К.К.Платонов [192]) або у відповідності з метою та ситуацією, в якій людині доводиться орієнтуватися (А.В.Усова, А.О.Бобров [250]); контрольовані елементи діяльності, що дозволяють щось робити з високою якістю (Р.С.Немов [168]) тощо.

Враховуючи правомірність наведених визначень поняття “уміння”, в нашому дослідженні будемо дотримуватися розуміння уміння як складного комплексу розумових і практичних дій, який передбачає: 1) усвідомлення мети і умов, в яких ці дії будуть здійснюватися; 2) формування завдань

діяльності; 3) планування і вибір способів виконання дій; 4) контроль і самоконтроль за процесом діяльності.

Відповідно, уміння, в основі яких лежить система інтелектуальних дій, що складається з логічних мислительних операцій (прийомів) [22, с.28] вважаються інтелектуальними.

Різноманіття описаних в психолого-педагогічній літературі інтелектуальних умінь пояснюється інтелектуальним характером значної кількості типів діяльності людини: навчальної, самоосвітньої, пізнавальної, загальнотрудової, професійної тощо. За ступенем узагальненості розрізняють загальні, інтелектуальні уміння, що використовуються в різних видах інтелектуальної діяльності, та окремі, які містять ознаки того чи іншого предмету діяльності. О.М.Кабанова-Меллер називає їх “широкими” та “вузькими”[103]. Перші, на думку Н.О.Менчинської, носять “міжпредметний характер”[158].

У зв’язку з тим, що більшість типів інтелектуальної діяльності людини супроводжується пізнавальними процесами, то загальними інтелектуальними уміннями доцільно вважати ті, які входять до змістовно-операційної компоненти пізнавальної діяльності. В дослідницьких роботах Ю.З.Гільбуха [59], А.З.Зака [94], З.І.Калмикової [106], Л.В.Мар’яненко [150], Р.П.Озоліниш [172], В.Ф.Паламарчук [182], Ю.А.Самаріна [216, 217], Н.Ф.Тализіної [239], в яких висвітлюються особливості пізнавальної діяльності учнів у процесі навчання та характеризуються основні якості мислення, виділяються наступні загальні інтелектуальні уміння: уміння визначати поняття; виділяти головне; аналізувати матеріал; уміння виконувати операції порівняння, узагальнення, класифікації; уміння встановлювати причинно-наслідкові відношення, “проникати” в сутність явищ (глибина мислення); систематизувати матеріал; здійснювати міжпредметні та внутрішньопредметні зв’язки; уміння доказувати і спростовувати; прогнозувати, передбачати, екстраполювати події, явища; уміння здійснювати перенесення знань, умінь в інші ситуації,

застосовувати їх в нових умовах; уміння самостійно робити висновки, осягати внутрішні зв’язки явищ, самостійно використовувати вивчений матеріал, самостійно міркувати (самостійність мислення); уміння знаходити раціональніші шляхи розв’язування завдань, проблем, суперечностей (конструктивність мислення); уміння переконструйовувати засвоєний матеріал з метою його більш поглибленого осмислення (перетворювальність мислення); уміння помічати, ставити, формулювати проблему (проблемність мислення); “уміння подати на суд” свою точку зору, навіть якщо існують несприятливі обставини (сміливість мислення); уміння розглядати явище, що вивчається, з різних точок зору, бачити різні підходи та шляхи розв’язування проблеми (широта мислення); уміння встановлювати “прямі” та “зворотні” оригінальні зв’язки при узагальненні; легко перебудовувати систему знань, умінь та навичок при зміні умов дії; легко переходити від одного способу дії до іншого, уміння виходити за межі звичного способу дій (гнучкість мислення) тощо.

Формуванню більшості з наведених загальних інтелектуальних умінь у процесі вивчення математики присвячені роботи М.Я.Ігнатенка [100], В.А.Крутецького [125], І.Д.Пасічника [184], С.П.Семенця [221], З.І.Слєпкань [228] та інших. Використання НІТ у процесі інтелектуальної діяльності учнів на основі навчального предмету математики розглядається в роботах О.В.Вітюка [49], М.С.Голованя [64], О.Б.Жильцова [88], В.І.Клочка [113], Т.В.Підгорної [187], Є.М.Смірнової [230] та інших. Правильне оцінювання адекватності програмного засобу щодо проблеми, проведення обчислювального та графічного експерименту, раціональне використання засобів НІТ при розв’язуванні задач, все це теж інтелектуальна діяльність, яка передбачає наявність відповідних часткових інтелектуальних умінь, пов’язаних зі знанням принципів функціонування того чи іншого програмного забезпечення.

Вплив засобів сучасних інформаційних технологій (СІТ) на формування часткових і загальних інтелектуальних умінь учнів у процесі навчання інформатики висвітлюється в роботах Н.Р.Балик [17], І.С.Іваськіва [99], Ю.С.Рамського [205]. Зокрема приділяється значна увага зростаючій інформатизації суспільства і у зв'язку з цим необхідності внесення в масову практику загальної освіти процедур пошуку, обробки і гнучкого подання інформації, що утворюють основу інформаційних умінь, серед яких виділяються: уміння організовувати зберігання інформації у вигляді когнітивних структур (наприклад, семантичної мережі, фреймів); уміння організовувати пошук інформації; грамотно формувати запит; уміння структурувати інформацію; створювати інформаційну модель задачі, що розв'язується; з'ясовувати принципи побудови тієї чи іншої інформаційної системи. Програмування розглядається як засіб інтелектуального розвитку учнів в роботах І.М.Антіпова [10], М.З.Грузмана [70], А.П.Єршова, Г.А.Звенігородського, Ю.А.Первіна [83], В.М.Монахова [165]. Навчання програмуванню сприяє формуванню алгоритмічного мислення, уміння планувати структуру дій, необхідних для досягнення заданої мети за допомогою фіксованого набору засобів; уміння будувати інформаційні структури для опису об'єктів і систем; уміння, використовуючи структурний підхід, конструювати власну програму на основі окремих процедур.

Інтелектуальні уміння у процесі конструкторської діяльності визначаються в роботах В.О.Моляко [164], В.С.Лозниці [139]. Конструкторські уміння в основному пов'язуються з умінням створювати нові об'єкти за допомогою різних процедур: створення нового шляхом приєднання до іншого; ускладнення існуючого об'єкта або зведення до простого; роз'єднання об'єктів та їх функцій; об'єднання об'єктів або їх функцій; заміна одного вузла іншим; створення нового шляхом перенесення за аналогією принципу дії вже відомого об'єкта тощо.

У зв’язку з тим, що інтелектуальна діяльність учнів, перш за все, пов’язана з їх навчальною діяльністю, в педагогіці і психології приділяється особлива увага формуванню навчальних інтелектуальних умінь. “Справа в тому, - пише Т.І. Шамова, - що процес ефективного навчання можливий лише в тому випадку, коли учні будуть володіти інтелектуальними і загальними учбовими уміннями” [276, с.73]. Питання, що стосуються формування навчальних інтелектуальних умінь розглядаються в роботах Ю.К.Бабанського [13], Д.Н.Богоявленського [27], О.М.Кабанової-Меллер [104], Н.О.Менчинської [157], Т.І.Шамової [276]. Вивчаючи можливості раціональної організації навчальної діяльності, Ю.К.Бабанський виділяє наступні групи загальнонавчальних умінь: навчально-організаційні (уміння визначати завдання діяльності, раціонально планувати діяльність, створювати сприятливі умови для діяльності); навчально-інформаційні (уміння здійснювати бібліографічний пошук, працювати з книгою, довідником, працювати з технічними джерелами інформації, здійснювати спостереження); навчально-інтелектуальні (уміння мотивувати свою діяльність, уважно сприймати інформацію, раціонально запам’ятовувати, логічно осмислювати навчальний матеріал, виділяючи в ньому головне, розв’язувати проблемні завдання, самостійно виконувати вправи, здійснювати самоконтроль в навчально-практичній діяльності). Досліджуючи проблему активізації навчання учнів, Т.І.Шамова по-своєму називає три групи навчальних умінь: інтелектуальні уміння (уміння здобувати інформацію, виконувати мислительні операції: аналіз, синтез, порівняння, узагальнення тощо); загальні навчальні уміння (уміння здійснювати процес самокерованої діяльності); спеціальні уміння.

Аналіз масової шкільної практики показує, що в початкових і середніх класах в основному відбувається формування в учнів загальнонавчальних умінь, які є запорукою повноцінного засвоєння знань і способів діяльності. Юнацький вік старшокласника характеризується прагненням до самостійної

діяльності, знайти своє місце у певній професійній діяльності, намаганням утвердити себе як особистість. Не випадково серед старшокласників становиться помітнішим явище – самостійне набуття знань зовні школи (читання додаткової літератури, пошук інших джерел інформації тощо). Але темп зростання з віком потреби в самостійності і дорослості в більшості випадків не збігається з темпом утворювання необхідних для цього способів самостійної навчальної роботи [159, с.176] та рівнем сформованості відповідних загальних інтелектуальних умінь учнів, хоч умови для їх утворювання і формування значно поліпшуються. Довільна, навмисна увага перетворюється в постдовільну, виникає звичка працювати уважно. Зростає оволодіння прийомами запам'ятовування, які стають все більш усвідомленими, різноманітними та гнучкими. Поліпшується усвідомленість власних навчальних дій, розуміння їх послідовності та необхідності їх планування, управління ними (саморегуляція в навчанні). Характерною є поява так званих перспективних ліній. Старшокласники намагаються дізнатися, для чого необхідно виконувати те чи інше завдання, вивчати ту чи іншу тему. Їм необхідно, щоб за шкільною буденністю відкривалися обрії їх майбутньої професійної діяльності.

Незважаючи на бурхливе зростання самосвідомості, вона залишається обмеженою, стосується головним чином окремих якостей особистості. Без допомоги дорослого старшокласнику важко оцінити себе в цілому, у співвідношенні різних сторін свого розвитку. Це і є причиною того, що відчуття дорослості випереджує становлення справжньої зрілості. Не руйнуючи це відчуття дорослості, а підкріплюючи його, рекомендується [159, с.180] розвивати реальні якості дорослості, спираючись на зазначені вище переваги юнацького віку, враховуючи орієнтацію старшокласника на однолітка, та його намагання уявити свою майбутню діяльність. У зв'язку з цим необхідно навчати старшокласника способам самостійного набуття знань, способам колективної навчальної роботи, що проводиться разом з

товаришами, способам включення результатів навчання до майбутніх видів діяльності. Окрім цього, становлення творчої особистості, здатної самостійно діяти і приймати рішення у динамічних, нестандартних ситуаціях, створення основ для подальшого розвитку учнів в різних галузях людської діяльності, постійне збільшення обсягу інформації, в якій вони повинні орієнтуватися, вимагає більш широкого впровадження у навчальний процес методичних компонент формування їх загальноінтелектуальних та інформаційних умінь. Наявність в учнів конструкторських умінь може стати в майбутньому запорукою творчого підходу в їх професійній діяльності.

1.2.2. Поетапна модель формування інтелектуальних умінь учнів

Якщо уміння пов’язується з самостійною і свідомою дією для практичного чи теоретичного застосування набутих знань, то формування уміння слід розглядати як формування цієї дії, тобто засвоєння методів її реалізації у процесі діяльності. У зв’язку з цим в основу пропонованих в нашій роботі компонент методичної системи формування інтелектуальних умінь нами покладено теорію діяльності, розроблену С.Л.Рубінштейном [211] та О.М.Леонтьєвим [136], а також запропоновану П.Я.Гальперіним концепцію поетапного формування розумових дій.

Згідно із зазначеною вище теорією діяльності дії, які реалізують діяльність, спонукаються її мотивом, але є спрямованими на мету. Дія, яка здійснюється, відповідає певному завданню, яке є метою, поставленою в певних умовах. Одна й та сама мета може ставитися в різних умовах, а отже, і дія, спрямована на її досягнення буде мати іншу якість, іншими словами, буде виконуватись іншими способами. Способи здійснення дій називають операціями або прийомами. Дія може виникнути з діяльності, яка отримала свій мотив; в свою чергу, вона сама може “трансформуватись” в спосіб досягнення, тобто в операцію або прийом; і навпаки дія може набути

самостійної спонукаючої сили і стати окремою діяльністю, а операція може обслуговувати іншу дію [133, с.21].

Окрім потреби, мотиву і цілі, до структури дії включаються ще орієнтовна, виконавча, контролююча та коригувальна частини. До орієнтовної частини відносяться операції усвідомлення умови задачі, пригадування і вибір способу дії, засобу тощо. Виконавчу частину становлять операції виконання, що забезпечують розв'язування задачі, здійснення дії. Контролююча частина вміщує операції перевірки результату діяльності на відповідність еталону. Операції коригувальної частини передбачають повернення до орієнтовної та виконавчої частин в залежності від виявлених помилок при здійсненні контролюючих операцій. Кожна з зазначених частин дії має власну специфічну функцію. Орієнтовними операціями визначається розумність і правильність виконуваної дії, а також швидкість включення в роботу, виконавчими - точність і якість цієї дії, контрольними і коригуючими – її усвідомленість [19, с.81]**.** Провідна роль відводиться орієнтовній частині дії, тобто сукупності тих операцій, завдяки яким відбувається виділення і розпізнання властивостей ситуації, які враховуються у виконавчій частині, і завдяки яким можна безпомилково виконати дану дію.

Для вдалого виконання дії в різних умовах людина повинна змінювати склад її орієнтовної частини, що дозволяє визначити нові властивості ситуації. В протилежному випадку задача залишиться нерозв'язаною. Операційна структура конкретної дії формується на основі попереднього засвоєння деякого розумового прийому. Він виконує функцію засобу побудови орієнтовної частини дії, який співвідноситься з різноманітністю умов свого застосування. Тобто у складі уміння лежить синтез побудови дії з прийомом-засобом [69, с.61].

Окрім результату використання уміння в нових ситуаціях, про якість сформованості уміння буде свідчити як ступінь його автоматизації, тобто швидкість виконання дії, так і кількість правильно виконаних операцій за

встановлений час. Достатня кількість часу при визначенні якості сформованості уміння за останнім показником буде надавати рівні можливості при тестуванні учням різних темпераментів.

Параметр сформованості уміння (автоматизації дії) в більшості залежить від виконання орієнтовних операцій. Чим краще побудовано орієнтування в ситуації (краще засвоєна вихідна інформація), тим меншу кількість ознак необхідно сприйняти учню, щоб правильно вибрати методику і зробити виконавчі операції дії. Коли досягається автоматизованість, дія виконується при згорнутій і скороченій орієнтовній частині та при виключенні коригування за рахунок безпомилкового виконання дії, так як інші частини дії (виконавча і контролююча) практично не підлягають згортанню.

Описана будова дії в плані системних уявлень характеризується психологами як базова система діяльності, тобто інваріант, загальний для будь-якої конкретної форми прояву [184, с.7]. Виходячи з інваріанту базової системи діяльності, існує структурна ізоморфність практичної діяльності і внутрішньої, мислительної. Причому в онтогенезі остання є вторинною зовнішній діяльності. Цей факт дає можливість з одного боку проаналізувати розумові дії і операції, а з другого – формувати їх у процесі навчання[184, с.8].

Виконаний П.Я. Гальперіним і його співробітниками аналіз генезису розумових дій дозволив певною мірою досліджувати перехід від зовнішньої дії з предметом до згорнутого психологічного акту, від зовнішніх предметних засобів до внутрішніх знакових. При цьому етапам зовнішньої матеріалізованої дії на початкових стадіях засвоєння дії ставляться у відповідність етапи внутрішніх процесів на стадії сформованої дії, згорнуті до одномоментного акту, який зберігає той самий зміст, що й розгорнута дія з предметом.

Результатом дослідження П.Я. Гальперіна стала концепція поетапного формування розумових дій і розроблені на цій основі конкретні методики, які

дозволяють активно її використовувати у навчанні. Процеси формування у загальному вигляді протікають таким чином:

- на першому етапі дія мотивується, тобто включається до структури певної діяльності і спрямовується на певну мету;
- на другому етапі складається схема орієнтовної основи дії, тобто даються умови досягнення мети. В ході засвоєння дії ця схема перевіряється і уточнюється;
- третій етап – формування дії в матеріальній (матеріалізованій) формі.
- на четвертому етапі відбувається послідовне “зменшення” зовнішньої звукової сторони мови;
- на заключному етапі в свідомості залишається тільки кінцевий результат – предметний зміст дії.

На кожному етапі дія виконується спочатку розгорнуто, деталізовано, а потім поступово скорочується, згортається.

В залежності від типу орієнтовної основи дії існують різні підходи до створення її схеми. В психології в основному розрізняють три типи орієнтовної основи (частини) дії, які визначають результат сформованості дії, а відповідно і уміння [56].

До орієнтовної основи першого типу відносяться зразки дії та її продукт. Учням вказівки щодо виконання дії не надаються. Вони самі шукають шляхи виконання завдання “наосліп”, методом проб і помилок. В наслідок таких пошуків завдання може бути виконаним, але дія, за допомогою якої воно виконується залишається нестійкою, не відтворюється при зміні умов, при перенесенні в нові умови.

Орієнтовна основа другого типу містить не тільки зразки дії, але й усі вказівки щодо правильного виконання дії з новим матеріалом. У цьому випадку навчання протікає швидко і без помилок. Учень набуває певного уміння аналізувати матеріал з позиції майбутньої дії. Остання виявляє стійкість при зміні умов і переноситься на нові завдання. Однак цей перенос

обмежений. Він відбувається при наявності у складі нових завдань елементів вже засвоєних знань.

Орієнтовна основа третього типу не подається учням в готовому вигляді. Вони мають скласти її самі. При навчанні за цим типом орієнтування вчитель повинен створити такі умови, які спонукають учня самостійно складати орієнтовну основу дії і потім діяти за нею.

В дослідженнях Н.І.Білоконної [22, с.34] звертається увага на те, що навчання за третім принципом орієнтування дещо складніше і на перших кроках потребує стільки ж часу і навіть трохи більше, ніж при навчанні за другим типом. Але наступні завдання виконуються одразу правильно і цілком самостійно. При достатньо великій кількості завдань темп навчання різко зростає, до того ж учні роблять значно менше помилок. Сформована у такий спосіб дія виявляє властивості переносу на широке коло завдань.

Інваріантність структури дії та існування концепції поетапного формування розумових дій П.Я.Гальперіна, підтвердженої в інших психолого-педагогічних дослідженнях [91], доводять практичну можливість формування умінь виконувати ці розумові дії, тобто формування інтелектуальних умінь.

Система формування інтелектуальних умінь, модель якої представлена на рис.1.1, грунтується на взаємодії вчителя і учня. Процес формування інтелектуальних умінь має поетапний характер, спирається на відповідні принципи та зміст навчального матеріалу і передбачає словесні, наочні, практичні методи роботи.

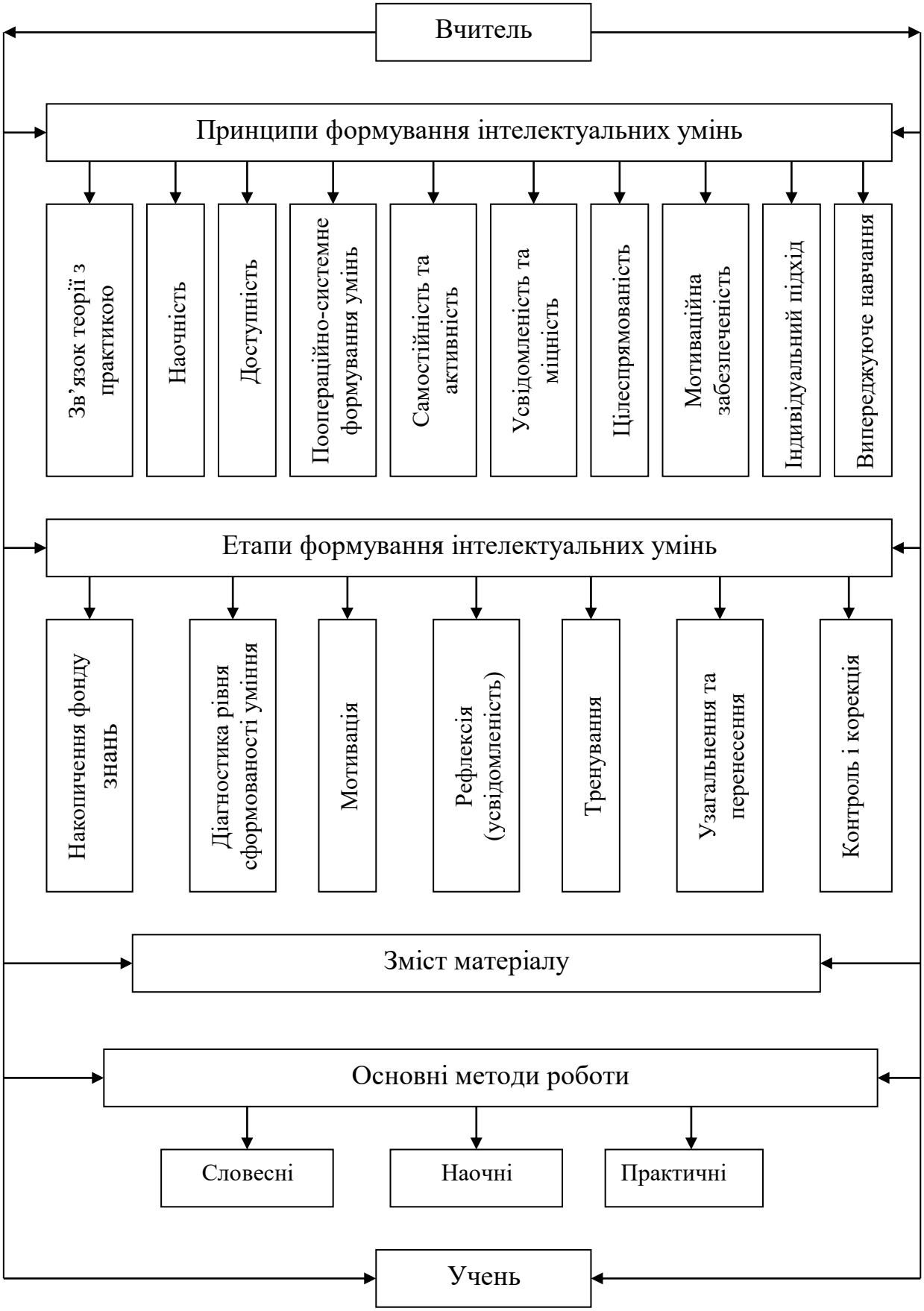


Рис.1.1 Модель системи формування інтелектуальних умінь.

Етапи процесу формування інтелектуальних умінь, наведені в моделі системи формування інтелектуальних умінь (рис.1.1), мають такий зміст [91, 103, 130, 182]:

1. Перший – це етап накопичення фонду знань як основи реалізації подальшого процесу розумового розвитку учнів. Акумуляція минулого досвіду учнів має мету підготовити їх до сприйняття нового.

2. Етап діагностики, на якому за допомогою діагностичних вправ визначається рівень сформованості умінь учнів. Вчитель визначає основну масу учнів, учнів, з якими потрібно працювати індивідуально, та “зону найближчого розвитку” для учнів, що випереджають основну масу.

3. Етап мотивації, на якому в учнів повинна створюватися атмосфера зацікавленості, позитивних емоцій, стійкого інтересу у процесі оволодіння прийомами розумової праці, наприклад, шляхом надання можливості успішно працювати з отримуванням позитивного результату.

4. Етап рефлексії, на якому відбувається усвідомлення прийомів уміння і правил їх реалізації.

5. Тренувальний етап, на якому певне вміння відпрацьовується у процесі репродуктивної діяльності та застосовується при виконанні творчих завдань.

6. На етапі узагальнення умінь має місце їх часткова автоматизація та перенесення прийомів в нові умови, на інші теми і предмети (ближнє перенесення), на інші сфери діяльності (далеке перенесення).

7. На етапі контролю і корекції здійснюється перевірка якості засвоєних умінь, наприклад, за допомогою завдань, розроблених в кількох варіантах (за рівнем складності та самостійності). Доцільно складати їх аналогічно до діагностичних вправ, щоб потім порівняти результати, побачити, чого досягнув кожен з учнів, спланувати подальше навчання.

1.2.3. Зміст основних загальних інтелектуальних умінь

Як зазначалося вище, під формуванням уміння розуміється засвоєння учнями способів (прийомів) виконання відповідної дії. Під прийомом будемо розуміти задану для засвоєння систему дій у вигляді правил, інструкцій, рекомендацій тощо [162, с.42]. Прийом містить дві компоненти: 1)знання, як треба діяти; 2) уміння користуватися цими знаннями, тобто володіння способом. “У процесі діяльності прийоми засвоюються учнями і набувають характеру інтелектуальних умінь” [182, с.4].

Для застосування індивідуального підходу у процесі формування інтелектуальних умінь вчитель повинен, перш за все, визначити рівень володіння способами інтелектуальної діяльності для кожного учня. “Як правило, в роботі виділяються три групи учнів. До першої групи належать учні, які не знають суті і структури прийому, але можуть його відтворити і застосувати під керівництвом вчителя. До другої групи належать учні, які частково знають суть і структуру прийому, вміють його застосувати, але не завжди правильно і свідомо. Третя група – це учні, які знають суть і структуру прийому, вміють його правильно і свідомо застосовувати. Відповідно це і є показники рівнів. Перший рівень ми називаємо репродуктивним, другий – конструктивним, третій – творчим (відповідно до трьох рівнів засвоєння знань)” [182, с.7].

Матеріал майже кожного навчального предмету дозволяє формулювати правила здійснення основних способів інтелектуальної діяльності. Намагаючись знайти раціональні методи навчальної праці, вчителі застосовують різні підходи щодо формування способів розумової діяльності, але в більшості вони мають емпіричний, стихійний характер. Незнання структури, прийомів розумової діяльності, раціональних шляхів її здійснення призводить до неякісного виконання багатьох завдань, а в результаті – до низької якості знань з багатьох предметів. Тому метою процесу навчання при

вивченні конкретного матеріалу повинно бути не тільки осмислення його змісту, а й засвоєння учнями основних способів інтелектуальної діяльності. Однак, сам по собі зміст навчання – без спеціального формування прийомів навчальної роботи – не може “автоматично” розвивати інтелект учнів. Він створює сприятливі передумови, можливості для формування мислення, а реалізувати їх покликаний вчитель за допомогою спеціальної методики, в основі якої повинна бути послідовність, систематичність, етапність, міжпредметність. Розробку компонент такої методики доцільно почати з розкриття структури і способів здійснення розумових дій, які засвоюються учнями у процесі діяльності і набувають характеру відповідних інтелектуальних умінь.

“Способи діяльності (прийоми) можуть бути простими і складними, розв’язувати тактичні і стратегічні цілі навчання. Наприклад, навчити прийому виділяти головне простіше, ніж навчити такому методу наукового пізнання, як моделювання” [182, с.4]. “Прийоми інтелектуальної діяльності не є рівноцінними і з точки зору вирішення різних задач: вони можуть бути цільовими, тобто грати головну роль, і допоміжними, обслуговуючими основний прийом” [182, с.4]. Деякі простіші прийоми можуть входити до складу складніших. Далі детальніше розглянемо стратегію формування конкретних прийомів інтелектуальної діяльності учнів.

Розумова дія “**визначення поняття”** привернула нашу увагу тому, що операції, виконання яких передбачається при її здійсненні, задіяні і в межах шкільного курсу інформатики при “**побудові інформаційної моделі об’єкта**” в процесі вивчення об’єктно-орієнтованого програмування, баз даних, експертних систем тощо. Поетапний процес формування інформаційної моделі адекватний процесу пізнання реального об’єкта, який позначається деяким поняттям.

В психології під поняттям розуміють “форму думки, в якій відображуються суттєві відмінності властивостей предметів і відношення між

ними. Поняття має зміст і обсяг: зміст – це сукупність властивостей і відносин реальних речей, а обсяг – множина (клас) предметів, кожен з яких має ознаки даного поняття" [182, с.8].

Найбільш розповсюдженим видом визначень понять є визначення через рід і видові відмінності. При цьому схема поняття має вигляд: поняття, яке визначається = найближчий рід + видові відмінності. Якщо у процесі навчання логічне визначення через рід і видову відмінність не передує формуванню понять, то поряд із зазначенням, поясненням, характеристикою складових частин і їх функцій, використовуються такі логічні прийоми, як порівняння одного предмета з іншим, виділення суттєвих і несуттєвих ознак.

За аналогією з поняттям зміст інформаційної моделі об'єкта при оголошенні складається з атрибутів, які задають його основні властивості, і методів, що визначають характерні дії над об'єктом або його функції. Обсяг визначається множиною об'єктів, що входять до заданого класу. Кожний об'єкт успадковує характерні особливості батьківського класу. Успадковані властивості належать до родових ознак, а видові відмінності задаються через значення атрибутів.

Під створенням інформаційної моделі об'єкта в об'єктно-орієнтованому програмуванні розуміють процес створення абстракції, тобто виконання розумової дії **абстрагування**. Абстрагування передбачає "виділення суттєвих для спостерігача характеристик деякого об'єкта, що відрізняють його від всіх інших видів об'єктів, випускаючи ті характеристики, які на даний момент несуттєві" [35, с.55]. Виконання абстрагування передбачає наявність дій: аналізу, порівняння, виділення суттєвого, узагальнення тощо. За способами виконання розрізняють три виду абстракції: 1) ізолююча абстракція виділяє деякий елемент з інших елементів; 2) підкреслююча абстракція не тільки виділяє один елемент, а й вказує на інші, що служать фоном для виділеного елемента; 3) абстракція протиставлення полягає у розділянні суттєвого і несуттєвого [227, с.23].

Для моделювання складного процесу функціонування об’єктів в реальному середовищі виділяються наступні типи абстракцій: реальні об’єкти– абстракції фактичного існування деяких предметів у фізичному світі; ролі – абстракції мети або призначення людини, частини обладнання або організації; інциденти – абстракції подій, що трапились; взаємодія – абстракція об’єктів, що отримуються в результаті відношень між іншими об’єктами.

Таким чином, визначення поняття або створення абстракції об’єкта передбачається виконання наступної послідовності дій: 1) виділення суттєвих і несуттєвих ознак; 2) визначити, які ознаки є загальними для роду, а які є характерними тільки для цього виду; 3) закріплення поняття або створення об’єкта у процесі практичних дій.

Формування умілого виконання розумової дії “**виділення головного**” є однією з перших задач в озброєнні учнів “азбукою” інтелектуальної праці. Головне – це предмет думки, сутність даної інформації. Виділення головного включає в себе більшість мислительних операцій: аналіз і синтез, абстрагування і узагальнення, порівняння і конкретизацію. Це один із способів фіксації навчального матеріалу в довгостроковій пам’яті, що сприяє його засвоєнню. Закономірністю запам’ятовування та збереження інформації є “ущільнення” (запам’ятовування ключових положень, на основі яких можна встановити всю логіку предмета), “конденсація”. Виділене і спеціально зафіксоване головне міцно і надовго зберігається в пам’яті учнів і може бути при необхідності знову “розгорнуто” – з деталями, аргументами, всім “фондом”. Запам’ятовуючи головне в одному предметі, можна значно збільшити обсяг знань з інших предметів, що полегшує встановлення міжпредметних зв’язків, формування наукового світогляду. Такі знання мобільні, їх легше застосовувати в нестандартних умовах. Уміння свідомо перебудовувати знання, “згортати” і “розгортати” їх, є властивістю творчого, самостійного мислення, розвиненого інтелекту. Для опрацювання текстової

інформації в літературі [182, с.15-18] виділяється наступна послідовність дій: 1) визначити предмет думки; 2) знайти ключові слова і поняття; 3) відділити головне від другорядного; 4) зафіксувати смислові опорні пункти; 5) за цими пунктами коротко запам’ятати головне.

В межах шкільного курсу інформатики уміння виділяти головне є надійною основою не тільки для роботи з науковими і науково-популярними джерелами, а й для створення власних творчих програмних проектів, виконання моделювання, що передбачає виділення головних структурних компонент об’єкту моделювання, їх функцій, закономірностей поведінки тощо. У зв’язку з цим, при виділенні головного необхідно виконувати наступну послідовність дій: 1) визначити предметну галузь функціонування об’єкта; 2) визначити його функціональні дії; 3) відділити суттєві і несуттєві для проекту функціональні дії; 4) визначити властивості, параметри об’єкта, важливі для виконання суттєвих функціональних дій.

Важливість загальної розумової дії “**порівняння**” визначається тим, що вона є основою усіх методів творчої діяльності. Порівняння – це встановлення схожості і відмінності. Предметами порівняння є об’єкти реальної дійсності, їх якості, ознаки, а також факти, явища, події, процеси, методи і прийоми роботи тощо. При цьому психологи рекомендують [182, с.19-25] завжди дотримуватись певних вимог до об’єктів порівняння:

1) порівнювати можна лише однорідні об’єкти, що відносяться до одного класу, що мають певну загальну основу;
2) спільне в об’єктах порівняння можна встановлювати лише в тому випадку, коли їх щось відрізняє, а встановлювати різницю між ними лише за наявності в них певної подібності (схожості);
3) нескладні об’єкти, факти порівнювати легше, ніж якості, ознаки, процеси та категорії. Тому об’єкти порівняння треба ускладнювати поступово, порівнювати краще починати з двох об’єктів, а потім поступово збільшувати

їх кількість. При порівнянні нескладних об’єктів корисно вводити третій, контрастний об’єкт, використовувати також і наочність.

Порівняння має відбуватися лише за суттєвими ознаками. Воно завжди цілеспрямоване, тобто відбувається під певним кутом зору. При здійсненні порівняння не слід обмежуватися виявленням лише загальних рис порівнюваних об’єктів. Невиконання паралельного пошуку відмінних, протилежних ознак знижує роль порівняння у процесі пізнання і розвитку мислення учнів. Невміння порівнювати в обох напрямках вказує на недостатню гнучкість та різнобічність їх мислення. Результатом порівняння може бути умовивід за аналогією, що широко застосовується у точних науках. На основі порівняння часто здійснюється узагальнення і систематизація.

За ступенем повноти розрізняють повне і часткове порівняння. Перше вимагає встановлення як подібності, так і відмінності, а друге – тільки подібності і тільки відмінності. При цьому, коли з’ясовуються тільки ознаки подібності в об’єктах, ми маємо справу з їх співставленням; якщо з’ясовуються лише відмінності, то це буде протиставлення. В плані послідовності виконання порівняння, людина спочатку співставляє об’єкти, ніби то накладаючи один на одного, а потім знаходить їх відмінності. Порівняння розрізняють не лише за ступенем повноти, але й за способами їх здійснення. Вони можуть бути паралельними, послідовними і відстроченими.

Правило-орієнтир прийому порівняння формулюється наступним чином: 1) встановити мету порівняння; 2) перевірити, чи відомий матеріал про об’єкти, які порівнюються; 3) виділити головні ознаки, за якими буде здійснюватися порівняння; 4) знайти відмінності і (чи) подібності; 5) зробити висновок з порівняння.

Сформованість уміння порівнювати рекомендується визначати в залежності від повноти порівняння та характеру встановлених ознак [182, с.21] за наступними рівнями: 1) рядоположний опис об’єктів; 2) неповне порівняння, коли вказується або подібність, або відмінність, але за всіма

подібними ознаками; 3) повне порівняння, коли учні проводять його за системою суттєвих ознак відмінності і подібності, вказуючи мету і висновок; 4) перенос вміння порівнювати з даного навчального предмету на інші, коли порівняння стає узагальненим прийомом інтелектуальної діяльності.

Сформований прийом порівняння дозволяє приступити до цілеспрямованого формування уміння **узагальнювати**. Дидактична сутність узагальнення полягає у виділенні і об'єднанні найбільш загальних ознак, характеристик у формулюванні понять, законів, провідних ідей предмета. Узагальнення передбачає уміння аналізувати явища, виділяти головне, абстрагувати, порівнювати. Об'єктом узагальнення можуть бути властивості об'єктів, факти, події, явища, якості і ознаки, відношення, зв'язки і процеси. Але узагальнювати можна лише такі явища, між якими є подібність.

У зв'язку з тим, що "логіка пізнання складається з багатоступеневого циклу переходу емпіричного змісту фактів і результатів спостережень у теоретичний сенс логічних конструктів" [147, с.15], розрізняють два типи узагальнення: емпіричне і теоретичне. Емпіричне узагальнення стосується порівняння зовнішніх, безпосередньо даних ознак з метою виділення головної ознаки. Здійснюється воно формально-логічним способом підведення конкретних понять під більш широке, родове поняття. Правило-орієнтир емпіричного узагальнення містить наступні дії: 1) виділити головне поняття з даного завдання; 2) відібрати основні, типові факти із матеріалу даної теми; 3)порівняти їх між собою, визначити загальне, суттєве; 4) зробити висновки, наприклад, сформулювати основні тенденції поведінки об'єкта.

Теоретичне узагальнення здійснюється на основі аналізу, синтезу і руху від абстрактного до конкретного. При цьому розглядається генезис явищ, що вивчаються, або виділяється сутність явища, на основі якої здійснюється теоретичне узагальнення діалектичним шляхом, коли явища розглядаються в розвитку, взаємозв'язку та взаємозалежності. Для того, щоб зробити теоретичне узагальнення, правило-орієнтир може бути таким: 1) виділити

головне поняття із заданого завдання; 2) виділити основні характеристики, відносини у матеріалі, що розглядається; 3) проаналізувати конкретний матеріал під кутом зору сформульованих вихідних характеристик, прослідкувати еволюцію його розвитку; 4) зробити висновки, тобто сформулювати тенденцію, закономірність, закон.

Обидва типи узагальнення є доцільними у навчальному процесі, проте кожний з них має свою область застосування – в залежності від особливостей навчального матеріалу, віку і можливостей учнів. Якщо об'єкти нескладні і їх можна вважати статичними, доцільно формувати узагальнення емпірично, формально-логічним шляхом. Якщо предмети вивчення є складними, об'єкти– динамічними, узагальнення формуються теоретичним шляхом, що розкриває діалектику розвитку, взаємозв'язки в явищах. З віком підвищується здатність учнів до абстрактного, теоретичного мислення. Тому, якщо в молодших класах переважаючим типом в формуванні понять є емпіричний, то в старших класах роль діалектичного методу пізнання, теоретичних узагальнень значно підвищується.

В узагальненні знань і способів діяльності велика роль належить **систематизації**. Систематизація – мислительна діяльність, у процесі якої об'єкти, що розглядаються організуються у певну систему шляхом встановлення суттєвих зв'язків, які об'єднують ці поняття. Діалектично протилежною систематизації є **класифікація** - тобто розподілення об'єктів за групами, класами на основі встановлення подібності і відмінності між ними з метою створення певної системи (наприклад, ієрархічної) класів деякого обсягу. Ієрархія передбачає розташування, упорядкування абстракцій за їх рівнями. Вона дозволяє створювати складні типи об'єктів. Одним з видів ієрархії є концепція успадкування. Успадкування означає таке відношення між класами (відношення предок/ нащадок), коли один клас запозичує структуру, функціональну частину іншого класу. Якщо клас-нащадок успадковує тільки одного класа-предка, то говорять про просте успадкування.

При успадкуванні від кількох предків говорять про множинне успадкування. Успіх виконання дій, що пов'язані із створенням дерева ієрархії, напряму залежить від рівня сформованості в учнів вміння класифікувати об'єкти.

За дослідженнями Н.Ф.Тализіної [239, с.37] класифікація включає в себе наступні дії: 1) вибір основи для класифікації; 2) розділення множини об'єктів за даною основою на ті, що входять до обсягу даного поняття, і тих, що не входять; 3) побудова ієрархічної класифікаційної системи.

Зворотною розумовій дії узагальнення є **конкретизація**. Уміння конкретизувати якесь правило або принцип передбачає уміння їх застосовувати на практиці. За особливостями процесу і кінцевого результату конкретизації теж розрізняють два типи: емпіричну і теоретичну. Емпірична конкретизація здійснюється чуттєво-наглядними засобами, за допомогою наочності. Теоретична (мислительна) конкретизація - словесними засобами.

Для формування уміння конкретизації частіше використовують нестандартні задачі, які передбачають використання розглянутих законів. При цьому правило-орієнтир може бути наступним: 1) прочитати уважно умову задачі, виділити те, що відомо, і те, що невідомо; 2) згадати, який закон, теорія, правило відповідні для даного випадку; 3) якщо теорія відома, то задача вирішується із застосуванням даного запасу знань; 4) якщо теорія невідома, то спочатку вивчити її, закон або правило; 5) розв'язати дану задачу.

Більшість наведених інтелектуальних дій передбачає виконання **аналізу** і зворотної для нього операції **синтезу.** Якщо аналіз – це міркування в напряму від того, що необхідно знайти або довести, до того, що дано або вже встановлено раніше, то синтез – це міркування в зворотному напрямку. Аналіз служить для виявлення ідеї розв'язування. Дає ж розв'язок синтез, який спираючись на дані аналізу, показує як із даних і раніше встановлених тверджень знаходиться шукане [227, с.128]. Ці прийоми вважаються простими і входять до складу інших. Аналіз розуміється як розчленовування

елементів на складові, які потім досліджуються окремо. В залежності від глибини пізнання людиною суті предмета В.Ф.Паламарчук [180, 181] визначає три рівні аналізу: емпіричний, елементарно-теоретичний, структурно-генетичний. При першому рівні аналізу здійснюється розділення інформації на окремі частини. Другий рівень потребує проникнення в суть явищ, третій – вивчення генезису явищ, подій. Рівні аналізу тісно пов'язані з його видами: поелементним, цілісним, проблемним. Поелементний аналіз частіше здійснюється на емпіричному рівні (аналіз умови задачі). Цілісний аналіз потребує знання теорії, закону, охоплення явища в його зв'язках. Проблемний аналіз передбачає висунення проблеми або структурної генетично вихідної клітини, навколо якої розгортається пізнання суті явища.

Поряд з аналізом і синтезом, теж найважливішим методом пізнання є **аналогія**. При виконанні аналогії на основі подібності об'єктів в одних ознаках передбачають їх подібність і в інших ознаках. Аналогія існує в двох формах: асоціативній та логічній. На базі асоціацій за подібністю одна думка викликає іншу. В одних випадках така асоціація допомагає досягненню істини, в інших – заважає йому. Часто аналогія має велику переконливість, оскільки асоціація, що викликала думку у однієї людини, може викликати її і в інших. Але таку переконливість не слід розглядати як доведення.

Описанні загальні інтелектуальні уміння є складовою частиною більшості специфічних (часткових) умінь в різних предметних галузях. Формування саме цих умінь ми взяли за основу нашого дослідження тому, що із-за їх загального застосування є можливість прослідкувати, як формування спеціальних умінь впливає на формування загальних умінь, які розглядаються як психологічні категорії з різними психологічними методиками визначення.

Саме за цими психологічними категоріями характеризують інтелектуальний розвиток учнів, оцінюючи його за наступними рівнями:

І (найвищий) рівень передбачає сформованість умінь аналізувати, узагальнювати тощо, й здатність використовувати набуті вміння у нових ситуаціях;

ІІ рівень засвідчує сформованість певного кола умінь, як правило, вербального характеру, що дозволяють виконати більшість завдань, але недостатні для оволодіння окремими вміннями, не дають можливості успішно завершити всі завдання;

ІІІ (середній) рівень виявляють учні, в яких недостатньо сформовані вміння аналізувати та узагальнювати матеріал вербального характеру, але вони непогано працюють з наочним матеріалом, у відповідях наводять конкретні приклади;

ІV (умовно незадовільний) рівень характеризує погане володіння вміннями аналізувати і узагальнювати вербальний матеріал, відповіді будуються лише на конкретних прикладах;

V (найнижчий) рівень вказує на те, що відповідні вміння не сформовані ні для вербального, ні для наочного матеріалу, під час відповідей учнями наводяться лише окремі приклади [147, с.73].

1.3. Задача як засіб формування інтелектуальних умінь учнів

1.3.1. Задача як основа технології формування інтелектуальних умінь учнів

Щодо розробки технології, загальних закономірностей інтелектуальної, творчої діяльності в світовій літературі висвітлюються дві протилежні думки. Перше міркування, згідно з яким розробка методології інтелектуальної, творчої діяльності взагалі неможлива, спирається на те, що суть будь-якої творчої, наукової діяльності полягає саме у вирішенні нових проблем, де всі вже відомі нам закономірності і методи діяльності не можуть привести до позитивного результату. Успіх результату наукового дослідження залежить від інтуїції дослідника. Якщо інтуїцію розглядати як нелогічну і нераціональну форму мислення, то її механізм не можна розкрити в раціональних формах. Якщо ж під інтуїцією розуміти згорнутість логічних операцій та дій, за рахунок чого досягається величезна швидкість розумових дій [73, с.12], то стає можливим розкриття в логічній формі інтуїтивних механізмів творчої, наукової діяльності. Таке міркування обгрунтовує правомірність розробки технології інтелектуальної діяльності людини [145, с.142].

Педагогічна діяльність на всіх історичних етапах завжди була пов’язана з поняттям “вміння”. Її функція вбачалась в тому, щоб перетворити будь-які знання про певні об’єкти людської діяльності у такі вміння, завдяки яким ці знання дадуть змогу краще досягти поставленої мети даної діяльності [145, с.142]. У зв’язку з тим, що в сучасному суспільстві інтелектуальна діяльність людини вважається головною, то робота педагогів повинна бути спрямована перш за все на розробку і застосування на практиці технологій формування інтелектуальних умінь учнів.

Під технологією навчання будемо розуміти систему методів, організаційних форм, засобів створення, застосування й визначення всього процесу навчання і засвоєння знань, з урахуванням технічних і людських

ресурсів та їх взаємодії, яка ставить своїм завданням удосконалення освіти [66, с.331].

Провідну роль як засіб ефективного засвоєння навчального матеріалу, формування і діагностики рівнів сформованості інтелектуальних і спеціальних умінь учнів в більшості технологій навчання відіграє задача (мета, дана в певних умовах), виконуючи навчальну (формування в учнів системи предметних знань, умінь і навиків), виховну (формування в учнів пізнавального інтересу, навиків навчальної праці, моральних якостей тощо), розвивальну (розвиток мислення учнів, формування умінь ефективної інтелектуальної діяльності) і контролюючу (встановлення рівнів сформованості інтелектуальних умінь, сформованості пізнавальних інтересів, якості засвоєних знань тощо) функції.

Увагу багатьох вчених (В.А.Крутецького [125], Н.О.Менчинської [158], Д.Пойя [190, 191], З.І.Слєпкань [227] та ін.) привертала саме розвивальна функція задач, завдяки чому при їх розв'язуванні не тільки засвоювалися і закріплювалися знання під час практичного застосування, а й формувався в учнів дослідницький стиль розумової діяльності [227, с.124], тобто відповідні йому уміння виконувати інтелектуальні операції (аналіз, синтез, порівняння, абстрагування, узагальнення, конкретизація, встановлення і використання аналогій тощо), які становлять основу загального методу вивчення явищ. Ці загальні розумові операції поряд із спеціальними (підведення під поняття, розгортання умови, встановлення суттєвих зв'язків) повинні виконуватися учнями у процесі розв'язування будь-якої задачі.

Згідно з існуючою в дидактиці і психології точкою зору процес розв'язування задачі повинен складатися з наступних етапів [227, с.128]:

1) аналіз умови задачі;
2) пошук плану розв'язування;
3) здійснення знайденого плану, перевірка і доведення того, що отриманий розв'язок задовольняє вимогам задачі;

4) аналіз проведеного розв’язування.

Під аналізом умови задачі розуміється аналіз її формулювання, який перш за все передбачає виділення даних і вимог. Якщо в задачі кілька даних і вимог, то вони повинні бути розділеними на елементарні. На цьому ж етапі операція аналізу супроводжується операцією синтезу при співставленні даних і вимог з тим, щоб з’ясувати, чи достатньо даних для відповіді на питання задачі, для виконання вимог задачі, чи немає серед даних зайвих. Окрім цього, вивчення умови задачі, завдяки встановленню суттєвих зв’язків і підведенню під поняття її об’єктів дає можливість встановити якому типу належить ця задача. Використання розумової операції порівняння при встановленні факту належності об’єкта до поняття дозволяє перевірити наявність у об’єкта сукупності необхідних і достатніх ознак. У випадку, коли об’єкт не має хоч однієї необхідної ознаки, з’ясовується, що він не належить до даного поняття [227].

Якщо встановлений тип задачі або її підзадач, то на другому етапі процесу розв’язування задачі за аналогією з існуючими можна намітити план розв’язування задачі. В іншому випадку пошук методу розв’язування задачі продовжується застосування розумової операції аналізу. Задача розбивається на окремі підзадачі до тих пір, поки не з’ясується план розв’язування її елементарних підзадач.

Здійснення знайденого плану розв’язування задачі починається із задання даних і шуканого у вигляді окремих абстракцій (змінних, структурних типів даних), побудова яких передбачає виконання абстрагування, узагальнення і систематизації. На цьому ж етапі у процесі перевірки і доведення того, що отриманий розв’язок задовольняє вимогам задачі, застосовується і зворотна до абстрагування - розумова операція конкретизація. Про правильність вибраного методу розв’язування задачі буде свідчити очікуване значення результату після підстановки на місце абстрактних конкретних даних, які доцільно вибирати не тільки із діапазону

допустимих значень, а й проводити аналіз межових вхідних даних, застосовувати метод припущення про помилку.

На останньому етапі розв'язування задачі проводиться аналіз розв'язування з точки зору його раціональності, можливості узагальнення даної задачі. Обговорення плану пошуку розв'язування дозволяє виявити, які прийоми сприяли досягненню успіху. Все це допомагає учням узагальнювати виконувані розумові дії у відповідні мислительні прийоми, що становлять основу інтелектуальних умінь.

Одним із факторів, які впливають на ефективність досягнення мети, орієнтованої на формування в учнів умінь виконувати певні розумові дії, вважається розгорнутість або ступінь повноти інформації, явно заданої в умові задачі. В тексті умови повно поставленої задачі розгорнута вся необхідна для її розв'язування специфічна інформація. Розумові дії учня у процесі розв'язування такої задачі будуть носити репродуктивний характер, що пояснюється наявністю в нього стереотипів, вироблених при розв'язуванні попередніх задач того ж типу. Ступінь виявлення продуктивності мислення при розв'язуванні учнями задач буде залежати від ступеня їх згорнутості. Згорнутість умови задачі може бути пов'язана з невказуванням: теорії, на основі якої слід розв'язувати дану задачу (незадана теорія може бути відомою учню, або як об'єктивно невідома отримується у процесі розв'язування задачі); особливостей мови, на якій викладена задача; засобів, які необхідно застосовувати для її розв'язування; деяких необхідних об'єктів задачі і відношень між ними тощо. Більшість задач, що зустрічаються у навчальному процесі, - це задачі з неповно заданою умовою, але ступінь її згорнутості може бути різним [255, с.53-54].

За ступенем повноти інформації, наданої в тексті умови, виділяються наступні типи задач [92, с.86-87]:

1. В умові задач першого типу міститься готова відповідь на питання задачі. Учневі необхідно тільки виділити її і зрозуміти, що це і є відповідь на питання.
2. Текст умови задачі другого типу не містить готової відповіді на питання, але надає необхідний обсяг інформації для її виконання.
3. Текст умови задачі третього типу містить тільки частину необхідної інформації. Іншу частину інформації учень повинен знайти сам з інших джерел інформації, отриманих раніше знань, шляхом розмірковування тощо.
4. Текст умови задачі четвертого типу взагалі не містить інформації, необхідної для її розв'язування.

Розв'язування задач першого типу найменше впливає на інтелектуальний розвиток учнів. Послідовність розумових дій з об'єктами логічної структури умови такої задачі збігається з логікою розгортання навчального матеріалу під час його викладення. Усвідомлення елементарних зв'язків між словами і реченнями, що описують відповідні об'єкти задачі, сприяє виникненню в учня розуміння того, що окремий фрагмент тексту її умови є відповіддю на питання. Розумова діяльність у процесі розв'язування задач такого типу не тільки спирається на наявний обсяг знань про об'єкт пізнання і логіку його викладення, але й регламентована ними, що не призводить до нарощування знань учнів.

Розумова діяльність з розв'язування задач другого типу теж регламентована наявним обсягом знань про об'єкт пізнання, але не збігається з логікою викладання матеріалу і відносно незалежна від неї. Для матеріалу з будь-якого навчального предмету характерна ієрархія елементів, в якій вплив одних елементів на інші здійснюється через треті [233, с.17]. Ті ж самі елементи присутні в умові задачі, але учень сам повинен виявити їх і приховані зв'язки між ними. Передбачена умовою задачі продуктивність, самостійність дій при її розв'язуванні робить використання у навчальному

процесі задач такого типу з метою формування інтелектуальних умінь учнів доцільнішим, ніж задач попереднього типу.

Але найбільше сприяють активізації інтелектуальної діяльності учнів задачі третього і четвертого типів. Розв'язування таких задач передбачає виконання учнями розумових дій різного характеру:

1) репродуктивного, коли частково або повністю відсутню інформацію можна заповнити шляхом її відтворенні з інших джерел;

2) продуктивного, коли проводиться аналіз наявної інформації та актуалізація, логічна обробка раніше отриманих знань, коли недостатні елементи знань з'являються шляхом розмірковування і отримання тим самим вивідних знань;

3) творчого, коли характер мети і діяльності з її досягнення, які є змістом задачі, знаходяться в протиріччі з наявним обсягом знань про об'єкт пізнання, і для виконання завдання необхідна активна мислительна діяльність з елементами творчості для отримання вивідних знань вищого порядку, ніж передбачено продуктивною діяльністю.

Другим фактором, від якого залежить якість формування інтелектуальних умінь учнів у процесі розв'язування задач, вважається їх об'єктивна складність. Серед факторів об'єктивної складності визначаються наступні:

1) рівень дидактичної складності задачі - можливість її розчленування на простіші дидактично доцільні ситуації (прості задачі, ступінь опосередкування невідомого в яких дорівнює одиниці);

2) ступінь узагальненості ознак явищ і способів розв'язування задачі, що розглядаються;

3) рівень новизни знань, які потребуються для розв'язування задач;

4) доступність умови задачі, яка залежить від факторів: обсягу життєвого та загального навчального досвіду (відсутність або неповнота знань з інших предметів), рівня розвитку просторового мислення, ефекту перенесення

способу дій, сформованості умінь аналізувати умову задачі, сформованості загальних інтелектуальних та спеціальних навчальних умінь.

В роботі [172] виділяються низький, середній, підвищений і високий рівні складності навчальних задач. В задачах низького рівня складності передбачається застосування знань і інтелектуальних умінь за деяким зразком або у знайомій ситуації. Діяльність учнів з розв'язування таких задач спрямована перш за все на сприйняття знань, їх розуміння і запам'ятовування. Тому містить в собі інтелектуальні дії, пов'язані з описанням фактів, явищ і закономірностей, розділенням фактів і явищ на групи і класи, їх аналізом і порівнянням із зразком.

Задачі середнього рівня складності передбачають мислительну переробку готових знань і способів інтелектуальної діяльності учнів. До таких задач належать ті, в яких необхідно порівняти різні процеси, об'єкти, пояснити суть явищ і фактів. Інтелектуальна діяльність учнів у процесі їх розв'язування більше пов'язана з виділенням спільних ознак елементів об'єкта пізнання, узагальненням даної задачі.

Задачі підвищеного рівня складності орієнтовані на оперування фактами і виявлення зв'язків, що потребує від учнів перетворювальної діяльності при їх розв'язуванні. До цієї групи належать задачі на доведення, застосування знань та інтелектуальних умінь в новій ситуації, самостійне встановлення причин і наслідків, визначення явищ і закономірностей, виконання чого в більшій мірі пов'язується з аналізом і синтезом навчального матеріалу.

Розв'язування задач високого рівня складності потребує від учня здійснення пошукової, творчої діяльності, результатом якої може бути виявлення нового способу розв'язування задачі або нової інформації. Це можуть бути випереджаючі або попередні задачі, задачі на абстрагування, узагальнення, розгорнуте доведення тощо, при розв'язуванні яких передбачається застосування учнями всього комплексу інтелектуальних умінь.

Звісно, що вибір вчителем задач, які доцільно використовувати для формування інтелектуальних умінь учнів, залежить від тематики навчального предмета, типу уроку, рівня інтелектуального розвитку учнів, наявних навчальних засобів, характеру організації навчально-пізнавальної діяльності тощо. Тому кожний з перерахованих типів задач займає своє належне місце у процесі інтелектуального розвитку учнів.

1.3.2. Використання задач в навчально-пізнавальній діяльності учнів

Під навчально-пізнавальною діяльністю будемо розуміти систему дій (розумових і практичних), здійснення яких забезпечує засвоєння знань, оволодіння уміннями і навичками та їх використання при розв'язуванні задач (І.С. Якіманська, [284, с.19]). Будучи діяльністю соціальною і нормованою, навчальна діяльність отримує свій предмет зовні і здійснюється у процесі спілкування з ним. Тому структура навчальної діяльності включає, повторює, відтворює різні види людської діяльності і складається з трьох головних компонент: мотиваційно-орієнтовної, центральної робочої (виконавчої, операційної) та конкретно-оцінювальної [252]. Іншими словами, структурними компонентами навчальної діяльності є мотиви (соціальні, моральні, пізнавальні тощо), навчальні задачі, дії виконання, дії контролю (Д.Б.Ельконін [281]). Навчальна і пізнавальна діяльності взаємозалежні одна від одної. Методи пізнавальної діяльності (емпіричні та теоретичні [93, 147]) переносяться на навчальну, а активність в навчальній діяльності підвищує інтерес до пізнавальної. Тому їх поєднання у навчальному процесі значно сприятиме успішному розвитку особистості за законом: виконавча діяльність $\rightarrow$ активно виконавча $\rightarrow$ активно самостійна $\rightarrow$ творчо-самостійна [280, с.16].

Таким чином, активність і самостійність учнів відображають рівень та характер навчально-пізнавальної діяльності. Г.І.Щукіною навчально-пізнавальна активність визначається як особистісне утворення, що виявляє стан учня і його відношення до діяльності (увага, прихильність, жива

співучасть у загальному процесі, цілеспрямованість пізнавальних дій та їх доцільність, швидке реагування на зміну обставин діяльності, мобільність використання знань, умінь та способів діяльності, бажання розширювати, поглиблювати пізнавальну діяльність за рахунок різних джерел). Вона впливає на формування мети, усвідомлення мотивації та способів діяльності, продуктивність навчання, активізацію всієї навчальної діяльності, як підготовчий рівень до самостійності [280, с.18].

Самостійність пов'язується з ініціативою, пошуком різних шляхів розв'язування навчально-пізнавальних задач без допомоги вчителя. Вона зростає при зростанні рівня евристики в навчальній діяльності учнів, тобто при використанні евристичних прийомів (таких, що спрямовані на змістовний аналіз проблем) і зниженні рівня репродукції до цієї стадії, коли учні з невеликою допомогою вчителя самі працюють над змістом задачі. За наявним рівнем евристики виділяють різні види навчально-пізнавальної діяльності:

1) індуктивно–репродуктивний: відтворення учнем понять відбувається в процесі розгляду часткових випадків, розв'язування задачі здійснюється за планом вчителя;

2) дедуктивно–репродуктивний: відтворення часткових випадків у процесі розв'язування задач, коли використовується загальне положення;

3) узагальнено–репродуктивний: відтворення вивчених фактів шляхом виконання вправ;

4) індуктивно–евристичний: самостійне відкриття фактів у процесі розглядання часткових випадків;

5) дедуктивно–евристичний: відкриття часткових випадків при розгляданні загального;

6) евристичне узагальнення: вчитель створює ситуацію, в якій учень самостійно (з деякою допомогою) підходить до узагальнення;

7) індуктивно–дослідницький: дослідження різних феноменів через вивчення їх конкретних проявів;

8) дедуктивно–дослідницький: організація досліджень засобами дедуктивного розвитку навчального матеріалу (використання методу моделювання);
9) узагальнене дослідження: наявність ситуацій, дослідження яких призводить до узагальненого знання [218, с.32].

Погляд на процес навчання як на трьохрівневу ієрархічну структуру, яка може містити репродуктивний, творчо-репродуктивний, творчо-пошуковий або дослідницький рівні [12, с.79], використання відповідних рівням методів навчання передбачає застосування педагогами задач, різних як за типом умови, так і за ступенем складності. Навчальна діяльність учнів на репродуктивному та творчо-репродуктивному рівнях може включати індуктивно–репродуктивний, дедуктивно–репродуктивний та узагальнено–репродуктивний методи, а на творчо-пошуковому (дослідницькому) – різні види евристичних та дослідницьких методів.

Характер навчальної діяльності учнів на репродуктивному рівні навчання (відтворення учнями навчальної інформації) визначає доцільність використання задач з повно заданою умовою, умовою, що містить правильну відповідь, задач низької та середньої складності.

Більш розвинутий тип репродуктивної діяльності – творчо-репродуктивна на рівні творчо-репродуктивного навчання містить в собі елементи відтворення інформації та відомих прийомів діяльності, можливе перенесення останніх на достатньо широкий клас пізнавальних задач, які подібні до розглянутих або зовсім відмінні від них. В процесі творчо-репродуктивної діяльності учні намагаються використовувати відомі знання і прийоми в нестандартних ситуаціях і пропонують обгрунтовані оригінальні розв’язки [117, с.35]. Тому на цьому рівні навчання доцільно застосовувати задачі з неповно поставленою умовою, перетворювальних задач, задач середньої та підвищеної складності.

Необхідність відкриття нових знань найбільше активізує мислительні процеси учнів при творчо-пошуковому навчанні. Організація такого процесу навчання може передбачати групове та індивідуальне розв'язування задач, розв'язування варіативних та творчих (проблемних) задач, задач, умова яких містить або частину, або зовсім не містить необхідної для розв'язування інформації, задач підвищеної та високої складності. Процес розв'язування творчої (проблемної) задачі може намічатися учнями під керівництвом вчителя на основі створеної їм проблемної ситуації. В іншому випадку учні самостійно ставлять перед собою задачі та індивідуально розв'язують їх для вирішення проблемної ситуації [172].

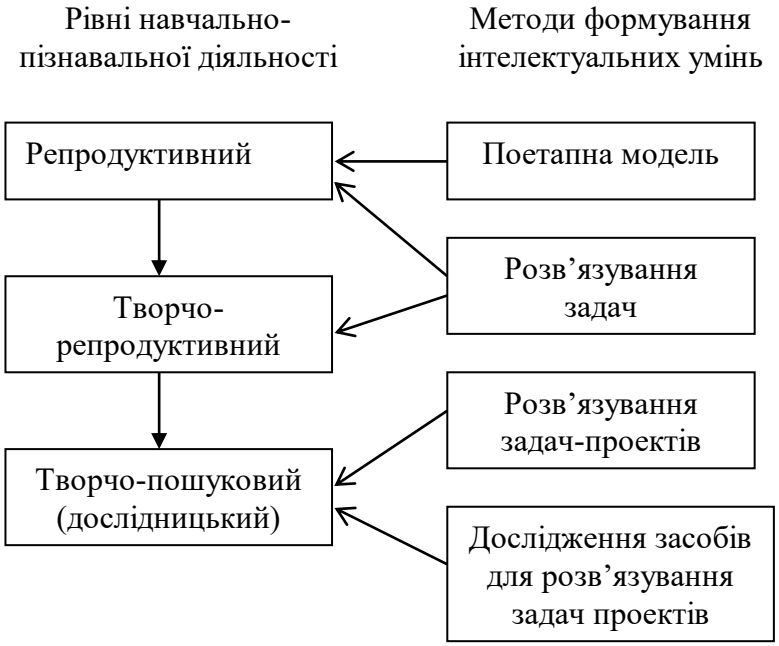


Рис.1.2. Схема процесу формування інтелектуальних умінь

Незважаючи на значні переваги творчо-пошукового навчання для реалізації мети формування інтелектуальних умінь учнів, до структури уроків вищого рівня, коли творче пізнання переважає над продуктивним, рекомендується переходити тільки через опрацювання системи попередніх уроків репродуктивного рівня [118], на яких учні повинні знайомитись з теоретичними основами предмету та прийомами виконання інтелектуальних

дій, необхідних для розв'язування задач. У зв'язку з цим розроблювана методична система формування інтелектуальних умінь учнів при навчанні інформатики передбачає застосування на різних рівнях навчально-пізнавальної діяльності учнів (рис.1.2).

1.3.3. Задачі, спрямовані на інтелектуальний розвиток учнів

Питання розробки задач з різних навчальних предметів, призначених для розвитку пізнавальної активності і розумових здібностей учнів, отримали доволі широку розробку в психолого-педагогічній літературі.

В загальнодидактичному аспекті ці питання розглядалися в роботах А.З.Зака [94], В.О.Оніщука [76], С.В.Лазаревського [130], О.І.Ляшенка [147], В.Ф.Паламарчук [182] та інших. Так, в роботах А.З.Зака [94], О.І.Ляшенка [147] проводиться аналіз методів розвитку теоретичного та емпіричного мислення. Згідно з [147] формування емпіричного та теоретичного мислення в процесі набуття знань повинно здійснюватись на грунті відповідних "двох психологічних механізмів: а) невмотивованому абстрагуванні та узагальненні на основі чуттєвого пізнання (емпіричні поняття і закони), б) активному пошуку сутності явищ шляхом побудови абстрактних моделей, формалізації, висунення і перевірки гіпотез про істотні ознаки даного класу понять (теоретичні поняття і закони, теорії)" [146, с.4].

Пропонуючи вимоги для конструювання завдань, наприклад, для формування теоретичних умінь А.З.Зак [94] наводить наступні: 1) завдання повинно включати кілька задач; 2) задачі повинні бути одного типу; 3) умови задач повинні відрізнятися за особливостями їх даних, що безпосередньо спостерігаються; 4) умови наступних задач (для ускладнення розв'язку) повинні включати більше несуттєвих обставин, ніж умови попередніх задач; 5) задач однакового ступеня складності повинно бути не менше двох (це дозволяє уникнути випадковості при оцінюванні способу розв'язування задач).

В роботах В.Ф.Паламарчук [180, 181, 182] наводяться правила-орієнтири виконання загальних інтелектуальних операцій і методи формування відповідних інтелектуальних умінь на прикладах уроків з різних навчальних предметів.

З метою розвитку пізнавальної активності і розумових здібностей учнів були розроблені:

1) система навчальних задач природничо-математичного циклу в старших класах, побудована з урахуванням основних дидактичних принципів (відповідності навчальному матеріалу, врахування дидактичної складності задачі, врахування складності навчальних задач) (Л.М.Павская, [178]);

2) система пізнавальних завдань з метою формування експериментальних, практичних, інтелектуальних умінь, побудована на основі дослідницьких завдань для лабораторних та практичних робіт (О.І.Забокрицька, [91]);

3) системи різних видів самостійних робіт для використання на уроках різних типів (Н.Ю.Лейкіна, [131]);

4) системи завдань міжпредметного характеру (Н.Р.Балик [17], Ю.С.Рамський [205, 206], Н.М.Рахманіна [207], Л.А.Хоміч [260]) та інтегровані завдання (А.М.Ясінський [286]);

5) системи завдань з математики для формування загальних інтелектуальних умінь (аналізу, синтезу, узагальнення, класифікації, порівняння, абстрагування, підведення під поняття, доведення) (З.І.Слєпкань [227]) та якостей мислення (гнучкості, згорнутості, раціональності, критичності) (В.А.Крутецький [125]);

6) системи завдань з застосування інформаційних технологій при розв’язуванні математичних задач з метою розвитку пізнавальної активності учнів та їх образного мислення (О.В.Вітюк [49], М.С.Головань [64], Ю.В.Горошко [67], М.І.Жалдак [87], О.Б.Жильцов [88], Є.М.Смирнова [230]);

7) системи завдань із застосування математичної логіки, теорії графів для розв’язування задач шляхом інформаційного моделювання та застосування

ППЗ (прикладного програмного забезпечення), зокрема СУБД, експертних систем для розв'язування задач прикладного змісту (Г.Ю.Цибко [266], Т.І.Чепрасова [272]) при навчанні інформатики;

8) система фізичних задач з динамічною структурою змісту [Циганок, 268] тощо.

При аналізі підручників і задачників [1, 28, 31, 44, 70, 71, 78, 79, 96, 98, 166, 194, 188, 214, 215, 271], що найчастіше використовуються вчителями при викладанні інформатики, нами були виявлені наступні типи задач:

1) в тексті умови повно заданої задачі міститься інформація про типи даних, операції над даними, про структуру алгоритму в термінології теоретично поданого матеріалу;

2) в тексті умови повно заданої задачі явно не вказується, які типи даних, операції над ними, структуру алгоритму доцільно використовувати для її розв'язання, тобто учню пропонується "упізнати" структуру, що вивчається в деякому текстовому неформальному описанні;

3) в умові задачі задається фрагмент програми, який необхідно оцінити з точки зору правильності його запису;

4) в умові задачі задається описання та фрагмент програми, в який необхідно внести відповідні зміни для отримання розв'язку описаної задачі;

5) тестові завдання, створені за правилами тестології, містять в умові правильну відповідь, яку необхідно вибрати із запропонованих, або шляхом встановлення співвідношень між відповідями;

6) варіативні задачі, в умові яких зазначені способи їх розв'язування;

7) конструювання власних прикладів ситуацій з участю структури алгоритму, що вивчається;

8) трансляція базових алгоритмічних структур з блок-схем до програмного коду і навпаки;

9) створення задач за блок-схемами з "порожніми" блоками і за псевдокодом, що містить тільки службові слова. Конструювання системи

завдань цього типу, передбачає поступове ускладнювання “логічного кістяка”, який учні повинні наповнювати реальним змістом. Такий підхід поступово готує учнів до складання програм методом покрокової деталізації;

10) задачі підвищеної складності (олімпіадні задачі), які передбачають варіацію, комбінування способів розв’язування, аналіз окремих випадків, про що в умові не повідомляється.

Проведений аналіз навчальної літератури показує, що в шкільній практиці поширене застосування на уроках інформатики задач з повно заданою умовою, низької складності (типи задач: 1, 2, 3, 4, 5, 8), середньої складності, як з повно (тип задач: 6), так і неповно заданою умовою (типи задач: 7, 9). Задачі підвищеного та високого рівня складності (олімпіадні задачі, дослідницькі роботи в межах МАН) розв’язуються учнями на факультативних заняттях або у процесі індивідуальної роботи.

Як передбачається процесом розв’язування будь-яких інших задач (1.2.4.), розв’язування вище наведених типів задач низького та середнього типу складності сприяє розвиткові алгоритмічності, критичності, логічності мислення, формуванню умінь виконувати розумові операції аналізу, синтезу, порівняння, узагальнення.

Але у процесі опрацювання педагогічної літератури з інформатики було виявлено, що недостатня увага приділяється розробці задач для розвитку такої важливої для високоінтелектуальної людини якості мислення як гнучкості. Маючи гнучкий розум, людина може легко переходити від прямих зв’язків до обернених, від однієї системи до іншої, відмовитись від усталених дій, якщо цього потребує розв’язування задачі. В [105] гнучкість мислення протиставляється інертності (ригідності), яка характеризується сповільненим мисленням, схильністю до стереотипних реакцій. Людина з ригідним мисленням неспроможна змінити спосіб дій, який перестав бути ефективним, навіть тоді, коли це можливо і доцільно. У зв’язку з цим в своїх роботах ми пропонуємо різні підходи до конструювання завдань з однієї чи кількох задач,

розв’язування яких спрямоване на розвиток гнучкості мислення, зокрема, завдання можуть включати:

1) задачі, що передбачають кілька способів розв’язування [37];
2) задачі, в яких незначна зміна в умові призводить до якісного перетворювання алгоритму розв’язування [38];
3) задачі, до яких можна побудувати обернені і розв’язати їх [38];
4) комбінацію задач різного типу [40].

Всі ці типи задач вимагають від учня обов’язкової перебудови процесу мислення, спрямовані на виявлення особливостей переключення від однієї розумової операції до іншої. В [37] розглядаються різні підходи до організації навчальної діяльності учнів на основі задач з кількома способами розв’язування. При розв’язуванні таких задач учень повинен самостійно знайти найбільшу кількість способів розв’язування задачі, намагаючись відшукати раціоніший. В роботі наводиться кілька способів організації роботи учня.

При першому способі, спочатку вчитель не ставить завдання знаходження можливих розв’язків задачі. Учень повинен просто розв’язати задачу. Якщо він не задоволений створеним алгоритмом, то в нього виникає потреба знайти найбільш простий, раціональний, який опрацьовував би більшу кількість вихідних даних.

При другому - перед учнем ставиться завдання – знайти найбільшу кількість різних способів розв’язування задачі. Якщо учень розв’язав задачу тільки одним способом або знайшов не всі можливі, то вчитель повинен повідомити йому їх. А учень вже має порівняти всі наведені способи розв’язування і вказати на той, який йому найбільше сподобався, мотивуючи свій вибір.

Наступний підхід дозволяє конкретніше розподілити учнів класу за рівнем розвитку гнучкості мислення. Задачі завдання поділяються на кілька груп, кожна з яких має певну функцію. Перші кілька задач установчі, вони

розв’язуються тільки одним деяким методом. Далі йде група діагностичних задач, що розв’язуються як попередньо розглянутим способом, так і більш простим. Якщо при розв’язуванні задачі учень замінить установчий спосіб на інший, значить він спонтанно виявить гнучкість мислительних операцій. Якщо ні, то дається “задача-підказка”, яка потребує іншого способу розв’язування. Розв’язавши “задачу-підказку”, учень усвідомлює, що попередню задачу можна розв’язати аналогічним способом, тим самим виявивши рівень викликаної гнучкості свого мислення.

Задачі, в яких незначна зміна в умові призводить до якісного перетворювання алгоритму розв’язування, та такі, до яких можна побудувати обернені і розв’язати їх, розглянуті нами в роботі [38]. Завдання, в якому передбачається зміна в умові задачі, складається з двох частин: вихідна задача та другий її варіант. В другому варіанті змінюється один з елементів першого варіанту, який зовні здається несуттєвим, але завдяки цьому змінюється зміст та розв’язування задачі. Учень повинен перебудувати свої дії стосовно розв’язування задачі у зв’язку зі зміною умови.

Прямі та обернені задачі дозволяють дослідити здатність до обернення процесу мислення, до переходу думки з прямого на зворотній хід. Учень повинен не тільки бути здатним зробити такий перехід, але й оцінити правильність такого переходу, встановити чи правомірне таке обернення. Таким чином він виконує цілу систему мислительних операцій: 1) аналіз прямої задачі (знаходження відомих та невідомих даних); 2) складання оберненої задачі (визначення кількості обернених задач, відомих та невідомих даних оберненої задачі); 3) перевірка правильності створення оберненої задачі (перевірка відповідності даних прямої та оберненої задач, встановлення факту використання відомих даних в прямій та оберненій задачах).

Використання задач різного типу в одному інтегрованому завданні, розглянуте в роботі [40], передбачає комбінацію наступних типів задач: основна, концептуальна задача (блок А); задача, аналогічна заданій задачі А

(блок Б); обернена до А задача (блок В); задача більшого рівня складності (блок Г); самостійно створена учнем задача за тематикою, що розглядається (блок Д). Виконувати задачі необхідно послідовно за схемою: А→Б→В→Г→Д, що зумовлено нарощуванням в них рівня складності та евристики.

Успіх формування загальних інтелектуальних умінь та гнучкості мислення учнів при розв'язуванні розглянутих типів завдань буде визначатися певними педагогічними умовами:

1) в процесі розв'язування задач навчальні і розумові дії повинні формуватися у тісному взаємозв'язку;

2) формування інтелектуальних умінь в процесі розв'язування задач, повинно проходити з врахуванням рівнів навчально-пізнавальної діяльності учнів;

3) побудова системи завдань для формування інтелектуальних умінь повинна проходити з урахуванням ускладнення і поглиблення розумової діяльності;

4) сформованість повноцінних інтелектуальних умінь можлива при умові формування в учнів установки на свідоме використання цих умінь в інших видах діяльності.

Формування інтелектуальних умінь, прийомів розумової діяльності шляхом розв'язування і навчання алгоритмів розв'язування розглянутих вище типів задач (це в основному задачі середнього і підвищеного рівнів складності) відповідає принципам діяльнісного підходу П.Я.Гальперина: при розв'язуванні основної і аналогічної задач відбувається складання схеми орієнтовної основи дії і її засвоєння, розв'язування інших задач в завданні передбачає здійснення ближнього перенесення виконання дії в нові умови. В процесі проведення експериментальної роботи ми впевнились, що цілеспрямоване управління розумовою діяльністю учнів у процесі розв'язування задач зазначених типів сприяє формуванню знань і умінь з визначеними якостями, забезпечує більш швидке оволодіння раціональними прийомами мислення. Як показує досвід, наведені типи завдань доцільно

використовувати на репродуктивному та творчо-репродуктивному рівнях навчально-пізнавальної діяльності. Слід відмити, що зазначений підхід до викладання інформатики в недостатній мірі передбачає надання учням необхідного простору свободи для прийняття самостійних рішень, творчості, індивідуального розвитку, як того потребує особистісно-орієнтоване навчання.

Ще півтора століття назад, закладаючи основу гуманістичної освітньої системи, М.В.Остроградський і І.А.Бум попереджували: “Чи треба під виглядом формування розуму і пам’яті з усіх сил мучити розум нескінченими вправами?” [177, с.47]. Доцільніше буде створити умови для індивідуально-творчого розвитку кожної особистості, залучення до цього процесу самої дитини. Для створення таких умов ми пропонуємо використовувати як при подачі нового матеріалу, так і в процесі самостійної роботи учнів, проблемні завдання – проекти моделювання функціонування деякої інформаційної системи з реального життя. Під проектом розуміється задача, що ясно описана, має кінцевий результат і практичну значимість.

Подібні задачі відносяться до типу задач з умовою, яка містить недостатню кількість або взагалі не містить необхідної проміжної інформації для її розв’язування. Як зазначалося в 1.2.4, розв’язування таких задач найбільше впливає на інтелектуальний розвиток учнів. Створюючи разом з вчителем проект від початку до кінця засобами, що вивчаються, на репродуктивному рівні навчально-пізнавальної діяльності учні набувають не тільки наукових знань з навчального предмету, а й засвоюють прийоми виконання різних інтелектуальних операцій (аналізу, синтезу, порівняння, виявлення суттєвих зв’язків між елементами інформаційної системи, абстрагування, узагальнення, систематизації тощо). Репродуктивний рівень може перетинатися з більш конструктивним творчо-репродуктивним, на якому вчитель має запропонувати учневі побудувати фрагмент проекту іншими засобами, оцінити раціональність або нераціональність такої заміни.

На творчому рівні учні створюють власні проекти, переносячи набуті знання і сформовані інтелектуальні уміння на новий матеріал, виявляючи творчий, дослідницький характер свого мислення.

Методика дослідницько-навчального проектування на творчому рівні реалізації передбачає, що учень самостійно вибирає тему проекту, який додатковий матеріал, програмне забезпечення він буде вивчати і застосовувати, складає графік реалізації проекту. Для старшокласника важливо почуття самостійності і відповідальності. Здійснюючи вибір теми того або іншого проекту школяр приймає на себе відповідальність за результат, а значить відповідальність за освоєння того або іншого навчального матеріалу. Розуміння практичної значимості досліджуваного матеріалу є потужним мотиваційним засобом.

Проектна організація курсу інформатики передбачає наявність наступних умов:

1) надання учням достатньо широкого набору тем проектів для реалізації можливості реального вибору. Слід зазначити, що проекти можуть бути як індивідуальними, так і колективними. Останні, окрім іншого, сприяють освоєнню учнем колективних засобів роботи, що безумовно важливо в сучасних умовах і відповідає характерній рисі юнацького віку – орієнтації на товариша;

2) у зв’язку з тим, що для учня важлива практична значимість отриманого ним результату і оцінка оточуючих, навчальний проект повинен передбачати для виконавця закінченість і цілісність виконаної ним роботи. Дуже важливо, щоб закінчений проект був презентований і отримав увагу інших учнів і дорослих;

3) створення умов, при яких учні мають можливість обговорювати один з одним свої успіхи і невдачі. При цьому відбувається взаємо навчання, яке обопільно корисне;

4) у зв’язку з тим, що реалізація проекту передбачає практичне поєднання різноманітного навчального матеріалу, навчальна програма з інформатики повинна містити не тільки описання необхідних теоретичних знань, але й набору засобів, якими необхідно буде оволодіти.

Робота над проектом висуває певні вимоги до організації навчального процесу. Значну частину часу учень повинен працювати самостійно. Він самостійно вибирає не тільки матеріал для вивчення, але і темп, режим своєї роботи. Найкращою формою для реалізації подібної позаурочної роботи є робота, наприклад, в шкільній комп’ютерній лабораторії. В ній учні готують свої проекти, знайомляться з тими або іншими можливостями програм. Присутність вчителя в цей момент не обов’язкова, а іноді – небажана. Роль вчителя зводиться до функцій консультанта, який не навчає, а допомагає в самостійній роботі. При цьому на уроках інформатики, що стоять в шкільному розкладі, доцільно проводити лекції, колективні обговорення ходу реалізації проекту, презентації готових результатів тощо.

Використання задач-проектів відповідає змісту особистісно-орієнтованої освіти, до якого належать аксіологічна, когнітивна, діяльнісно-творча та особистісна компоненти [29, с.14].

Аксіологічна компонента має на меті ввести учнів у світ цінностей і надати їм допомогу у виборі особистісно значимої системи ціннісних орієнтацій. Ще М.В.Остроградський і І.А.Бум констатували властивий і сьогоденній освіті факт, що “з п’ятдесяти учнів меншою мірою сорок відчували відразу і втрачали віру в себе через абстрактність ідей, що подавалися нам до того, як вони ставали зрозумілими на прикладах, із життєвої практики” [177, с.46]. Викладання наукового матеріалу на основі прикладу моделювання функціонування інформаційної системи з реального життя, “у найбільш пристосованому для дітей вигляді” [177, с.47] дозволяє запобігти виникненню подібної ситуації.

Когнітивна компонента змісту забезпечує учнів науковими знаннями про людину, культуру, історію, природу, ноосферу як основу духовного розвитку. Створення проектів в межах навчального предмета інформатики передбачає вивчення і глибоке дослідження учнями наукового матеріалу з тих наукових галузей, функціонування об'єктів якої моделюється, що сприяє набуттю наукових знань не тільки з інформатики, а й з інших навчальних предметів.

Діяльнісно-творча компонента впливає на формування і розвиток різноманітних способів діяльності, творчих здібностей, необхідних для самореалізації особистості в праці, в науковій, художній та інших видах діяльності. Навчання у процесі самостійної, активної творчої праці при створенні проектів дозволяє індивідуально підійти до інтелектуального розвитку учнів. Якщо раніше традиційна система навчання орієнтувалася на учня з "середніми розумовими здібностями", то діяльнісно-творчий підхід надає широкий простір без обмежень, що задаються умовою задачі, забезпечує "вивільнення" для творчості з метою розумового розвитку та творчої реалізації всіх учнів, особливо тих, які "окрилені науковим ентузіазмом і бажанням присвятити себе глибоким дослідженням" [177, с.47], гарантуючи їм особистісну свободу, можливість самопізнання, оволодіння способами саморегуляції, самовдосконалення, тобто реалізуючи особистісну компоненту навчання.

РОЗДІЛ 2. МЕТОДИЧНІ ОСНОВИ ФОРМУВАННЯ ІНТЕЛЕКТУАЛЬНИХ УМІНЬ УЧНІВ У ПРОЦЕСІ НАВЧАННЯ ІНФОРМАТИКИ

2.1. Формування інтелектуальних умінь учнів на основі технології об’єктно-орієнтованого програмування (ТООП)

2.1.1. Методологічна основа методичної системи формування інтелектуальних умінь учнів на основі ТООП

У зв’язку з тим, що метою нашого дослідження є формування інтелектуальних умінь учнів у процесі навчання інформатики, то в розроблюваних компонентах методичної системи будемо використовувати зміст цього навчального предмету, а також засоби, методи і організаційні форми його вивчення.

Як зміст для методичної системи ми вибрали технологію об’єктно-орієнтованого програмування, яка включає в себе: об’єктно-орієнтований аналіз (ООА), об’єктно-орієнтоване проектування і об’єктно-орієнтоване програмування. Під об’єктно-орієнтованим аналізом (ООА) розуміють таку методологію аналізу, при якій вимоги до системи сприймаються з точки зору класів і об’єктів, виявлених в предметній галузі. Об’єктно-орієнтоване проектування – це методологія проектування, яка поєднує в собі процес об’єктної декомпозиції та прийоми побудови логічної, фізичної, статичної та динамічної моделі системи, що проектується. Методологія об’єктно-орієнтованого програмування грунтується на поданні програми у вигляді сукупності об’єктів, які є екземплярами окремих класів, що утворюють ієрархію успадкування [35, с.49-54].

Методологічною основою застосування ТООП як навчального матеріалу, засобу формування інтелектуальних умінь учнів є метод моделювання. Під моделюванням розуміють дослідження яких-небудь явищ, процесів або систем об’єктів шляхом побудови і вивчення їх моделей. Моделювання є потужним методом пізнання зовнішнього світу, а також прогнозування і управління [152, с.574]. Метою моделювання є створення моделі (образу об’єкта для відтворення певних його характеристик), яка

виділяє необхідні для дослідження сторони об’єкта, відображаючи ознаки, факти, зв’язки, відношення в певній галузі знань у формі, зручній і доступній для аналізу і висновків [11]. “Ступінь відповідності моделі оригіналу може бути різним: подібність, аналогія, ізоморфізм (взаємооднозначна відповідність структур моделі і прототипу), гомоморфізм (узагальнена відповідність). Головні функції моделей – описувальна, конструктивна і евристична. Описувальна функція моделі в тому, що в досліджуваному об’єкті виділяються і узагальнюються суттєві компоненти і зв’язок між ними. Конструктивна функція моделі заключається в її здатності служити орієнтиром, застосовувати здобуті знання в нових ситуаціях. Евристична функція моделі сприяє прогнозуванню (наприклад, вивченої закономірності в нових умовах)” [182, с.47].

Таким чином, процес створення моделей і роботи з ними передбачає наявність у людини умінь виконувати розумові дії: аналіз, синтез, порівняння, узагальнення, виділення головного тощо. У зв’язку з цим ступінь оволодіння методом моделювання Л.В.Венгер вважав критерієм інтелектуального розвитку. Дослідження [99, 182, 242] та ін. підтверджують думку, що навчання учнів моделюванню розвиває їх інтелект і творче мислення. Окрім цього в [242] зазначається, що у старшому шкільному віці виникають всі необхідні передумови (прагнення до самостійної пізнавальної діяльності, більш високий рівень інтелектуального розвитку, поліпшені властивості пам’яті і уваги, накопичений фонд знань з різних навчальних предметів тощо) застосування моделювання як методу навчання.

Розрізняють фізичне, математичне, інформаційне, вербальне моделювання. Створення математичної моделі передбачає виконання опису об’єкта мовою математики.

Інформаційне моделювання є ланкою, яка, пов’язуючи розум і творчість людини і технічні можливості комп’ютера в єдине ціле, надає людині широкі можливості розв’язування найрізноманітніших задач з

використанням комп'ютера [267, с.103]. "Інформаційне моделювання – клас знакових моделей, які описують інформаційні процеси (виникнення, передавання, перетворення, використання інформації) в системах різноманітної природи"[163, с.5]. Побудова інформаційної моделі вирішує питання про вибір структур вхідних даних про об'єкт, що містять всю необхідну і достатню інформацію про досліджувані об'єкти чи процеси, та результатів розв'язування, про встановлення зв'язків між елементами даних, вибору способів їх обробки. "Уміння учнів самостійно створювати інформаційну модель об'єкта прирівнюється до уміння здобувати нові знання" [283, с.71]. Побудова інформаційних моделей підпорядкована певним крокам, які відображають ієрархію рівнів відтворення предметної сфери та пов'язані зі стадіями набуття нового знання:

1) виокремлення у відповідній предметній сфері об'єкта, який підлягає моделюванню, і присвоєння йому імені на основі наявної в учня особистісно-значущої інформації з приводу поведінки об'єкта;
2) аналіз поведінки об'єкта в різних ситуаціях та присвоєння змодельованому об'єкту на основі результатів аналізу певних абстрактних властивостей та логічних відношень;
3) визначення властивостей самих властивостей, що присвоєні;
4) уточнення істотних рис змодельованого об'єкта і виявлення його основних, суттєвих властивостей;
5) описання моделі об'єкта відповідними законами, що характеризують його поведінку;
6) коригування побудованої моделі, у процесі чого особистісно-значуща інформація вбудовується в систему знань учня і тим самим стає його знанням;
7) узагальнення результатів дослідницької роботи через порівняльний аналіз мети і досягнутих результатів [33, с.150-156; 283, с.71].

На основі математичної та інформаційної моделей будується комп’ютерна модель, що реалізується відповідними апаратно-програмними засобами, серед яких виділяють:

1) інструментальне використання базових програмних засобів (текстових редакторів, СУБД, табличних процесорів, телекомунікаційних пакетів);
2) комп’ютерне моделювання, яке передбачає обчислювальне (імітаційне) моделювання, “візуалізацію явищ і процесів” (графічне моделювання), використання “високих технологій” (спеціалізовані прикладні технології, які використовують комп’ютер, як правило, в режимі реального часу в поєднанні з вимірювальною апаратурою, датчиками, сенсорами тощо) [163, с.7].

Застосування моделювання є однією з складових, що забезпечують успіх програмного проекту. Чим більша і складніша система з предметної галузі, задана в умові задачі, тим більшого значення набуває моделювання при розробці її програми-проекту. Обмеженість у сприйнятті людиною сутностей не дозволяє їй сприйняти складну систему як єдине ціле. Для звуження проблеми у процесі моделювання, для загострення уваги людини в даний момент часу тільки на одному аспекті доцільно застосовувати принцип Едсгера Дейкстри “розділяй і володій”, за яким визначається, що складну задачу простіше розв’язувати, якщо поділити її на кілька менших. Основу цього підходу становить не що інше, як виконання однієї з найважливіших інтелектуальної операції аналізу. Окрім цього, моделювання посилює можливості людського інтелекту, оскільки правильно вибрана модель надає можливості створювати проекти на вищих рівнях абстракції (на рівні бітів, операторів алгоритмічної мови, систем візуального програмування).

Кожну систему з деякої предметної галузі можна описати з різних точок зору, для чого використовуються різні моделі, кожна з яких, відповідно, є семантично замкнутою абстракцією системи. Модель повинна включати елементи, які суттєво впливають на результат, і не включати ті, що не мають особливого значення на даному рівні абстракції. Виявлення суттєвих

елементів системи передбачає виконання інтелектуальних операцій аналізу, порівняння, виділення головного тощо. Модель може бути структурною, тобто підкреслювати організацію системи, або відображати її поведінку, динаміку.

Успіх вирішення проблеми шляхом моделювання залежить від дотримання наступних принципів [36, с.30-32]:

1. Правильний вибір моделі. Правильно вибрана модель висвітлить самі підступні проблеми розробки та дозволить проникнути до суті задачі. Неправильна модель може призвести до тупикової ситуації, оскільки увага буде загострюватись на несуттєвих питаннях. Так при створенні проекту бази даних основну увагу слід приділити моделям “сутність-зв’язок”. При структурному програмуванні в центрі моделі знаходяться алгоритми і передача даних від одного процесу до іншого. При об’єктно-орієнтованому програмуванні архітектура моделі грунтується на множині класів і зразках взаємодії, які визначають, як ці класи діють разом.
2. Реалізація моделі на відповідному рівні абстракції. При розв’язуванні задачі-проекту, її модель в різний період часу слід розробляти на різних рівнях деталізації.
3. Наближеність моделі до реальності. Оскільки модель завжди спрощує реальність, то таке спрощення може призвести до значних втрат, до суттєвої невідповідності моделі реальному об’єкту.
4. Використання майже незалежних одна від одної моделей, створених з різних точок зору, при розробці проекту системи з реальної предметної області. Так, для розуміння архітектури такої системи необхідно мати кілька різних моделей: з точки зору прецедентів, або варіантів використання (щоб виявити вимоги до системи); з точки зору проектування (щоб побудувати словник предметної галузі і галузі розв’язку); з точки зору процесів (щоб здійснити моделювання розподілу процесів і потоків в системі); з точки зору

реалізації (щоб розглянути фізичну реалізацію системи); з точки зору розгортання (для розподіленої на різних комп'ютерах системи).

Згідно з принципом Едсгера Дейкстри при створенні програмної моделі складної реальної системи необхідно розділяти її на менші підсистеми, кожна з яких потім незалежно удосконалюється. Процес розбиття складної системи на підсистеми називається декомпозицією. В програмуванні виділяють такі види декомпозиції: алгоритмічну і об'єктно-орієнтовану. Алгоритмічна декомпозиція відповідає принципам структурного програмування і концентрує увагу на порядку подій, що відбуваються. Кожний модуль системи виконує один з етапів загального процесу. При альтернативному об'єктно-орієнтованому способі декомпозиції особливе значення приділяється не подіям, а агентам, які є або об'єктами, або суб'єктами дії. Об'єкти щось виконують, і ми можемо послати їм повідомлення, попросити дещо виконати [35, с.34].

При дослідженні процесів створення учнями програмних проектів – моделей систем з реальних предметних областей на основі застосування об'єктно-орієнтованої декомпозиції, нами було виявлено виконання послідовності загальних інтелектуальних дій, описаних в 1.2.3.

Створення будь-якого програмного проекту починається з аналізу функціонування системи, розділення її на менші підсистеми, виділення можливих об'єктів предметної області, для чого виконується розумова операція аналізу на емпіричному рівні (не проникаючи в суть явищ). Виділення ознак, що характеризують об'єкт, визначення властивостей (атрибутів), поведінки, будови і функцій об'єкта передбачає проникнення в суть явища, тобто виконання учнями наступного рівня мислительної операції аналізу – елементарно-теоретичного. Дослідження генезису явищ, подій при моделюванні операцій взаємодії, встановленні відношень між об'єктами потребує від учнів застосування не тільки розумової операції аналізу найвищого, структурно-генетичного рівня, а й синтезу.

Побудова об’єкта як деякої абстракції у процесі об’єктно-орієнтованого програмування здійснюється на основі кількох розумових операцій. Виконання розумової дії порівняння використовується для встановлення подібності та відмінності об’єктів і явищ деякої предметної області. Мислительна дія узагальнення передбачає знаходження взаємозв’язків загального і одиничного в множині об’єктів, що дозволяє розглядати об’єкт не ізольовано, а як представника відповідного класу. При виконанні розумової операції абстрагування, здійснюється виділення суттєвих характеристик деякого об’єкта, що відрізняють його від усіх інших видів об’єктів, випускаючи ті характеристики, які для абстракції на даному рівні несуттєві.

Узагальнення, в результаті якого з’являються класи в об’єктно-орієнтовному плані, виконує підготовчу роботу для операції класифікації, в тому числі і систематизації. Класифікація є одним з видів систематизації – мислительної діяльності, у процесі якої об’єкти, що розглядаються організуються в певну систему, наприклад, в ієрархічну діаграму успадкування за певними загальними ознаками.

Застосування технології об’єктно-орієнтованого програмування для створення комп’ютерних моделей при розв’язуванні задач-проектів має ряд переваг порівняно з алгоритмічною як з точки зору програмування, так і з точки зору інтелектуального розвитку учнів:

1. Мислення категоріями об’єктами в інформатиці є найбільш наближеним до реального людського мислення. Все, що оточує людину є об’єктом зі своїми індивідуальними або загальними для деякої групи об’єктів характерними рисами, лінією поведінки, взаємодії з іншими об’єктами тощо. Технологія об’єктно-орієнтованого програмування надає зручні засоби для створення найбільш функціонально повних моделей реальних об’єктів.
2. Побудова об’єктної моделі більше потребує виконання комплексу розумових дій: аналізу, синтезу, абстрагування, порівняння, узагальнення, класифікації тощо, ніж структурне програмування. Систематичне виконання

цих розумових дій сприяє формуванню відповідних інтелектуальних умінь. Структурне програмування також передбачає виконання таких інтелектуальних операцій, але в меншій мірі (не виникає потреби виділяти об’єкти, їх властивості тощо). На думку М.З.Грузмана [70] структурному програмуванню більше притаманний значний вплив на розвиток алгоритмичності мислення.

3. Об’єктно-орієнтована декомпозиція дозволяє створювати гнучкі архітектури систем. Основним будівельним блоком алгоритмічної декомпозиції є процедура або функція, а увага приділяється перш за все питанням передачі управління і розділенню більших алгоритмів на менші. Але такі програмні проекти дуже важко адаптувати до змін вимог або збільшення розмірів. На відміну від алгоритмічної, основу об’єктно-орієнтованої декомпозиції становлять об’єкти або класи. В загальному сенсі об’єкт – це сутність, взята із словника предметної області, а клас є описанням множини однотипних об’єктів. Кожний об’єкт характеризується ідентичністю (найменування), станом (деякі дані, пов’язані з об’єктом) і поведінкою (з об’єктом щось можна робити або він сам може щось робити з іншими об’єктами). Зміна вимог до проекту системи або збільшення її розмірів передбачає або введення нових об’єктів до архітектури системи, або внесення змін до структури окремих незалежних об’єктів, що зробити значно простіше, ніж заново перебудовувати весь проект.

4. При об’єктно-орієнтованому підході існує можливість об’єднати всі майже незалежні представлення системи з різних точок зору в єдине семантичне ціле. При структурному аналізі невідповідність між прийнятою в ньому моделлю системи з реальної предметної області з часом і в результаті перетворень буде збільшуватись, а поведінка створеної програмної системи буде все більше відрізнятись від задуманого.

5. Більшість сучасних мов програмування, інструментальних засобів і операційних систем є в тій чи іншій мірі об’єктно-орієнтованими, що надає

вагоме обгрунтування для доцільності міркування в програмному світі в термінах об’єктів.

6. Об’єктно-орієнтоване програмування забезпечує механізм відокремлення операцій занесення і отримання даних від реалізації конкретного формату об’єкта. Тим самим при використанні сучасних об’єктно-орієнтованих програмних засобів надається можливість створювати великі програмні компоненти, придатні для багаторазового використання, що дозволяє значно економити час учнів при створенні програмних проектів і концентрувати їх зусилля на творчому мисленні.

7. Застосування бібліотек об’єктів багаторазового використання надає можливість користувачу створювати програми на основі опису програми своєю професійною мовою, що скорочує ланцюг традиційної технології розв’язування задач (Користувач $\rightarrow$ Аналітик (прикладний програміст) $\rightarrow$ Програміст-оператор $\rightarrow$ Обчислювальна машина (програма)) до ланцюга (Користувач $\rightarrow$ Обчислювальна машина (програма)). При традиційній технології створення програмних проектів користувач здійснював опис задачі своєю професійною мовою. Аналітик будував математичну модель задачі, описував структури для подання вхідних і вихідних даних, вибирав метод, розробляв алгоритм розв’язування задачі. Програміст на основі розробленого алгоритму складав програму для обчислювальної машини і налагоджував її. Після цього здійснювалось тестування програми. Якщо результати, які вона видавала, не відповідали вимогам користувача, то в саму постановку задачі та математичну модель вносилися зміни і технологічний ланцюжок розв’язування спрацьовував заново. Недоліками такого підходу є великі затрати часу та непрофесійна інтерпретація результатів виконання програми. Саме ТООП дозволяє запобігти виникненню цих недоліків, здійснюючи прямий доступ до обчислювальних машин завдяки створенню об’єктних бібліотек багаторазового використання, в яких назва, функціонування окремого об’єкта відповідає професійній термінології користувача.

Наведені переваги об’єктно-орієнтованого підходу до побудови програмної моделі системи з реальної предметної області свідчать:

- про правомірність включення ТООП в методичну систему формування інтелектуальних умінь учнів у процесі навчання інформатики;
- про необхідність виявлення інваріантів зазначених вище розумових дій учнів, засвоєння яких сприяло б правильному використанню технології об’єктно-орієнтованого програмування при розв’язуванні задач-проектів;
- про доцільність розробки компонентів зазначеної методичної системи для формування умінь виконувати інтелектуальні операції: аналіз, синтез, порівняння, абстрагування, узагальнення, класифікацію;
- про доцільність застосування відповідних об’єктно-орієнтовних інструментальних засобів для практичної реалізації моделі системи з деякої предметної галузі.

Методику формування інтелектуальних умінь учнів при навчанні інформатики ми будуємо на основі наступних принципів:

1. *Від простого – до складного.* Мета формування умінь ставиться при розв’язуванні перших задач, виконанні перших вправ з оволодіння новим прийомом, орієнтовною основою дії уміння, методом розв’язування задач, що показують практичну цінність набуття цього уміння. Тому на початкових етапах формування уміння доцільно використовувати задачі, при розв’язуванні яких учні привчаються оперувати відповідним прийомом в конкретній ситуації. Ці задачі не повинні бути складними, в них повинно чітко виявлятися прийоми, що входять до складу уміння. Поступово до задач можуть вводитися ускладнення, так щоб уміння, що формується, включалося у вже існуючу систему умінь учнів. Перші такі задачі доцільно розв’язувати з докладним обговоренням і записом орієнтовної основи дії уміння. Це допомагає усвідомленому формуванню уміння, усвідомлені уміння формуються швидше і довше зберігаються.

2. *Від конкретного – до абстрактного, зв'язок з реальним життям, міжпредметні зв'язки.* Формування уміння на основі задач із змістом з повсякденного життя дозволяє учню бачити за поняттями, вправами, структурами даних знайомі йому реалії. У фактах і спостереженнях знаходиться вихідний початок сприйняття предмету, після чого здійснюється перехід від окремого і конкретного до загального і абстрактного в предметі. Потім у подальшому житті при використанні сформованих умінь і набутих знань при розв'язанні практичних завдань, завдань з інших шкільних предметів здійснюється зворотний перехід від загального і абстрактного до окремого і конкретного.

3. *Від відомого – до невідомого.* Актуалізація внутрішніх зв'язків предмету сприяє формуванню системного характеру розумової діяльності, що здійснюється через узагальнення попередніх знань і включення їх у зв'язок з новим знанням, тобто зв'язок вищого порядку.

4. *Від спостереження – до роздумів.* Включення учня у процес пізнавального навчання на основі проблемного підходу дозволяє активізувати його інтелектуальну діяльність при зіткненні з проблемами, які необхідно розв'язати, при пошуку нових засобів для вирішення проблеми, можливостей застосування отриманих знань в нових ситуаціях.

5. *Диференціація і індивідуалізація навчання* забезпечують високий ступінь врахування індивідуальних особливостей учнів та їх самостійність роботи. Індивідуалізоване навчання, грунтуючись на виділенні найсуттєвіших особливостей окремого учня, реалізується за адаптованою для кожного учня програмою, протікає самостійно і не залежить від діяльності інших учнів. Диференційоване навчання передбачає виділення найтиповіших рис для групи учнів, реалізується учнями через власну навчально-пізнавальну діяльність в межах спільної навчальної програми, оптимально поєднує як індивідуальну роботу, так і групові, колективні форми роботи, формуючи

таким чином різнопланові способи діяльності, що складають індивідуальний стиль кожного учня, сприяють усвідомленню ними власних надбань.

Наведені принципи методики формування інтелектуальних умінь учнів та її методологічна основа відповідають основним дидактичним принципам навчання: науковості (ТООП відповідає сучасному рівню технології програмування, застосовується метод наукового пізнання – моделювання, на основі виконуваних інтелектуальних операцій та міжпредметних зв'язках в учнів формується уявлення про процес пізнання і його закономірності), усвідомленості, активності і самостійності (усвідомлене засвоєння учнями орієнтованих основ інтелектуальних дій, пізнавальна діяльність), системності і послідовності (актуалізація внутрішніх зв'язків предмету), доступності (формування умінь при вивченні базового матеріалу на основі руху від простого до складного, від відомого до невідомого), індивідуалізації та диференціації навчання (виконання індивідуальних завдань, групова робота над розв'язуванням задач-проектів).

Для реалізації наведених принципів навчання з метою формування інтелектуальних умінь учнів розроблювані компоненти методичної системи передбачають застосування наступних методів навчання: пояснювально-ілюстративного та репродуктивного методів, які забезпечують учнів фондом дійових знань, умінь, що є необхідною умовою для можливостей організації самостійної пізнавальної, продуктивної, творчої діяльності учнів; проблемного викладення матеріалу та методу доцільних задач, що забезпечують мотивацію як для вивчення теоретичного матеріалу, так і для оволодіння відповідними інтелектуальними уміннями; частково-пошукового (евристична бесіда) та дослідницького методів для забезпечення розв'язування пізнавальних задач або у процесі добору засобів для реалізації власних задач-проектів, або у процесі вивчення предметної галузі, моделювання об'єктів якої здійснюється за змістом умови задачі.

Різні методи навчання реалізуються у відповідних формах навчальної роботи. Форма навчальної роботи визначається як “конструкція відтинку процесу навчання, яка характеризується особливими засобами управління, організації та співпраці учнів в навчальній діяльності” [270, с.12]. Форми навчальної діяльності учнів поділяються на колективні (колективно-фронтальні, колективно-групові, робота в парах) та індивідуальні (індивідуально-фронтальні, індивідуально-групові, індивідуальні). У фронтальних, масових методах роботи (урок, лекція, семінар тощо) задіяний цілий клас. Індивідуальна група форм включає домашні та класні завдання, реферати, доповіді, досліди. Колективні (групові) форми – це творчі завдання, практичні роботи, ігри, гуртки тощо.

На початкових етапах формування інтелектуальних умінь учнів в процесі застосування пояснювально-ілюстративного та репродуктивного методів навчання доцільно пояснення нового матеріалу та ознайомлення учнів з орієнтовною основою дії уміння проводити орієнтуючись на слабкого учня. Для забезпечення роботи в повну силу сильніших учнів їм можна надати індивідуальні класні завдання. Індивідуальні домашні та практичні завдання дозволяють усунути недолік фронтальних домашніх та практичних завдань, коли для сильних учнів завдання виявляється занадто легким, а слабкіші учні, впевнившись у даремності своїх зусиль виконати занадто складну для них роботу, втрачають цікавість до навчання.

Виконання завдань проблемного характеру, розв’язування задач-проектів передбачає практичне застосування, удосконалення інтелектуальних умінь, сформованих на простіших задачах. В цьому випадку порівняно з індивідуальним доцільнішою є групова форма роботи. Саме у процесі такої форми роботи виявляється інтелектуальна ініціатива кожного учня і групи в цілому: з’являється предмет для дискусії, зав’язуються спори, є можливість навчитися доводити та спростовувати. Для організації інтелектуальної діяльності учнів групи у процесі розв’язування задач-проектів (задач

проблемного характеру) можна використовувати різні методи: “мозкового штурму”, морфологічного аналізу, “делфі”, “синектики” тощо. Суть методу “мозкового штурму” полягає в тому, що процеси генерування ідей та їх практичний аналіз розділені за часом, при тому ж здійснюються різними людьми. Перша група людей, схильних до генерування ідей, висувають ідеї (при цьому критика не дозволяється). Інша група по закінченні “штурму” виносить міркування про цінність висунутих ідей (це люди, як правило, з високим рівнем сформованості уміння аналізувати, розвинутим критичним мисленням). При застосуванні методу морфологічного аналізу в модельованій системі виділяється кілька характерних морфологічних властивостей (напрямків функціонування). По кожній з властивостей визначають конкретні варіанти розв’язування. Простий перебір варіантів, їх аналіз призводить до вирішення проблеми. В методі “делфі” використовується принцип “делфійських оракулів”, при якому рішення приймаються самостійно, а потім порівнюються і перевага надається тим, які збіглися у різних людей. Метод “синектики” передбачає пошук нової інформації за допомогою різних видів аналогії (гра слів, визначень тощо) [63, с. 141-142].

Групи (або пари) можна створювати як з учнів, що мають однаковий рівень сформованих інтелектуальних умінь та успішності, так і з різним рівнем. Створення груп з однаковим рівнем успішності та інтелектуального розвитку потребує диференціації завдань для різних груп. Зрозуміло, що група слабких учнів не справиться із завданням сильнішої групи. Включення в групу сильних учнів (5-7 осіб) слабкого учня є неприйнятним як для цього учня, так і для сильніших товаришів. В цьому випадку слабкий учень може отримати психологічну травму або виконувати роль “утриманця”, а серед сильніших учнів можуть з’явитися настрої самозаспокоєності. Побудова груп з рівною кількістю слабких, середніх і сильних учнів дозволяє їм здійснювати корисну взаємодію у процесі навчання. “Колективна навчальна діяльність в групі породжує так званий груповий ефект – добавку до можливостей

кожного” [285, с.120]. В малій групі (парі) особливо виявляється різниця в рівнях успішності та інтелектуального розвитку учнів, що входять до її складу. Для організації колективної роботи в такій групі, наприклад, при розв’язуванні задачі-проекту доцільно чітко диференціювати обов’язки, діяльність її членів. Це може бути розподілення учнів в парі за ролями: розробник програмного забезпечення та менеджер з реклами для його продажу. Як перша, так і друга типи діяльності потребують виконання аналізу функціонування як реальної системи, так і її моделі, прояву творчості мислення, наявності уміння прогнозувати, передбачати можливості використання розробленого програмного забезпечення тощо.

Отже, формування умінь, зокрема інтелектуальних, слід починати з простих доцільних задач на основі репродуктивного методу навчання та індивідуального підходу. Застосування відпрацьованого у процесі репродуктивної діяльності уміння доцільно проводити при виконанні задач проблемного характеру (дослідницьких завдань, реалізації задач-проектів) на основі як індивідуальних, так і групових форм організації навчальної діяльності учнів.

2.1.2. Формування інтелектуальних умінь учнів у процесі вивчення основ об’єктно-орієнтованого програмування

Згідно з визначеними на основі концепції поетапного формування розумових дій П.Я.Гальперина в 1.2.2 етапами формування інтелектуальних умінь учнів реалізація процесу їх розумового розвитку повинна починатися з накопичення фонду знань. У зв’язку з тим, що в межах нашого дослідження формування інтелектуальних умінь учнів проводиться на основі ТООП у процесі навчання інформатики, то для переходу до вивчення основ об’єктно-орієнтованого програмування учні повинні володіти фондом знань з основ алгоритмізації і програмування. Звісно, що навчання буде більше сприяти розумовому розвитку учнів, якщо методи формування їх інтелектуальних

умінь і відповідних якостей мислення застосовувати не тільки при вивченні окремих тем з предмету, а протягом всього предметного курсу. Так, в процесі накопичення знань з основ алгоритмізації і програмування з метою інтелектуального розвитку учнів доцільним буде застосування розроблених нами типів задач, описаних в 1.3.3.

У виборі мови програмування для вивчення основ алгоритмізації і програмування слід враховувати як її простоту і доступність для вивчення учнями, так і можливість реалізації на ній сучасних методів програмування, наприклад, об'єктно-орієнтованого програмування.

Реалізація основних принципів ТООП (абстрагування, інкапсуляція, успадкування, поліморфізм) в мовах програмування грунтується на теорії побудови систем у вигляді структурованих абстракцій (Е. Дейкстра), на ідеї приховування інформації (Д.Парнас), на механізмах абстрактних типів даних (Дж. Гуттаг, С. Жиль, Б. Лісков, М. Шоу), на теорії типів і підкласів (Д.Хоар). Вперше поняття класу і об'єкта введено в мові програмування Simula 67. Система Flex та діалекти Smalltalk-72, -74, -76, -80, взявши за основу методи Simula, довели їх до логічного завершення, тобто забезпечили виконання всіх дій на основі класів. В 1970-х роках створено ряд мов, що реалізують ідею абстракції даних: Alphard, CLU, Euclid, Gypsy, Mesa, Modula. Методи, що використовувалися в цих мовах та мовах Simula і Smalltalk, були використані в традиційних мовах високого рівня. Внесення механізмів об'єктно-орієнтовного програмування в С призвело до виникнення мов С++ та Objective C. На базі мови Pascal виникли Object Pascal, Eiffel і Ada, на базі мови LISP- Flavors, LOOPS, CLOS.

Але не всі із зазначених мов є об'єктно-орієнтованими. Л. Карделлі і П. Вегнер [296] вважають, що мова програмування є об'єктно-орієнтованою тоді і тільки тоді, коли виконуються наступні умови:

- якщо в мові підтримуються об’єкти, тобто абстракції даних, що мають інтерфейс у вигляді іменованих операцій і власних даних з обмеженням доступу до них;
- якщо об’єкти належать відповідним типам (класам);
- якщо типи (класи) можуть успадковувати атрибути супертипів (суперкласів);
- якщо реалізується можливість встановлення відношення “is-a” (“це є”) між класом і його суперкласом.

Згідно з цим визначенням об’єктно-орієнтованими мовами є Smalltalk, Object Pascal, C++ і CLOSS, а інші перераховані мови, наприклад Ada, є об’єктними.

Вважається, що найбільш зручною з об’єктно-орієнтованих мов програмування для вивчення в шкільному курсі інформатики є Object Pascal. К. Шмукер стверджує, що “ Object Pascal – це “скелет” об’єктно-орієнтованої мови. В ній немає змінних класу, множинного успадкування і метакласів. Ці механізми виключені спеціально, щоб зробити мову простою для вивчення “об’єктними” програмістами початківцями” [297]. Object Pascal створювався співробітниками компанії Apple Computer (деякі з яких були учасниками проекту Smalltalk) разом з Никлаусом Виртом, творцем мови Pascal. Безпосереднім попередником Object Pascal є Clascal (об’єктно-орієнтована версія Pascal для комп’ютера Lisa). Object Pascal відомий з 1986 року як перша об’єктно-орієнтована мова програмування – засіб розробки програмного забезпечення для комп’ютерів Macintosh фірми Apple. Object Pascal характеризується наявністю змінних і методів екземпляра, методів класу, модульністю, реалізацією одиночного успадкування, агрегації, інкапсуляцією змінних і методів, тобто має ті загальні механізми, що реалізовані в більшості об’єктно-орієнтованих мовах програмування. Відсутність таких механізмів, як “дружні” (friend) функції, шаблони, множинне успадкування, що присутні в мові професійних програмістів C++, не зменшує значення Object Pascal для створення гнучких об’єктних структур.

Отже, мова програмування Object Pascal поєднує в собі простоту для вивчення і розвинуті механізми реалізації об’єктно-орієнтованого програмування, що характерні і для інших об’єктно-орієнтованих мов програмування. Окрім цього, вивчаючи спочатку основи об’єктно-орієнтованого програмування, наприклад, в TURBO Pascal 7.0, а потім в Object Pascal для Delphi учні матимуть змогу пройти шлях поступового розвитку мови, аналізуючи її удосконалення, відкриваючи нові механізми для реалізації програмних моделей. Ці факти доводять доцільність використання цієї мови програмування для вивчення основ алгоритмізації та об’єктно-орієнтованого програмування в шкільному курсі інформатики. У зв’язку з цим пропонуємо наступний план викладання основ об’єктно-орієнтованого програмування (таблиця 2.1).

Таблиця 2.1.

План теми “Основи об’єктно-орієнтованого програмування”.

№	Назва питання	Мова програмування	Кількість годин	
			Лекц.	Практ.
1.	Об’єкт як абстрактний тип даних	TURBO Pascal 6.0 -7.0	1	2
2.	Реалізація успадкування	TURBO Pascal 6.0 -7.0	1	2
3.	Поліморфізм і механізм віртуальних методів	TURBO Pascal 6.0 -7.0	2	2
4.	Обмін даними між об’єктами різних об’єктних типів, зони видимості елементів об’єкта	TURBO Pascal 6.0 -7.0	1	2
5.	Динамічна об’єктна модель	TURBO Pascal 6.0 -7.0 Object Pascal Delphi	2	2
6.	Лінійні списки динамічних об’єктів	Object Pascal Delphi	1	2
7.	Механізм агрегації	Object Pascal Delphi	1	2
8.	Властивості	Object Pascal Delphi	2	2
9.	Методи класів	Object Pascal Delphi	2	2
10	Особливості поліморфізму об’єктної моделі Delphi	Object Pascal Delphi	1	2
	Всього		14	20

Фонд знань з об’єктно-орієнтованого програмування пов’язаний з такими основними поняттями ТООП: “об’єкт”, “клас”, “інкапсуляція”, “поліморфізм”, “успадкування”, “агрегація” тощо. Як зазначалося в 2.1.1, оперування кожним з цих понять передбачає наявність в учнів умінь виконувати розумові дії: аналіз, синтез, абстрагування, порівняння, виділення головного, суттєвого, класифікацію, узагальнення тощо. Про рівень наявних інтелектуальних умінь може свідчити як загальний рівень успішності учня та його досягнення при розв’язуванні задач з різних предметів, зокрема з програмування при навчанні інформатики, так і результати психологічного тестування, що проводиться з метою діагностики сформованості умінь виконувати окремі розумові дії. Обгрунтуванням допустимості застосування психодіагностики для виявлення рівня сформованості інтелектуальних умінь учнів при навчанні об’єктно-орієнтованого програмування виступає той факт, що розумові дії, які виконують учні у процесі об’єктно-орієнтованого програмування в більшості збігаються з виконуваними у певних психологічних тестах. Наприклад, порівняємо наступні завдання психологічних методик [5] та завдання з основ об’єктно-орієнтованого програмування.

Завдання 2.1(фрагмент психологічного тесту “класифікація понять” або “виключення зайвого”). Вам надані п’ять слів, чотири з яких можна об’єднати в одну групу і дати їй назву. Одне слово не належить до цієї групи. Його необхідно знайти і викреслити.

1. Токар, 2. Вчитель, 3. Лікар, 4. Книга, 5. Космонавт.

Завдання 2.2 (з об’єктно-орієнтованого програмування). Побудувати ієрархічну залежність між об’єктами: людина, військовий, солдат, офіцер, цивільний, викладач, студент.

Психологічна методика “виключення зайвого” спрямована на дослідження наявності умінь узагальнювати, абстрагувати, виділяти суттєві ознаки, класифікувати поняття. Так, при роботі над завданням 2.1 учень

повинен зрозуміти, що чотири слова (токар, вчитель, лікар, космонавт) об’єднує те, що це назви професій людини, а книга – це неживий предмет, яким користується людина. Класифікація понять на основі загальних суттєвих ознак дозволяє розбити їх на різні групи. Якщо, до групи входить тільки одне слово, то воно вважається зайвим.

Ці ж самі інтелектуальні дії задіяні і при виконанні завдання 2.2. Тільки результатом виконання завдання повинна бути побудована ієрархічна залежність між поняттями (рис.2.1). Солдат і Офіцер належать до групи Військових (вони служать в армії), а Викладач і Студент – до групи Цивільних. Всі Військові і Цивільні є Людьми. У зв’язку з цим в корені ієрархії повинний знаходитися об’єкт Людина, на другому рівні – об’єкти Військовий і Цивільний, на третьому рівні відповідно – Солдат і Офіцер, Викладач і Студент.

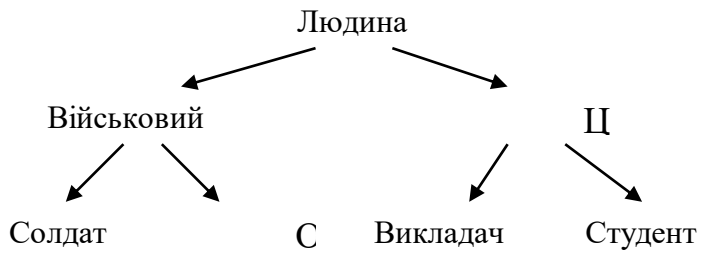


Рис. 2.1. Приклад ієрархічної залежності між об’єктами.

Завдання 2.3 (фрагмент психологічної методики “виділення суттєвих ознак”). В кожному рядку Ви знайдете одне слово, що стоїть перед дужками, і далі – п’ять слів в дужках. Всі слова, що знаходяться в дужках мають якесь відношення до того, яке стоїть перед дужками. Виберіть тільки два і підкресліть їх.

Ділення (клас, ділене, олівець, дільник, папір).

Завдання 2.4 (з об’єктно-орієнтованого програмування). Описати об’єкт “двоопераднa арифметична операція”, який виконує арифметичну операцію ділення двох цілих чисел.

В психологічних тестах типу завдання 2.3 слова підібрані таким чином, щоб учень продемонстрував своє уміння побачити абстрактне значення тих чи

інших понять та відмовитися від легшого, неправильного способу розв’язування, який зразу кидається в очі, і при якому замість суттєвих виділяються окремі, конкретно – ситуаційні ознаки. Так, в наведеному прикладі тестового завдання з арифметичною операцією ділення пов’язуються її суттєві елементи (характеристики) ділене та дільник. Інші слова означають місце та засоби організації процесу виконання цієї арифметичної операції. Ті ж самі характеристики (атрибути) повинні міститися у об’єкті, створення якого передбачається в завданні 2.4.

З метою вимірювання рівня сформованості певних якостей особистості психологи рекомендують застосовувати в одній тестологічній методиці 10-20 завдань. Звісно, що більша кількість завдань в тесті дозволить точніше розділити учнів за їх п’яти рівнями сформованості інтелектуальних умінь (за Л.В.Мар’яненко), наведеними в 1.1.3. Але, як показує досвід експериментальної роботи, на початку процесу формування інтелектуальних умінь учнів з метою диференціації учнів за їх успішністю та інтелектуальним розвитком достатньо використовувати поділ на слабких, середніх (виконано 70-90% тестових завдань) та сильних учнів (виконано більше 90% тестових завдань). Такий підхід дозволить вчителю виконати завдання другого етапу формування уміння (1.2.2): визначити основну масу учнів (належать до групи середніх), учнів, з якими потрібно працювати індивідуально (слабкі), “зону найближчого розвитку” для учнів, що випереджають основну масу (сильні).

Якщо на етапі початкової діагностики (другий етап процесу формування уміння) доцільно використовувати психологічні тести, так як учні ще не володіють знаннями, уміннями і навичками з об’єктно-орієнтованого програмування, то на етапі контролю і корекції (останній сьомий етап) вже є можливість застосування аналогічних вправ з основ об’єктно-орієнтованого програмування. Схожість вправ надає можливість у процесі порівняння результатів точніше оцінити досягнення учня і спланувати його подальше навчання.

На наступному етапі формування уміння виконувати розумову дію – етапі мотивації згідно з діяльнісним підходом ця дія включається в структуру певної діяльності за допомогою доцільних задач. Доцільність теми задачі визначається, або можливістю застосування набутих знань, умінь і навичків у подальшій навчальній та професійній діяльності, або новим матеріалом, вивчення якого дозволить учням розв'язувати ширше коло задач, що доводиться за допомогою поставленої в умові проблеми. В процесі її розв'язування створюється протиріччя між знаннями, засвоєними раніше, і знаннями, необхідними для розв'язування проблеми. Це протиріччя – головна рушійна сила переходу від раніше засвоєних знань до нових, якщо поставлена проблема прийнята кожним учнем і якщо в її розв'язуванні він бере посильну участь. Тому важливо, щоб запропонована задача задовольняла принципам наступності і посильної трудності, щоб вона мала форму і фабулу практичної, а іноді і заохочувальної задачі.

Так, процес вивчення основ об'єктно-орієнтованого програмування доцільно починати з ознайомлення з таким новим структурним типом даних як "об'єкт", створення якого передбачає виконання цілої сукупності інтелектуальних операцій (абстрагування, аналіз, виділення головного тощо). Але це не перший структурний тип даних, з яким зустрічаються учні при вивченні інформатики. Тому зміст умови застосовуваної доцільної задачі повинен бути спрямований на доведенні необхідності застосування саме структурного типу "об'єкт", а не, наприклад, "запис". Доцільність використання нового поняття обумовлює необхідність оволодіння відповідними інтелектуальними уміннями для його застосування.

Задача 2.1. Написати програму заповнення людьми електронної анкети для влаштування на роботу. При цьому передбачити визначення особистісних якостей людини згідно з їх знаком зодіаку.

Ключовим елементом цієї задачі є такий знайомий учням об'єкт, як Людина. Чи можна описати Людину змінною простого типу, як, наприклад,

число цілого типу? Звісно, що ні. Як об’єкт суспільства Людина має багато характеристик (прізвище, ім’я, дата народження, професія тощо). Тоді для описання Людини учні можуть запропонувати вже відомий їм структурний тип даних такий, як запис. Але, наприклад, підбираючи собі колектив робітників, створюючи сприятливу робочу обстановку, підприємець хоче враховувати деякі психічні якості кандидатів (окремих Людей), що на його переконання залежать від знаку зодіаку, під яким народилася людина. Так трапилось, що у зв’язку з новими астрологічними теоріями (дати знаків зодіаку зміщені на 15 днів вперед) не всі Люди, що прийшли влаштовуватися на роботу, знають під яким знаком зодіаку вони народилися. Бажано, щоб до комп’ютерної анкети, яку заповнює Людина, був би приєднаний спеціальний метод – процедура, яка за введеними даними визначає знак зодіаку Людини. Таким чином, абстрактний тип запис вже не буде задовольняти програміста, який створює електронну анкету. Він буде більше задоволений абстрактним типом даних – об’єктом, який поєднує як атрибути (характеристики об’єкта), так і його методи (дії).

При поясненні понять “успадкування”, “поліморфізм”, застосування механізмів яких здійснюється на основі виконання інтелектуальних операцій аналізу, класифікації, порівняння, узагальнення тощо, зміст умови доцільної задачі можна обгрунтовувати доречністю раціонального оформлення програмного коду розв’язування задачі.

Задача 2.2. Описати об’єкти Студент і Викладач. Використати ці об’єкти для обчислення суми доходу конкретного студента і конкретного викладача.

Представники об’єктів Студент і Викладач є людьми, тобто вони мають, як і всі люди ім’я, прізвище, дату народження тощо. Тому при визначенні об’єктів Студент і Викладач ці властивості можна два рази не перевизначати, а успадкувати від об’єкта Людина, в якому їх визначити тільки один раз. В оголошенні ж об’єкта-нащадка вказуються тільки ті властивості, які характерні для екземплярів цього об’єкта. Наприклад, студент

характеризується успішністю, яка задається середнім балом отриманих в останню сесію оцінок, тобто об’єкт студент має додатково до Людини атрибути середній бал, розмір стипендії та метод нарахування стипендії в залежності від середнього балу. Викладач додатково до Людини характеризується назвою курсу лекцій, який він читає, кількістю годин, заробітною платою та має метод нарахування погодинної заробітної плати за кількістю прочитаних годин.

Доцільність розгляду динамічної об’єктної моделі обгрунтовується наданням можливості працювати з сукупністю об’єктів, оформленою, наприклад, у вигляді лінійного списку, на основі якого можна проводити формування інформаційно-пошукових умінь: добавити до списку новий об’єкт, поставити його на відповідне місце, вилучити зі списку і із пам’яті об’єкт, що має певні значення атрибутів, знайти необхідний об’єкт тощо.

Як вже зазначалося в 1.2.2 уміння виконувати певну дію є стійкішим при зміні умов, при перенесенні на нові завдання, якщо учню для засвоєння надається готова орієнтовна основа дії, яка містить не тільки зразки дії, але й усі вказівки щодо правильного її виконання з новим матеріалом. Тому на четвертому етапі формування інтелектуального уміння – рефлексії ознайомлення учнів з прийомами уміння для кращого їх усвідомлення доцільно проводити на основі відповідних правил-орієнтирів. У зв’язку з тим, що більшість інтелектуальних дій, які виконуються учнями у процесі об’єктно-орієнтованого програмування, збігаються із загальними розумовими діями, то за основу правил їх виконання доцільно взяти правила-орієнтири загальних розумових дій, висвітлених в 1.2.3. Додавання до правил-орієнтирів загальних розумових дій елементів, характерних для об’єктно-орієнтованого програмування, дозволило створити їх інваріанти.

Правило-орієнтир створення об’єктів системи як абстрактних типів даних – інваріант розумових дій аналізу-синтезу, абстрагування:

1) пригадати поняття об’єкта та його визначення на понятійному рівні;

2) виділити можливі об’єкти з предметної галузі (фабули задачі);
3) виділити ознаки, що характеризують окремий об’єкт в цілому;
4) визначити поведінку об’єкта, його функції та зв’язок з іншими об’єктами системи;
5) із загальних характеристик об’єкта виділити суттєві з точки зору його функціонування;
6) на основі суттєвих характеристик і принципів функціонування визначити структуру об’єкта: його атрибути, методи (для TURBO Pascal) і властивості (додатково для Object Pascal під Delphi) та позначити їх окремими ідентифікаторами (абстрагуватись від конкретних їх значень);
7) визначити домени – можливі значення, що можуть набувати атрибути об’єкта, а на їх основі – тип атрибутів;
8) перенести проведений аналіз-синтез на всі об’єкти задачі;
9) скласти абстрактні схеми, малюнки з виділеними об’єктами з метою дослідження їх взаємозв’язку;
10) реалізувати створення і функціонування системи об’єктів в програмі засобами конкретної об’єктно-орієнтованої мови програмування.

Таким чином, логіка розумових дій аналізу-синтезу (виділення важливих рис, суттєвих і менш суттєвих властивостей, встановлення зв’язків між ними) доповнюється об’єктно-орієнтованими рисами: виділення об’єктів, визначення властивостей, методів, атрибутів, встановлення відношень між ними на основі зв’язків між об’єктами системи.

Правило-орієнтир інваріанта розумової дії порівняння:

1) виділити характерні риси та поведінку кожного з об’єктів;
2) встановити подібності в рисах і поведінці об’єктів;
3) встановити відмінності в рисах і поведінці об’єктів;
4) знайти правильне підгрунтя (суттєві атрибути) для порівняння об’єктів;
5) використати для порівняння об’єктів більш узагальнені знання (множина, клас);

6) скласти таблицю порівняння об’єктів і дій за атрибутами об’єктів, в якій кожний рядок – це дія виконувана певним об’єктом, а стовпчики – атрибути об’єктів. Певний стовпчик напроти відповідної дії об’єктів помічається, наприклад, символом “+”, якщо цей атрибут використовується при виконанні дії.

Розумова дія порівняння є складовою для узагальнення, правило-орієнтир виконання якої наступне:

1) визначити, до яких категорій відносяться об’єкти: людина, тварина тощо;
2) на основі виділених рис і поведінки визначити атрибути об’єктів;
3) створити і заповнити таблиці, в яких визначити подібності і відмінності об’єктів;
4) на основі знайдених подібних рис різних об’єктів визначити назви множин об’єктів (класів);
5) об’єднати виділені об’єкти у відповідні множини (класи).

Використання узагальнення виконує підготовчу роботу для операції класифікації. За результатом узагальнення об’єктів в об’єктно-орієнтованому програмуванні будуються класи. Слід зазначити, якщо під об’єктом розуміють деяку абстракцію реально існуючого предмета, процесу або явища, то клас – це описання множини об’єктів, що мають загальні атрибути, операції, відношення і семантику. В TURBO Pascal 7.0 програміст в тілі програми працює з екземплярами деяких об’єктних типів, в Object Pascal – з об’єктами, які є екземплярами відповідних класів. Тобто поняття об’єктний тип і клас є тотожними.

Правило-орієнтир для інваріанта дії класифікації, яка тісно переплітається з такими поняттями об’єктно-орієнтованого програмування як успадкування і поліморфізм:

1) об’єднати об’єкти в окремі групи на основі їх загальних ознак;
2) знайти ознаку, яку необхідно покласти в основу класифікації;

3) в кожній групі об’єктів визначити загальні риси, які віднести до їх об’єкта-пращура;
4) виконати аналогічну операцію над виділеними об’єктами-пращурами і т.д.;
5) побудувати згорнуті діаграми успадкування та класифікаційні схеми, на яких схематично зобразити групи об’єктів;
6) побудувати розгорнуті діаграми успадкування та класифікаційні схеми, в яких позначити атрибути, методи об’єктів, зони їх видимості.

На наступному (п’ятому) етапі формування інтелектуального уміння надана учням орієнтовна основа дії відпрацьовується на основі тренувальних задач. Задача вважається тренувальною, якщо система розумових і практичних дій з її розв’язування відома учню. У зв’язку зі спорідненістю загальних інтелектуальних умінь, що формуються, та умінь з об’єктно-орієнтованого програмування, як тренувальні можна використовувати задачі з основ об’єктно-орієнтованого програмування.

Розподіл учнів на різні групи (слабкі, середні, сильні), що був зроблений на етапі попередньої діагностики (другий етап), висуває вимогу диференціації тренувальних задач. Як зазначалося в 1.3.1, ступінь виявлення інтелектуальних умінь при розв’язуванні учнями задачі залежить від згорнутості її умови. Слабким учням доцільно давати тренувальні задачі, які містять докладне описання об’єктів з перерахуванням всіх необхідних атрибутів та методів. Середні учні зможуть розв’язати задачу, умова якої містить частину докладних характеристик об’єктів. Відповідно, сильним учням надається задача зі згорнутим змістом умови. Одну й ту ж саму задачу можна представити в різних видах в залежності від ступеня згорнутості умови.

Задача 2.3 (для слабких учнів). Об’єкт Точка характеризується координатами *x, y* та кольором, має метод побудови свого зображення. Атрибути Точки та методи встановлення нових значень координат успадковуються її об’єктами-нащадками Кругом і Квадратом. Круг додатково

має атрибут – радіус, а Квадрат – довжину сторони. В кожному з цих об’єктів перевизначається метод малювання їх зображення. Реалізація переміщення фігури по екрану в графічному режимі передбачає малювання фігури в попередній позиції, а потім кольором малюнку – в новій позиції. Написати програму переміщення цих фігур по екрану за деякою траєкторією.

Задача 2.4 (для середніх учнів). Реалізація переміщення фігури по екрану в графічному режимі передбачає малювання фігури в попередній позиції, а потім кольором малюнку – в новій позиції. Геометричні фігури круг, квадрат і точка знаходяться в ієрархічній залежності (Круг і Квадрат є нащадками Точки) і мають власні методи малювання. Написати програму переміщення цих фігур по екрану за деякою траєкторією.

Задача 2.5 (для сильних учнів). Описати об’єкти: круг, квадрат, точка. Використати ці об’єкти для переміщення відповідних фігур по екрану за деякою траєкторією.

Так, в задачі 2.3 надається повне описання об’єктів, їх ієрархічної залежності, перераховуються всі необхідні атрибути об’єктів та методи, надається рекомендація щодо розв’язування. Умова задачі 2.4 побудована так, що учень отримує тільки частину інформації про об’єкти системи, про деякі атрибути і методи (про метод малювання, атрибути позиція і колір), про їх ієрархічну залежність та рекомендацію стосовно розв’язування задачі. В умові задачі 2.5 вказуються тільки задані об’єкти та їх поведінка (об’єкти переміщуються). У випадку розв’язування останньої задачі учень повинен самостійно виконати комплекс розумових дій для побудови ієрархічної структури об’єктів та програмної реалізації їх поведінки.

На цьому ж тренувальному етапі учням можна запропонувати творчі завдання, при виконанні яких вони могли б застосувати відпрацьовані вміння. Наприклад, придумати нове застосування для вже відомих і використаних в попередніх задачах об’єктів. Так, об’єкт точка можна застосувати для імітації на екрані комп’ютера зоряного неба, при малюванні графіка деякої функції.

Виконання подібних задач корисно також для формування інтелектуального уміння прогнозувати, передбачати поведінку об’єкта в нових умовах. До складу цього уміння входять операції:

1) аналіз умов нової ситуації для застосування об’єкта;
2) аналіз структури і можливостей об’єкта стосовно задоволення потреб нової ситуації;
3) застосування об’єкта, його пристосування до нових умов.

Підгрунтям для успішного виконання учнями таких завдань може бути підхід вчителя при викладанні нового матеріалу, коли він використовує в пояснювальних прикладах об’єкти, що раніше вже розглядалися, навчаючи учнів застосовувати один і той же об’єкт в різних ситуаціях, формуючи тим самим уміння прогнозувати, розвиваючи неординарність, гнучкість їх мислення.

На наступному (шостому) етапі узагальнення засвоєні прийоми уміння переносяться на інші теми і предмети (ближнє перенесення) та на інші сфери діяльності (далеке перенесення). Значні можливості, порівняно з іншими шкільними предметами, для застосування загальних інтелектуальних умінь у процесі об’єктно-орієнтованого програмування надає математика, курс чисельних методів [222], елементи якого вивчаються як в шкільній математиці, так і інформатиці. Наприклад:

Задача 2.6. (для Object Pascal) Клас наближеного обчислення інтегралів як методи містить реалізацію двох чисельних методів обчислення інтегралів: ($S=\left(hf\left(a+\frac{h}{2}\right)+f\left(a+\frac{h}{2}+h\right)+\ldots+f\left(a+\frac{h}{2}+(n-1)h\right)\right), h=\frac{a-b}{n}$) - метод середніх прямокутників та ($S=h(f(a+h)+f(a+2h)+\ldots+f(a+nh)), h=\frac{a-b}{n}$) - метод правих прямокутників, при реалізації яких функція *f(x)* передається їм як параметр процедурного типу. Написати програмний код реалізації та використання цих класів для наближеного обчислення інтеграла деякої функції, вивести і порівняти результати обчислень.

Прикладом застосування набутих інтелектуальних умінь у сфері економічної діяльності можуть бути різні задачі нарахування заробітної плати, стягнення податків, штрафів, виконання банківських операцій, визначення економічних характеристик підприємства тощо. Наприклад:

Задача 2.7. Описати об’єкт “вчитель”, заробітна плата якого залежить від педагогічного стажу. В цьому об’єкті передбачити метод нарахування заробітної плати за правилом: якщо стаж більший 3-х років, то заробітна плата збільшується на 5%, якщо стаж більший 10 років, то – на 10%, якщо стаж більший 20 років, то на - 15%. Використати цей об’єкт в програмі для нарахування заробітної плати конкретному вчителю і розміру податку з неї (10%).

Перенесення, застосування уміння в межах того самого предмету, на основі якого воно формується, дозволяє розглядати окремий навчальний матеріал теми не ізольовано, а в тісному взаємозв’язку з іншими темами предмету, що сприяє як формуванню цілісного уявлення про предмет, так і набуттю стійких знань і умінь. Опора на пройдений матеріал, наступність в навчанні надає можливість для подальшого розвитку наявних в учнів знань, умінь і навиків.

Логічним продовженням вивчення основ об’єктно-орієнтованого програмування, яке використовується і як засіб моделювання знань предметної галузі, є розгляд моделей подання знань в межах теми “Бази знань інтелектуальних систем”, зокрема, фреймів і семантичних мереж. Порівняння, проведення аналогій між згаданими моделями подання знань дозволить учням побачити спільні риси між цими поняттями і зрозуміти можливості застосування набутих умінь на новому матеріалі, в нових умовах.

Сам термін “фрейм” походить від англійського слова “Frame”, що означає “каркас” або “рамка”, і використовується для позначення структури знань (абстрактного образу) для подання деякого стереотипу сприйняття ситуацій, сцен з реального життя. Об’єкт, як структурний тип даних,

використовується для подання в абстрактному структурованому вигляді інформації про об’єкт, явище оточуючого середовища.

Розрізняють фрейми-зразки, або прототипи, що зберігаються в базі знань, і фрейми-екземпляри, які створюються для відображення реальних фактичних ситуацій на основі отриманих даних. В блоці описання типів Type програми, створеній на основі ТООП, задається об’єктний тип або клас (об’єкт-каркас, зразок), а в її тілі використовуються об’єкти-екземпляри об’єктного типу (класу), атрибути яких у процесі виконання програми набувають конкретних значень.

Для відображення різноманітності знань про оточуючий світ використовують (В.Хорошевський, [261]): 1) фрейми-структури, що застосовуються для позначення об’єктів і понять (поїзд, вагон, будинок, залог, вексель); 2) фрейми-ролі (начальник вокзалу, диспетчер, вчитель, учень, касир, клієнт); 3) фрейми-сценарії (банкрутство, збори акціонерів); 4) фрейми-ситуації (аварія, робочий режим пристрою).

Для ідентифікації об’єктів шляхом розгляду концептуальних сутностей або “предметів”, пов’язаних з проблемою, що аналізується, використовуються наступні категорії об’єктів (С.Шлеер, С.Меллор [278]): 1) реальні об’єкти – абстракції фактичного існування деяких предметів у фізичному світі (поїзд, вагон, будинок, залог, вексель); 2) ролі – абстракції мети або професійного призначення людини, частини обладнання або організації (начальник вокзалу, диспетчер, вчитель, учень, касир, клієнт); 3) інциденти – абстракція події, що трапилась (прибуття поставки, банкрутство, аварія); 4) взаємодія – об’єкти, що отримуються в результаті відношень між іншими об’єктами (перехрестя – місце перетину двох вулиць).

Традиційно структура фрейму представляється як список його властивостей (слотів):

Ім’я фрейма:
Ім’я 1-го слота: значення 1-го слота,
Ім’я 2-го слота: значення 2-го слота,
. . . . . .
.

Загальна структура об’єктної моделі для Object Pascal містить список атрибутів (характеристик об’єкта), його методів (процедур або функцій, що реалізують його поведінку), а в Delphi введені додаткові елементи такі як методи класів (для удосконалення реалізації механізму об’єктів різних класів) та властивості (для спрощення реалізації механізму доступу до атрибутів). Так, для TURBO Pascal 7.0 структура об’єктної моделі матиме вигляд:

Ім’я = Object (ім’я пращура)
Ім’я 1-го атрибута: тип значень 1-го атрибута;
. . . . . .
Ім’я К-го атрибута: тип значень К-го атрибута;
Ім’я 1-го методу;
. . . . . .
Ім’я М-го методу;
End;

Існує кілька способів отримання слотом значення у фреймі-екземплярі: 1) за замовчанням від фрейму-зразка (Default - значення); 2) через успадкування властивостей, вказаних у слоті АКО; слот АКО (зв’язок *A-Kind-Of = це є*) вказує на фрейм вищого рівня ієрархії, звідки неявно успадковуються, тобто переносяться, значення аналогічних слотів; 3) за формулою, вказаною в слоті; 4) через приєднану процедуру; 5) явно із діалогу з користувачем; 6) із бази даних.

В наведеній (рис.2.2) мережі фреймів фрейм “Учень” успадковує властивості фреймів “Дитина” і “Людина”, що знаходяться на вищому рівні ієрархії. Так, на питання “чи люблять їсти учні солодке” буде дана відповідь “так”, тому що цю властивість мають всі діти, як вказано у фреймі “Дитина”. Успадкування властивостей може бути частковим. Наприклад, вік учнів не успадковується із фрейму “Дитина”, оскільки явно вказаний у своєму власному фреймі.

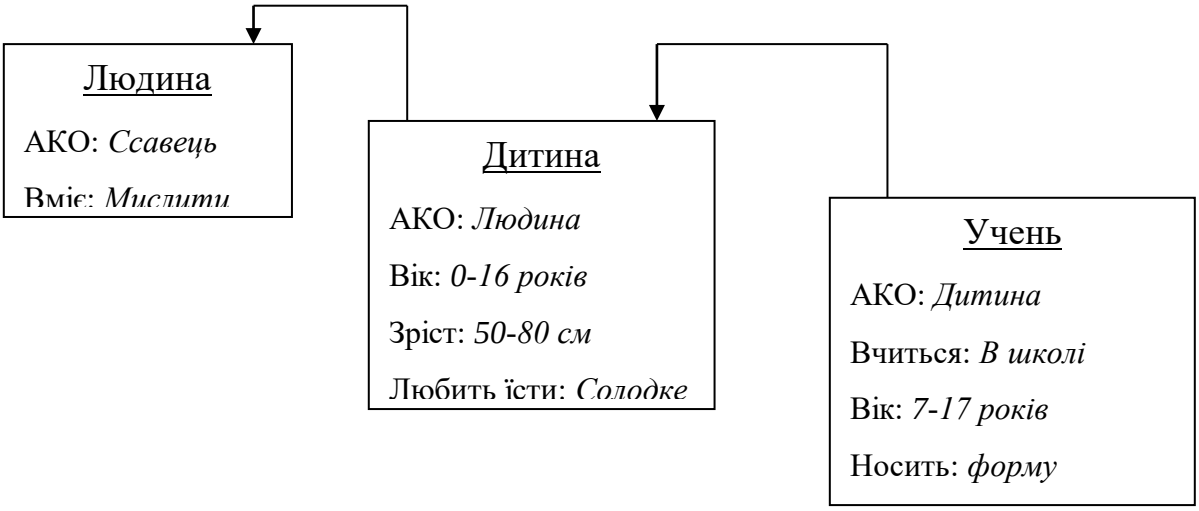


Рис.2.2. Ієрархічна мережа фреймів.

Для отримання значень окремими атрибутами об’єкта-екземпляра можна також навести аналогічні способи: 1) атрибут набуває значення за замовченням (Object Pascal Delphi) через відповідну властивість, якщо в оголошенні цієї властивості після службового слова Default стоїть якесь значення; 2) об’єкт-нащадок успадковує від об’єкта пращура його атрибути разом з їх значеннями, а також може змінити значення атрибутів пращура як своїх власних; 3) атрибут набуває значення за допомогою власних методів, або через втручання інших об’єктів, яке здійснюється або через методи, або через властивості (для Object Pascal Delphi) цього ж самого об’єкта; 4) через методи класів інших об’єктів (для Object Pascal Delphi); 5) із діалогу з користувачем.

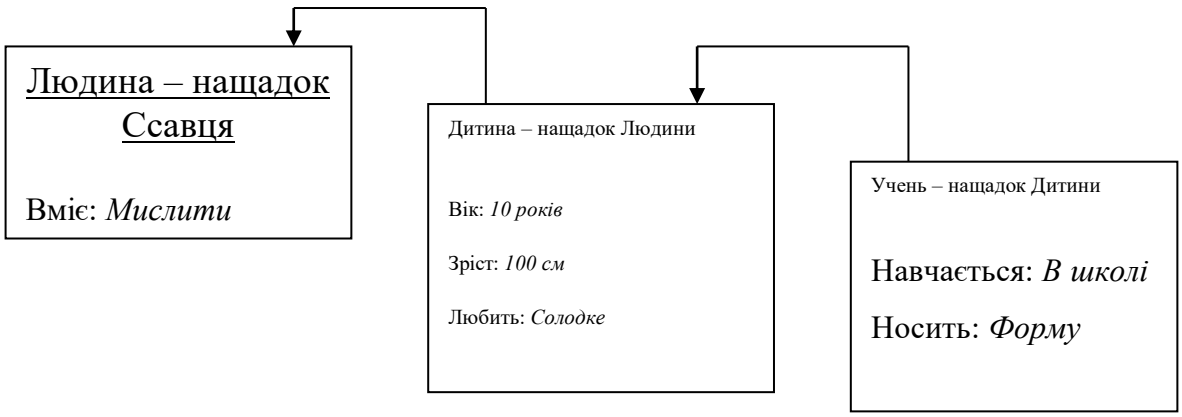


Рис.2.3. Ієрархічна залежність між об’єктами.

Розглянемо, як буде виглядати реалізація залежності понять “Людина”, “Дитина”, “Учень” в об’єктній ієрархічній структурі (рис.2.3). В наведеній ієрархії об’єктів об’єкт “Учень” успадковує всі атрибути з їх значеннями від об’єктів “Дитина” і “Людина”. Так, для реалізації відповіді на питання “чи люблять їсти учні солодке” необхідно зчитати значення з атрибута “Любить”, успадкованого від об’єкта “Дитина”. Якщо зчитане значення буде збігатися зі словом “Солодке”, то надати відповідь “так”. Щодо перевизначення атрибута “Вік”, то слід зауважити, що об’єкти, які знаходяться в ієрархічній залежності не повинні мати атрибутів з однаковими іменами.

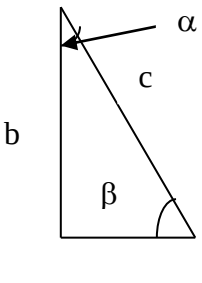
Проведений аналіз правил побудови фреймової моделі знань та об’єктної моделі представлення даних доводить доцільність застосування об’єктно-орієнтованого програмування для реалізації баз знань інтелектуальних систем. Якщо під даними розуміти окремі факти, що характеризують об’єкти, процеси і явища предметної області, а під знаннями– структуровані данні, або данні про данні, то через їх структурність як фреймову, так і об’єктну модель можна вважати моделями подання знань.

Не менш зручним є застосування об’єктної моделі для реалізації семантичних мереж в системах прийняття рішень. Семантика - це наука, що встановлює відношення між символами, які позначають реальні об’єкти. У зв’язку з цим під семантичною мережею розуміють орієнтований граф, вершини якого – поняття (об’єкти), а дуги – відношення між ними (зв’язок “*A-Kind-Of = це є* “, “is”, “частина - ціле”, “належить” тощо). Проблема пошуку розв’язку в базі знань типу семантичної мережі зводиться до задачі пошуку фрагменту мережі, що відповідає поставленому запиту до бази знань.

Розглянемо приклад застосування об’єктно-орієнтованого програмування для побудови інтелектуальної системи на основі семантичної мережі з метою розв’язування шкільної задачі з курсу геометрії.

Задача 2.8. За двома катетами прямокутного трикутника знайти радіус вписаного в цей трикутник кола.

Для отримання розв’язку задачі готової формули в підручнику не існує, вона виводиться з інших формул. Ті знання (формули), що відомі про прямокутний трикутник, опишемо у вигляді окремих об’єктів:



Об’єкт S1: $\cos\alpha=\frac{b}{c}$; (1)

Об’єкт S2: $a=b\cdot tg\alpha$; (2)

Об’єкт S3: $a^2+b^2=c^2$; (3)

Об’єкт S4: $\alpha+\beta=90^0$; (4)

Об’єкт S5: $R=\frac{c}{2}$; (5)

Об’єкт S6: $p=\frac{a+b+c}{2}$; (6)

Об’єкт S7: $S=\frac{a\cdot b}{2}$; (7)

Об’єкт S8: $S=p\cdot RR$; (8)

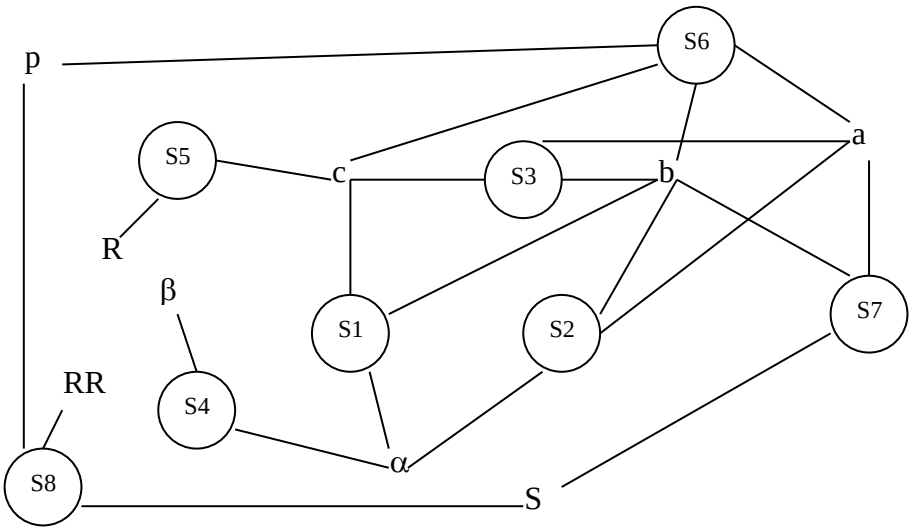
Рис. 2.4. Зображення прямокутного трикутника та пов’язаних з ним формул.

У формулах використовуються позначення: *a, b* – катети трикутника, *c* – гіпотенуза, α, β - кути прямокутника, *p* – півпериметр, *S* – площа трикутника, *R* – радіус описаного кола, *RR* – радіус вписаного кола.

Людина, що розв’язує задачу 2.8, проглядаючи формули (1)-(8), може здогадатися про послідовність дій процесу розв’язування: з формули (3) знайти значення гіпотенузи трикутника, за формулою (6) обчислити півпериметр, а за формулою (7) – площу трикутника, потім із формули (8) знайти радіус вписаного кола. В інтелектуальних системах про це повинен здогадатися сам комп’ютер. Підгрунтям для реалізації інтелектуальної системи розв’язування задачі 2.8 є семантична мережа, яка відображає зв’язок формул і змінних (рис.2.5).

Розглянемо, як буде здійснюватися пошук розв’язку задачі 2.8 – значення вписаного в прямокутний трикутник кола (*RR*) з точки зору реалізації наведеної (рис.2.5) семантичної мережі. Згідно з умовою задачі заданими є значення катетів *a* і *b* трикутника, тобто їх значення можна присвоїти атрибутам, наприклад, об’єкта S2. За лініями зв’язків між об’єктами значення a до відповідних атрибутів об’єктів: S3, S6, S7, а значення *b* запишеться до атрибутів об’єктів: S1, S3, S6, S7. В об’єкті S3 є всі необхідні данні для

обчислення значення *c*. Обчислене значення *c* записується до атрибутів об’єктів: S1, S5, S6. В об’єкті S7 є всі необхідні дані для обчислення площі трикутника – атрибута *S*, яке потім записується до атрибуту об’єкта S8. В об’єкті S6 є всі необхідні дані для обчислення значення півпериметра *p*, яке теж записується до параметра об’єкта S8. На останньому кроці обчислень в об’єкті S8 при наявності всіх необхідних даних знаходиться радіус вписаного в трикутник кола (*RR*). Приклад програмної реалізації цієї інтелектуальної системи розв’язування задач, пов’язаних з прямокутним трикутником і



вписаним, описаним колами, наводиться в додатку А.

Рис. 2.5. Приклад представлення знань для задачі 2.8 у вигляді семантичної мережі.

В шкільному курсі інформатики вивчення основ штучного інтелекту закінчується вивченням експертних систем, наприклад, систем діагностичного характеру, однією з яких є оболонка для створення діагностичних експертних систем DESS (Diagnostics Expert Systems Shell). Створення діагностичної експертної системи для деякої предметної галузі передбачає виконання її аналізу, у процесі чого виділяються окремі етапи діагностики. На кожному етапі задається чергове уточнююче питання та за можливими відповідями на нього формується логіка переходів від одного етапу до іншого. Етап діагностики, що здійснюється системою, можна розглядати як певний об’єкт,

який характеризується назвою, певним поясненням, питанням, відповідь на яке спричиняє перехід до іншого етапу. Відповідь на питання можна розглядати як повідомлення, яке один об’єкт посилає іншому. Загальні ознаки предметів або явищ предметної галузі, що розглядається, відносяться до початкових етапів діагностики (об’єктів вищих класів), а за відмінними ознаками формуються інші, кінцеві етапи діагностики. Подібність виконуваних дій при об’єктно-орієнтованому програмуванні та створенні експертних систем обгрунтовує доцільність застосування однакових прийомів для їх здійснення, а тим самим для перенесення в нові умови, на інші предмети (математику, ботаніку, зоологію, географію тощо). Рекомендації щодо застосування оболонки DESS у навчальному процесі та приклади завдань для розв’язування в її середовищі за тематикою різних навчальних предметів наведені в додатку К.

Як показують наведені приклади, між ТООП і технологією створення інтелектуальних систем існує тісний взаємозв’язок. Тому правомірним є припущення, якщо існує можливість реалізації баз знань в інтелектуальних системах засобами об’єктно-орієнтованого програмування, то доцільним є використання методів набуття знань, що застосовуються в теорії штучного інтелекту, для побудови об’єктних моделей. Серед методів набуття знань визначають наступні: 1) комунікаційні методи: а) пасивні (спостереження, протокол “думки вголос”, лекції); б) активні: групові (“мозковий штурм”, круглий стіл, ролева гра) та індивідуальні (анкетування, інтерв’ю, діалог, експертна гра); 2) текстологічні методи (аналіз підручників, аналіз літератури, аналіз документів). Не тільки активні, але й пасивні методи вимагають від розробника моделі знань (об’єктної моделі) уміння чітко аналізувати потік інформації предметної галузі з метою виділення в ньому суттєвих фрагментів знань. Тому застосування цих методів при розв’язуванні творчих задач (задач-проектів) на основі об’єктно-орієнтованого програмування можна вважати одним з засобів формування інтелектуальних умінь учнів.

Як бачимо, назви методів набуття знань в більшості збігаються з педагогічними методами організації навчальної діяльності учнів, у процесі якої вони набувають нових знань, умінь та навичок. Отже, навчання учнів методів набуття знань на уроках інформатики, формування загальних інтелектуальних умінь сприяють покращанню загальної успішності учнів за рахунок застосування при вивченні інших предметів умінь застосовувати зазначені методи.

Перенесення набутих умінь на інші предмети та сфери діяльності, повторювальність дій дозволяють зміцнювати асоціації, між завданням, знаннями, необхідними для його виконання, і застосуванням знань на практиці. Варіації завдань дозволяють зробити ці асоціації точнішими. У такий спосіб формуються риси й ознаки умінь: гнучкість (спроможність раціонально діяти в різноманітних ситуаціях), стійкість (зберігання точності і темпу, незважаючи на побічні впливи), міцність (уміння не втрачається в той період, коли воно практично не застосовується).

2.2. Метод проектів як засіб формування інтелектуальних умінь учнів

2.2.1. Методологічні основи застосування методу проектів на уроках інформатики з метою формування інтелектуальних умінь учнів

Згідно з дослідженнями І.Я.Лернера [137, с.46] засвоєні знання про спосіб діяльності, сформоване у процесі репродуктивної діяльності уміння виконувати цю діяльність ще не забезпечують його бездоганного виконання на необхідному рівні. Вправність, швидкість реакції, адаптованість до конкретних умов дії забезпечується досвідом здійснення цієї дії. Оволодіння досвідом інтелектуальної, творчої діяльності здійснюється, за І.Я.Лернером [137, с.101] у процесі розв'язування учнями спеціально розробленої системи проблем і проблемних задач, що становлять основу проблемного навчання. Для окремого учня проблема виникає з проблемної ситуації (утруднення, подолання якого потребує нових знань, способів і дій) при усвідомленні останньої, при її прийнятті до розв'язування на основі наявного фонду знань і умінь, який набувається у попередній репродуктивній діяльності.

Проблемною є задача, самостійне розв'язування якої спрямоване, виходячи з відомого, на отримання нових знань, на створення нових засобів пошуків знань або досягнення мети, на розширення галузі застосування відомого знання. Змістом такої задачі є проблема, в основі якої лежить протиріччя між відомим і шуканим, що знаходиться через проміжні операції [137, с.80-81]. Роль проблемної задачі у формуванні інтелектуальних умінь обумовлюється її структурністю і потребує умінь виконувати розумові дії: "аналіз умови задачі і її співвіднесення з питанням задачі; перетворювання основної проблеми в ряд окремих проблем, що підпорядковані головній; проектування плану і етапів вирішення проблеми; формулювання гіпотези; синтез різних напрямків пошуку; перевірка розв'язку тощо"[137, с.105]. Таким чином, виконуючи проміжні операції, учні набувають досвіду здійснення інтелектуальної діяльності.

Використовувані в нашому дослідженні задачі-проекти за своїми характеристиками повністю відповідають проблемним задачам і можуть бути використані для набуття учнями досвіду виконання розумових операцій, в ході чого вдосконалюються, поглиблюються їх загальні інтелектуальні уміння. В загальному розумінні проектом вважається сукупність певних дій, документів, попередніх текстів, задум для створення реального об'єкта тощо. Метод проектів завжди передбачає розв'язування деякої проблеми, що потребує застосування різних методів, інтегрування знань з різних галузей науки, техніки, технології тощо. Робота над задачею-проектом, як і над будь-якою іншою проблемною задачею передбачає усвідомлення певної проблеми та процес її вирішення, який включає в себе планування дій, наявність задуму або гіпотези вирішення цієї проблеми, у випадку групової роботи – чіткий розподіл ролей, тобто завдань для окремого учня групи при умові їх тісної взаємодії, аналіз результатів. Форми організації роботи учнів над проектом можуть бути різними: індивідуальні проекти, парні проекти, групові проекти.

Метод проектів, запропонований у 1919 році в місті Дальтон (США) американським педагогом Е. Паркхарстом з метою індивідуалізації процесу навчання, використовувався в Радянському Союзі в 20-30 ті роки XX століття. Бурхливий розвиток інформаційних технологій, як засобу реалізації проектів, сприяє відродженню цього методу навчання в теперішній час. Все більше українських педагогів-практиків використовують цей метод в своїй викладацькій діяльності, зокрема, Т.П.Караванова [109], М.Е. Єгорова [77] та інші. В інформатиці метод проектів в основному використовується в двох напрямках: телекомунікаційні проекти та проекти створення комп'ютерних моделей реальних систем.

Телекомунікаційні проекти передбачають навчально-пізнавальну, творчу або ігрову діяльність учнів-партнерів, організовану на основі комп'ютерної телекомунікації, що має єдину мету, узгоджені методи, способи діяльності, спрямовані на досягнення єдиного результату діяльності. Беручи

участь у телекомунікаційному проекті, учні різних шкіл, міст, країн засобами мережі Internet на основі навчання в співробітництві проводять дослідження з певної проблеми. Робота над телекомунікаційним проектом проводиться в кілька етапів: організаційний (представлення проекту, пошук партнерів), вибір і обговорення головної ідеї проекту, організація роботи учнів на уроці і позаурочний час, структурування проекту з виділенням підзадач для певних груп учнів, підбір необхідних матеріалів, безпосередня робота над проектом, підведення підсумків.

Напрямок використання методу проектів, пов'язаний із створенням моделей функціонування реальних систем, спрямований на поглиблене вивчення деякого наукового матеріалу, на практичне застосування, удосконалення, формування умінь учнів, проводиться в межах одного класу, групи (2-5 осіб) або індивідуально окремим учнем. Практика показує ефективність застосування таких проектів в навчальній діяльності. Так, наприклад, помічено, що максимальні знання із навчаючої програми отримують розробники цієї системи, а не учні, для яких ця програма призначена. В ході розробки моделі системи або навчаючої програми відбувається "перетворювання знань, що використовуються, вони повертаються різними своїми аспектами і становляться глибшими" [137, с.86].

Робота над проектом із створення моделі функціонування реальної системи може проводитись двома способами: 1) завдяки застосуванню інформаційно-рецептивного методу - на уроці при поясненні нового матеріалу в ході евристичної бесіди разом із вчителем всі учні класу на своїх робочих місцях створюють одну і ту ж саму модель, програму, можливо на основі заготовок, заздалегідь зроблених вчителем; 2) в ході самостійної роботи учнів або на практичних заняттях, або в позаурочний час.

В першому випадку учні мають можливість разом із вчителем пройти всі етапи створення програмного проекту від початку до кінця, слідкуючи за логікою його мислення. В результаті діяльності вчителя перед учнем виникає

мета уважно слухати, дивитись, здійснювати практичні дії з програмними об’єктами, зрозуміти і запам’ятати отриману інформацію. Виконання цієї діяльності викликає в учня психологічні процеси, в ході яких розвиваються його воля, увага, пам’ять, мислительні операції, за допомогою яких він усвідомлює інформацію.

На основі зазначеного підходу до реалізації методу проектів урок складається з таких частин:

1) постановка перед учнями задачі, що містить в собі певну проблему;
2) актуалізація попередньо засвоєних знань учнів про можливості реалізації елементів проекту, виявлення необхідності удосконалення відомих способів реалізації зазначених в умові задачі дій через використання нових засобів, про які вчитель має повідомити учнів;
3) пояснення теоретичного матеріалу про нові засоби реалізації проекту;
4) створення проекту під керівництвом вчителя з використанням тільки окремих елементів з розглянутих засобів;
5) проведення порівняльного аналізу невикористаних засобів з метою виявлення можливостей їх застосування в проекті, що розробляється, з урахуванням збереження виконання визначених в умові завдання дій;
6) самостійна робота учнів із заміни в проекті одних засобів іншими, наприклад, при вивчені систем візуального програмування – одних компонент іншими.

На останніх етапах репродуктивний рівень засвоєння знань перетинається з конструктивним: учням пропонується побудувати фрагмент проекту іншими засобами (за допомогою інших компонент), у процесі чого вони повинні оцінити раціональність або нераціональність такої заміни. Така організація навчальної діяльності учнів потребує від вчителя створення попередніх заготовок. Якщо передбачається створення комплексного проекту, то слабкі учні можуть не встигати з реалізацією проміжних фрагментів проекту. В цьому випадку невстигаючих учнів доцільно ознайомити з

можливою реалізацією цього фрагменту, яка входить до складу тих заготовок, що готує вчитель на урок. Тобто надати при необхідності учням зразок виконуваних дій.

В додатку Б наводиться приклад реалізації зазначеного підходу при вивченні засобів створення програмних проектів в системі візуального програмування Delphi. Вчитель в окремому файлі створює зображення окремого графічного об’єкта, наприклад, бджоли (тому зазначений проект має назву “Політ бджоли”). Цей файл підключається до створюваного учнем проекту, в якому він використовує визначені вчителем допоміжні алгоритми як оператори деякої мови програмування. Компоненти Delphi, що вивчаються на уроці, розглядаються як засоби керування переміщенням графічного об’єкта (польотом бджоли). Це комплексний проект, розрахований на кілька уроків (8 годин), передбачає застосування на окремих етапах свого створення компонент різного функціонального призначення.

В процесі роботи над подібними проектами учням доводиться виконувати інтелектуальні дії: аналізу і синтезу при усвідомленні проблеми, розгляді окремих засобів; прогнозування, порівняння, проведення аналогій при виявленні можливостей для їх подальшого використання тощо.

Для сильніших учнів більш цікавою буде друга форма роботи за методом проектів – самостійна робота над створенням комп’ютерних моделей реальних систем, що передбачає як індивідуальну так і групову роботу. Робота над таким проектом проводиться за такими етапами:

1. Постановка задачі. Усвідомлення практичної значущості вибраної теми проекту виступає основним мотиваційним аспектом для зацікавлення учня, для спонукання його до розв’язування багатьох проблем, з якими доведеться зіткнутися у процесі роботи над проектом.

2. Осмислення теми проекту, самостійне вивчення теорії, дослідження явищ реального середовища, функціонування системи, що моделюється, тощо. При

груповій роботі над проектом на цьому етапі відбувається розподілення ролей між членами групи, задача розбивається на ряд підзадач.

3. Побудова математичної, інформаційної моделі. Робота над алгоритмом реалізації задачі у вибраному середовищі програмування.

4. Проведення уроку-конференції, на якій учні роблять доповідь, показують свою програму, отримують оцінку з боку своїх однокласників.

До створюваних задач-проектів згідно з методологією методу проектів висуваються наступні вимоги: 1) наявність значущої в дослідницькому і творчому плані проблеми, розв'язування якої передбачає застосування інтегрованого знання; 2) наявність практичної, теоретичної, пізнавальної значущості результатів, що очікуються; 3) структурований зміст проекту (бажано з указуванням поетапних результатів).

Так, наприклад, програмний проект "Озеленення міста" передбачає створення навчально-контролюючої програми, метою якої є ознайомлення учнів з видами рослин, які доцільно висаджувати біля окремих об'єктів міста (заводів, лікарень, шкіл, дитячих садків, в парках тощо). Не викликає сумнівів наявність такої значущої проблеми, як ознайомлення людства з одним із методів поліпшення екологічної ситуації в місті. Від учнів в дослідницькому плані вимагається: збір інформації про дерева, які використовуються для озеленення міст, вивчення їх особливостей, впливу на екологічний стан міста, розподілення їх за групами для висаджування біля окремих об'єктів господарського, промислового значення. Так, шумопоглинаючі дерева доцільно висаджувати біля доріг, антимікробні – біля лікарень тощо. Структурований зміст проекту включає наступні блоки: ознайомлення з деревами, що застосовуються для озеленення міста, створення комп'ютерної моделі міста, на яку в процесі роботи програми наносяться позначки розташування відповідних господарських та промислових об'єктів, автоматичне нанесення зображень відповідних рослин навколо визначених об'єктів (озеленення), перегляд списків дерев, які доцільно було б замовити

для озеленення змодельованого міста, контролююча частина програми для перевірки якості засвоєння знань з екології. Цей проект передбачає значну роботу з графічними об’єктами при створенні моделі міста, з компонентами управління при реалізації контролюючої частини проекту. У зв’язку з цим проект має також і практичне значення з точки зору інформатики і може бути використаний при вивченні засобів реалізації графічних зображень, елементів управління в системі візуального програмування, наприклад, Delphi. Принципи реалізації цього проекту та приклади інших тем-проектів наводяться в додатку В.

Робота над будь-яким проектом грунтується на використанні дослідницького методу, який передбачає розвиток уміння освоювати оточуючий світ на основі наукової методології, що є однією із задач загальної освіти. Навчальний проект на основі загальнонаукового методологічного підходу структурується наступним чином: визначення мети і формулювання гіпотези про можливі способи розв’язування поставленої проблеми і передбачуваних результатів, уточнення виявлених проблем і визначення процедури збору і обробки необхідних даних, збір інформації, її обробка і аналіз отриманих результатів, підготовка відповідного звіту і обговорення можливостей застосування отриманих результатів, створеного програмного забезпечення.

Реалізація методу проектів і дослідницького методу на практиці призводить до зміни позиції вчителя. Із носія готових знань він перетворюється в організатора пізнавальної діяльності своїх учнів. Робота учнів набуває характеру самостійної, дослідницької, пошукової, творчої діяльності. Тобто вчитель виступає не в ролі викладача, що навчає учнів, а виконує функції координатора, консультанта, надаючи поради більше технічного характеру з реалізації проекту.

Прагматична спрямованість методу проектів на результат розв’язування проблеми, який можна побачити, осмислити, застосувати в реальній

практичній діяльності, навчає учнів самостійно мислити, знаходити і розв’язувати проблеми, використовуючи для цього знання з різних галузей, уміння прогнозувати результати і можливі наслідки розв’язування, уміння встановлювати причинно-наслідкові зв’язки.

Отже, метод проектів надає значні можливості для: 1) формування дослідницьких умінь: збирати необхідну інформацію, висувати гіпотезу, робити висновки і умовиводи, використовувати для роботи з інформацією нові інформаційні технології; 2) навчання самостійному мисленню і діяльності, системному підходу у самоорганізації, груповій взаємодії; 3)підвищення мотивації навчання та самонавчання: актуалізація незатребуваних знань та стимуляція надбання нових; 4) реалізації методики випереджуючого навчання; 5) формування особистісних якостей учнів: взаєморозуміння, взаємоповага, відповідальність; 6) організації творчої діяльності учнів; 7) структурування знань учнів шляхом встановлення міжпредметних зв’язків; 8) вироблення навиків ефективного використання комп’ютера в своїй подальшій професійній діяльності; 9) формування проектної культури (знань, умінь визначення потреб і можливостей діяльності при виконанні проекту, висування спектру ідей виконання проекту, вибір оптимальної ідеї, дослідження цієї ідеї, планування, організація і виконання роботи з реалізації проекту, включаючи набуття додаткових знань та умінь, оцінка і його презентація); 10) формування інформаційної культури (знання, уміння використовувати методи збору, зберігання, обробки та використовувати інформацію з різних джерел).

2.2.2. Реалізація методу проектів на основі ТООП

Як зазначалося вище, метод проектів припускає індивідуальну і групову організацію роботи учнів. Перевага групової форми роботи обгрунтовується не тільки виховною метою формування таких особистісних якостей учнів як

взаєморозуміння, взаємоповага, відповідальність, а і складністю реальних систем. Реальні системи можуть бути ієрархічними, складатися із взаємозалежних підсистем, які у свою чергу також можуть бути розділені на підсистеми, і т.д. Обсяг короткочасної пам'яті обумовлює здатність людського мозку одночасно слідкувати тільки за 5-7 структурними одиницями інформації. У зв'язку з цим проблема створення моделі складних реальних систем може бути вирішена в процесі колективної організації роботи шляхом розділення її на частини так, щоб одна частина найменше впливала на іншу.

Більшість методів, що використовуються для проектування складних реальних систем, які передбачають розділення її на частини, поділяються на три групи: метод структурного проектування зверху вниз, метод потоків даних (для систем інформаційного забезпечення), об'єктно-орієнтоване проектування. Передбачається, що в своїй попередній навчальній діяльності учні вже ознайомленні з особливостями структурного (при вивченні допоміжних алгоритмів) та об'єктно-орієнтованого (при вивченні основ об'єктно-орієнтованого програмування) проектування. Для забезпечення вибору учнями об'єктно-орієнованого методу для розв'язування власних задач-проектів доцільно провести порівняльний аналіз і навести переваги об'єктно-орієнованого методу порівняно із структурним.

Під структурним проектуванням розуміють форму проектування, при якій логіка програми може бути подана комбінацією тільки трьох базових структур, до яких належать: функціональні елементи чи їх лінійна послідовність; розподільна структура (альтернативна чи структура вибору); циклічна структура. Особливість кожної з цих структур – наявність однієї точки входу і однієї точки виходу, і як наслідок: структури можуть об'єднуватися, вкладатися одна в одну будь-яким чином (Р.Миллс, Р.Лингер [160]). В основі об'єктно-орієнтованого проектування "лежить уявлення про те, що програмну систему необхідно проектувати як сукупність взаємодіючих

між собою об’єктів, розглядаючи кожний об’єкт як екземпляр певного класу, причому класи утворюють ієрархію” (Г.Буч [35, с.37]).

Серед переваг застосування об’єктно-орієнтованого проектування слід зазначити: 1) можливість виділення абстракцій і забезпечення обмеженого доступу до даних; 2) наявність засобів для організації паралелізму; 3)наявність засобів організації ієрархічної структури об’єктів; 4) можливість внесення змін у вже налагоджений програмний продукт. При структурному підході до проектування складних систем умовно виділяють п’ять етапів: аналіз, проектування, програмування, тестування, зборка. Внесення в систему наступних змін вимагає послідовного проходження всіх етапів розробки. Об’єктно-орієнтований підхід дозволяє суттєво спростити процес розробки. Це послідовний ітеративний процес, в якому результати одного з етапів (аналіз, проектування, еволюція, модифікація) можуть впливати на рішення, прийняті на попередніх (рис.2.6) [245, с.3].

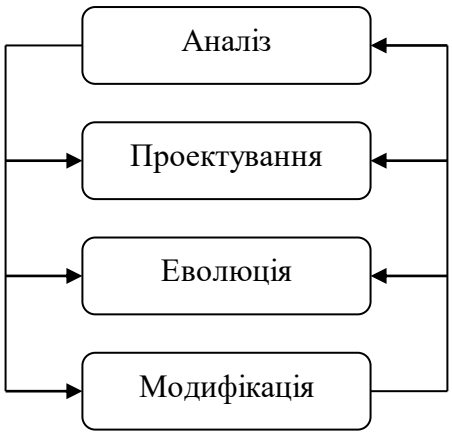


Рис.2.6. Цикл розробки програмного забезпечення на основі ООТП.

Для вчителя використання учнями об’єктно-орієнтованого підходу для розв’язування задач-проектів є одним із засобів застосування, удосконалення інтелектуальних умінь, формування яких проводилося у процесі вивчення основ об’єктно-орієнтованого програмування. Проведення об’єктно-орієнтованого аналізу (ООА) системи, що лежить в основі об’єктно-орієнтованого проектування та програмування передбачає виконання тих же самих інтелектуальних операцій (аналіз, синтез, абстрагування, виділення

головного, порівняння, узагальнення тощо), але в нових умовах, деяких з них- в більш розширеній формі. Окрім цього, розділення системи на майже незалежні підсистеми на основі ООА, дозволяє організувати колективну роботу учнів з чіткім розподіленням обов'язків в групі. Пояснення матеріалу з метою оволодіння методами ООА доцільно проводити з усіма учнями класу на основі проекту однієї з груп. Це викликає як активність учнів групи, проект якої обговорюється, так і інших, які за аналогією можуть проводити ООА власного проекту, а також на основі власного досвіду висувати цікаві ідеї для реалізації чужого проекту.

Виконання об'єктно-орієнтованого аналізу передбачає проходження наступних етапів, кожний з яких вимагає від учнів виконання цілого комплексу інтелектуальних операцій:

1. Постановка задачі (визначення і уточнення вимог до поведінки системи).
2. Виділення об'єктів предметної галузі, класифікація об'єктів та їх властивостей (виділення ключових і додаткових абстракцій, їх класифікація).
3. Визначення зв'язків між об'єктами.
4. Визначення аргументів і результатів (виділення властивостей об'єктів).
5. Побудова алгоритму (реалізація зв'язків між об'єктами через методи об'єктів, побудова діаграм потоків даних і діаграм процесів тощо).

Об'єктно-орієнтований аналіз починається з визначення вимог до поведінки системи, бо саме функціонування (поведінка) системи відомо задовго до виявлення інших її властивостей. Наприклад, при створенні програмного проекту "Бібліотека" у процесі аналізу виділяються такі функції: 1) збереження книжок, журналів, газет, а також інформації про них; 2) пошук книжки за заданою властивістю як серед наявних в бібліотеці, так і серед взятих читачами; 3) видача і повернення книжки; 4) запис нового читача; 5)пошук даних про читача тощо. Виконувані системою функції як обов'язки розподіляються між складовими системи, для нашого прикладу – між Абонементом, Каталогом, Видачею, Книжкою тощо. Так, об'єкт Каталог

зберігає у лінійному списку інформацію про різні видання і відповідає за пошук книжки за прізвищем автора і назвою. У зв’язку з цим можна ознайомити учнів з таким поняттям як проектування на основі обов’язків або проектування на основі відповідальності (Responsibility – Driven – Design (RDD)), що доцільніше проводити на основі порівняльного аналізу зі структурним проектуванням. Це не тільки сприятиме формуванню уявлення учнів про обидва методи проектування, а й спонукатиме їх до виконання операцій аналізу-синтезу при виділенні їх характерних рис, порівняння, узагальнення.

Традиційне структурне програмування грунтується в основному на вказівках “щось зробити”, наприклад, модифікувати запис або обновити масив даних. Тим самим фрагмент програмного коду прив’язаний до багатьох інших розділів програмної системи через реалізацію засобів передачі управління (завдяки використанню глобальних змінних, значень покажчиків) і домовленості щодо структур даних. Проектування на основі відповідальності дозволяє робити ці зв’язки настільки слабкими, наскільки це можливо. Тобто в об’єктно-орієнтованому програмному проекті не повинно бути проблемно-залежних компонентів – воно повинно повністю віддавати окремим фрагментам конкретної системи відповідальність за особливу поведінку.

Так, незначна зміна в структурі даних, наприклад, введення до запису про книжку нового атрибута при структурному програмуванні призведе до необхідності змін в усіх допоміжних алгоритмах, що звертаються до списку книг (Каталогу). Навпаки, при об’єктно-орієнтованому програмуванні зміна структури даних книжки буде стосуватися тільки об’єкта Книга, який як єдине ціле інкапсульованих даних заноситься до списку (до Каталогу) за допомогою методу *Add: Каталог.Add(Нова книга)*. Цей метод в Object Pascal Delphi навіть не треба описувати.

Технологія RDD проектування вимагає від учнів застосування основних принципів її реалізації: модульності і інкапсуляції інформації, правила

виконання яких вже розглядалися при вивченні основ об’єктно-орієнтованого програмування. Тобто раніше сформовані дії знаходять тут своє практичне застосування, а відповідні їм уміння своє перенесення. Під модульністю розуміють властивість системи, яка була розділена на внутрішньо зв’язані, але слабо пов’язані між собою модулі [35, с.69]. Рекомендується в окремому модулі визначати всі структури даних. Доступ до них зробити можливим для всіх процедур цього модуля і закритий для всіх інших. Тому доступ до даних із модуля з інших складових системи повинен здійснюватися тільки через процедури цього модуля. Під інкапсуляцією інформації розуміють ізолювання елементів об’єкта, що визначають його устрій і поведінку, від користувачів. Користувачу відомо тільки те, що даний об’єкт здатний виконувати деякі функції за допомогою відповідних методів. Окрім цього, слід зазначити, що для роботи з даними об’єкта можуть застосовуватися тільки ті операції, які оголошені серед його методів. Тобто, доступ до внутрішнього устрою об’єкта дозволений тільки коду, за допомогою якого реалізовані його методи, що і забезпечує незалежність даних об’єкта від інших складових системи.

Наступний етап ООА передбачає виділення в предметній галузі об’єктів, які виконували б відповідні функції. З цим етапом пов’язуються поняття “об’єкт”, “клас”, “абстракція”, “атрибут”, “домен”, “класифікація об’єктів”, “ключові абстракції” тощо. Визначення цих понять наводяться в додатку Д. Більшість з цих понять вже знайомі учням з вивчення основ об’єктно-орієнтованого програмування, а робота з ними передбачає застосування вже сформованих раніше інтелектуальних умінь (2.1.2), пов’язаних зі створенням об’єкта як окремої абстрактної структури даних. На цьому етапі проектування розширюється поняття учнів про інтелектуальну операцію – класифікацію. Якщо при вивченні основ об’єктно-орієнтованого програмування виконання класифікації передбачало тільки впорядкування об’єктів за рівнями їх ієрархії, то в при ООА класифікацією користуються по-перше як методом виділення класів і об’єктів в предметній галузі на основі

загальних властивостей об’єктів, а вже потім для побудови ієрархічної системи класів.

Процес проведення класифікації містить в собі елементи відкриття та винахідництва. За допомогою відкриття розпізнаються ключові поняття та механізми, що утворюють словник предметної галузі. За допомогою винахідництва конструюються узагальнені поняття, а також нові механізми, які визначають правила взаємодії об’єктів.

Існує кілька методів виконання інтелектуальної операції відкриття, тобто виділення окремих об’єктів, класів в предметній галузі: класичний, аналіз поведінки, аналіз подібної предметної галузі, аналіз варіантів, неформальне описання, CRC картки тощо. Засвоєння учнями методів виділення об’єктів і класів значно розширює їх арсенал мислительних прийомів, користування якими передбачає виконання інтелектуальних операцій, зокрема, аналізу та класифікації, порівняння та аналогії.

Формування цих прийомів можна здійснювати за певною орієнтовною основою дії як в процесі роботи над індивідуальним або груповим проектом, так і за допомогою певного набору завдань. Виконання учнем різних завдань в другому випадку сприяє набуттю багатшого досвіду у виявленні об’єктів з фабули задачі.

Завдання 2.5. Визначити об’єкти, що задіяні в таких ситуаціях:

Ситуація 1. Читач приходить до бібліотеки з метою здати прочитані і взяти нові книжки. Звернувшись до електронного каталогу, він визначає наявність необхідної книжки у фонді бібліотеки за введеним прізвищем автора або за назвою книги. В разі наявності книжки в кількості більше одного екземпляру, вона видається читачеві. Якщо книжка вже видана, то в абонементі бібліотекар знаходяться відповідні дані про читача, який її взяв, і визначає орієнтовну дату повернення шуканої книжки.

Ситуація 2. В закладі харчування шеф-повар за допомогою електронної бази рецептів визначає меню на наступний день, формує список продуктів, які

необхідно закупити. Бухгалтер закладу підраховує вартість блюд згідно із цінами на продукти та процентною надбавкою закладу. Менеджер, турбуючись про престиж закладу, періодично доповнює базу даних новими рецептами.

В умові завдання можна передбачити постановку задачі застосування для виділення об'єктів або одного з розглянутих нижче методів, або кількох. В останньому випадку застосування для розв'язування однієї проблеми різних підходів формує в учнів неординарність, гнучкість мислення. Як зазначалося в 1.1.3, саме ці якості мислення належать до критеріїв інтелектуального розвитку людини. З іншої точки зору, для слабких учнів у процесі ознайомлення з прийомами проведенні ООА буває доречним надання саме такого завдання, яке включає аналіз двох аналогічних ситуацій. Проведення аналізу першої ситуації разом з вчителем надає учневі зразок для мислення, за аналогією з яким самостійно виконується аналіз другої ситуації.

Класичний метод передбачає ідентифікацію об'єктів і класів шляхом розглядання концептуальних сутностей або "предметів", пов'язаних з проблемою, що аналізується, та віднесення їх до відповідних категорій, наприклад, до наступних [278]: 1) реальні об'єкти; 2) ролі; 3) інциденти; 4) взаємодія; 5) місця – абстракції галузей, пов'язаних з людьми або предметами (Місто).

Згідно з цим методом в наведених вище ситуаціях завдання можна, наприклад, виділити такі реальні об'єкти, як Книга, Журнал, Читач, Рецепт, Меню, Продукт харчування; ролі – Бібліотекар, Шеф-повар, Бухгалтер, Менеджер; інциденти – Видача книги, Формування списку продуктів для закупки; місця – Бібліотека, Каталог, Абонемент, База даних рецептів. Але не всі з виділених об'єктів будуть важливими з точки зору моделювання функціонування системи. Наступний метод – метод аналізу дозволяє не тільки визначати об'єкти, а й виділяти головні з точки зору функціонування системи.

При використанні методу аналізу поведінки для виділення об’єктів класи і об’єкти ідентифікуються шляхом аналізу функціонування системи. Зіставляючи форми поведінки із складовими системи, виявляють, яка складова ініціює поведінку і які складові в ньому приймають участь. Ініціатори і співучасники визнаються як об’єкти, що відповідальні за свої ролі.

Так, наприклад, у процесі аналізу функціонування бібліотеки можна з’ясувати, що операція пошуку необхідної книжки за введеним прізвищем її автора або назвою здійснюється в об’єкті Каталог, а пошук даних про читача, який взяв необхідну для іншого читача книгу, виконується у відділі Абонемент. Об’єкти Книги і Читачі у вигляді упорядкованих списків входять відповідно до об’єктів Каталог, Абонемент. В такому самому відношенні в другій ситуації знаходяться об’єкти Рецепт блюда – База даних рецептів, Блюдо – Меню.

Іноді в пошуках корисних ідей щодо виділення об’єктів буває доречним звернутись в межах даної предметної галузі до розроблених додатків, що вже довели свою працездатність. Розгляд подібних систем за аналогією дозволяє зрозуміти, які ключові абстракції і механізми, що використані в них, будуть корисні в новому проекті, а які ні. Відбираються ті абстракції і механізми, що відповідають меті заданої задачі.

Паралельно з трьома попередніми методами буває доцільним використання аналізу варіантів, який передбачає пророблення в думці сценарію функціонування системи. При цьому встановлюється, які об’єкти беруть участь в сценарії, які їх обов’язки, за допомогою яких операцій вони взаємодіють, враховуються виняткові ситуації.

Найпростішим методом щодо виділення об’єктів в предметній галузі є неформальне описання. Згідно з цим підходом рекомендується описати функціонування системи звичайною мовою, а потім підкреслити іменники та дієслова. Іменники будуть кандидатами на роль об’єктів, а дієслова можуть стати іменами операцій.

Але найзручнішим для навчання об’єктно-орієнтованого програмування є виділення окремих об’єктів і класів на основі пророоблення в думці сценарію функціонування системи з записом на відповідних CRC (Component – Responsibility – Collaborator (компонента – обов’язок-співробітники)) картках інформації про виділені об’єкти. Під компонентою будемо розуміти абстрактну одиницю, яка може виконувати деяку роботу (тобто має деякі обов’язки). Потім виявлена компонента може бути перетворена в окрему функцію, структуру, або клас. Для виявлення окремих компонент і визначення їх обов’язків необхідно проаналізувати сценарій роботи системи. Тобто в уяві запустити програму так, якщо вона була б вже готова. Будь-яка дія, що трапляється, приписується деякій компоненті як її обов’язок. Доречніше аналізувати функціонування системи за допомогою циклу питань “що/хто”. Спочатку визначається: “Що необхідно зробити?” Це питання викликає інше: ”Хто буде виконувати цю дію?” На картках зверху записується назва класу або об’єкту, знизу в лівій частині – за що він відповідає, а в правій частині – з ким він співпрацює.

Рис.2.7. Загальний вигляд CRC картки**.**

Рухаючись в уяві за сценарієм роботи системи, на кожний знайдений клас заводиться окрема картка і до нею записуються нові пункти. При цьому можна переносити відповідальність з одного великого класу на кілька

Компонента (назва)	Список компонент, з якими співпрацює
Обов’язки, приписані даній компоненті	

підкласів або передавати частину обов’язків іншому класу.

Далі на основі виділених об’єктів створюються абстракції. На цьому етапі інтелектуальна операція абстрагування тісно переплітається з класифікацією. Побудова абстракцій є роботою інтелектуальною, тому вважається [35, с.150], що найкращий спосіб її проведення є послідовний ітеративний процес. При ітеративному процесі за основу береться деяка визначена структура класів,

яка поступово удосконалюється. На пізнішій стадії розробки проекту, коли вже отриманий деякий досвід використання такої структури, критично оцінюючи якість проведеної класифікації, можна створити новий підклас на основі вже існуючого (виведення), або розділити більший клас на кілька маленьких (факторизація), або об'єднати кілька існуючих класів в одному (композиція). Для проведення класифікації з метою створення окремих абстракцій на основі виділених в предметній галузі об'єктів учнів можна озброїти наступними трьома методами: класична категорізація, концептуальна кластеризація та теорія прототипів. В класичній категоризації всі об'єкти, що мають дану властивість або сукупність властивостей, формують деяку категорію (неперетинну множину). Наявність цих властивостей є достатньою і необхідною умовою, яка визначає категорію. Властивості не обов'язково повинні бути вимірювальними, це може бути і поведінка. Наприклад, та обставина, що птахи літають, а риби ні, дозволяє відрізнити орла від форелі. Якщо за допомогою класичної категоризації не вдається побудувати необхідну структуру класів, тобто об'єкти предметної галузі неможна розбити на непересічні множини, то доцільно звернутися до концептуальної кластерізації, при якій об'єкт одночасно може належить до кількох категорій з різним ступенем точності. Увага концентрується на поведінці об'єктів, коли вони взаємодіють. При такому підході спочатку формуються концептуальне описання класів (кластерів об'єктів), а потім сутність відноситься до того класу, під описання якого вона більше підходить. Але існують абстракції, які не мають ні чітких властивостей, ні чіткої поведінки, наприклад, пуфик і перукарське крісло. В такому випадку слід скористатися теорією прототипів. Клас визначається одним об'єктом-прототипом, і новий об'єкт можна віднести до класу при умові, що він має суттєву подібність з прототипом, як в даному випадку пуфик і перукарське крісло схожі на прототип стілець, тому і відносяться до одного класу. Головним критерієм визначення "родинної" подібності вважаються властивості, що визначаються при взаємодії з об'єктом

(пуфик і перукарське крісло мають теж саме призначення, що і стілець). Визначені абстракції класів і об’єктів за вже знайомим учням правилом-орієнтиром інтелектуальної операції класифікації (2.1.2) розташовуються на відповідних рівнях ієрархії, що зробити зразу дуже важко. Доцільніше рекомендувати учням спочатку створювати кілька незалежних ієрархій, усвідомити їх загальні риси, а потім створити один або кілька суперкласів.

Але ієрархічна залежність (відношення узагальнення) – це не єдине відношення, у якому можуть знаходитись класи і об’єкти. Взагалі виділяються [36, с.200] такі основні відношення між класами: 1) відношення залежності; 2)відношення асоціації; 3) відношення узагальнення; 4) відношення агрегації (відношення композиції). Означення вказаних відношень наводяться в додатку Д. Виявлення відношень, в яких знаходяться об’єкти системи, є основою для визначення структури їх класів. Так, якщо два класи знаходяться у відношенні узагальнення, то один з класів є нащадком іншого, при агрегації- один клас як атрибут входить до структури іншого класу, при асоціації – це незалежні класи, але один клас може бути списком екземплярів іншого класу, відношення залежності може бути оформлено по-різному (в більшості випадків як агрегація).

Важливість виявлення відношень між об’єктами для визначення структури їх класів потребує проведення попереднього ознайомлення учнів з типами відношень між класами на основі тренувальних завдань, наприклад:

Завдання 2.6. Визначити, в якому відношенні знаходять об’єкти:

1. а) бібліотека – абонемент, б) бібліотека – каталог, в) каталог – книга, г)абонемент – читач (Відповіді: а) агрегація, б) агрегація, в) асоціація, г)асоціація);

2. а) кафе – кухня, б) кафе – бухгалтерія, в) меню - блюдо, г) кафе – офіціант; д) продукт (ціна продукту) – блюдо (вартість блюда) (Відповіді: а) агрегація, б) агрегація, в) асоціація, г)асоціація, д) залежність);

3. а) геометрична фігура – прямокутник, б) коло – точка, в) дитяча гра "Конструктор" – куб, г) вікно програми – кнопка, меню, д) кнопка – прямокутник, е) кнопка – назва (графічне зображення і текстовий рядок на кнопці); (Відповіді: а) узагальнення, б) узагальнення, в) асоціація, г)композиція, д) узагальнення, е) агрегація);

4. а) автомобіль – вантажний автомобіль, б) легковій автомобіль – модель ВАЗ-21099, в) автомобіль – мотор, г) автомобіль – бензоколонка, д) пальне (ціна на пальне) – такси (вартість проїзду) (Відповіді: а) узагальнення, б)узагальнення, в) композиція, г) асоціація, д) залежність).

На наступному етапі ООА передбачається визначення зв'язків між об'єктами, механізму їх взаємодії. Під механізмами розуміють зміст засобів (методів), за допомогою яких об'єкти взаємодіють між собою для досягнення поведінки вищого рівня. Реалізація конкретного механізму розкладається на відповідні атрибути і методи класів. В процесі взаємодії перший об'єкт змінює стан другого об'єкта, тобто змінює значення певного атрибута або атрибутів, користуючись його методами. Тому реалізація механізму взаємодії з іншими об'єктами передбачає виділення відповідних атрибутів (абстрактних даних, що відповідають за стан об'єкта) та методів доступу до цих атрибутів.

При описанні алгоритму функціонування складної системи і побудови її архітектури дуже складно охопити одним поглядом всі її деталі. В залежності від типу системи, що моделюється, вона може бути описана за допомогою кількох взаємопов'язаних видів або уявлень про її архітектуру, кожний з яких є однією з можливих проекцій організації і структури системи та загострює увагу на деякому аспекті її функціонування. До таких уявлень про систему належать наступні:

- *вигляд з точки зору прецедентів* охоплює прецеденти, що описують поведінку системи, і виявлені в процесі спостереження за її функціонуванням.

Цей вигляд описує рушійні сили, від яких залежить формування системної архітектури;

- *вигляд з точки зору проектування* охоплює класи, інтерфейси (сукупність операцій, що виконуються класом) та кооперації (сукупність ролей, за допомогою яких здійснюється взаємодія класів), що формують словник задачі. Цей вигляд описує функціональні вимоги до системи, тобто ті послуги, які вона повинна надавати користувачу;
- *вигляд з точки зору процесів* охоплює всі послідовності і процеси, які формують паралелізм та синхронізацію в системі (особлива увага приділяється активним класам, що задіяні в кількох процесах). Цей вигляд описує пропускну здатність системи, її продуктивність;
- *вигляд з точки зору реалізації* охоплює компоненти і файли, що використовуються для зборки і випуску кінцевого програмного продукту. Цей вигляд призначений для управління конфігурацією системи, що складається із незалежних (до деякого ступеня) компонентів і файлів, які можуть по-різному об'єднуватися між собою;
- *вигляд з точки зору розгортання* охоплює вузли, що формують топологію апаратних засобів, на яких вона виконується [36, с.48-49].

Для візуалізації системи з різних точок зору, що допомагає вирішити проблему розбиття проекту реалізації її моделі на окремі модулі, доцільно використовувати UML (Unified Modeling Language) технологію графічного моделювання на основі об'єктно-орієнтованого аналізу, офіційне створення якої почалося у 1994 році при об'єднанні методів проектування трьох видатних авторів Грейди Буча (мова Booch, компанія Rational Software Corporation), Айвара Джекобсона (мова OOSE (Object-Oriented Software Engineering), компанія Objectory) і Джеймса Рамбо (мова OMT (Object Modeling Tecnique), компанія General Electric). У 1997 році UML технологія була прийнята за стандарт об'єктно-орієнтованого проектування.

Основу UML технології становлять діаграми. Діаграма – це графічне представлення набору елементів системи, що найчастіше зображуються у вигляді зв'язаного графа з вершинами (сутностями) і ребрами

(відношеннями). Діаграма – це одна з проекцій системи, що дає згорнуте представлення елементів, з яких складається система. Один і той же елемент може належати всім діаграмам, або тільки в кількох, або не належати жодній. У відповідності з перерахованими вище видами описання системи, в [36, с.42-49] виділяються наступні типи діаграм (описані в додатку Е): 1)діаграма класів (описує вигляд з точки зору проектування); 2) діаграма об'єктів (описує вигляд з точки зору проектування, процесів); 3) діаграма прецедентів або випадків використання (описує вигляд з точки зору прецедентів); 4) діаграма функцій (описує вигляд з точки зору прецедентів); 5)діаграма послідовностей (описує вигляд з точки зору процесів); 6) діаграма кооперацій (описує вигляд з точки зору прецедентів, проектування, процесів); 7) діаграма діяльності (описує вигляд з точки зору прецедентів, проектування, процесів); 8) діаграма станів (описує вигляд з точки зору прецедентів, проектування, процесів); 9) діаграма модулів (описує вигляд з точки зору реалізації); 10) діаграма розгортання (описує вигляд з точки зору розгортання).

При розгляді статичних складових системи використовуються діаграми класів, об'єктів, розгортання. Для роботи з динамічними частинами системи використовуються діаграми прецедентів, послідовностей, кооперації, стану, діяльності. Діаграми послідовностей і кооперацій іноді об'єднуються в одну діаграму взаємодії тому, що в них представлені зв'язки між об'єктами, повідомлення, якими об'єкти можуть обмінюватися. Під взаємодією розуміють поведінку об'єктів, при якій об'єкти обмінюються між собою повідомленнями в межах конкретного контексту для досягнення визначеної мети.

При моделюванні системи з різних точок зору фактично здійснюється її конструювання зразу в кількох вимірюваннях. Правильний вибір сукупності видів або уявлень дозволяє задати необхідні питання, що стосуються системи, виявити ризик, який необхідно врахувати. Але не завжди всі можливі діаграми доцільно створювати при моделюванні системи. Наприклад, при моделюванні простого додатку, що буде виконуватися на одному комп'ютері, можуть знадобитися тільки діаграми прецедентів, класів, взаємодії, компонентів

(модулів). Якщо система будується на архітектурі “клієнт/сервер”, то в роботу з її створення доречно буде додатково включити діаграму розгортання для моделювання конкретних фізичних деталей реалізації. Діаграми будуються за допомогою відповідних позначень. В додатку Е наводяться приклади побудов зазначених діаграм.

Попереднє ознайомлення учнів з UML технологію проектування доцільно проводити на основі завдань, що передбачають побудову за заданою діаграмою описаної в умові системи іншої діаграми тієї ж системи (наприклад, за діаграмою об’єктів побудувати діаграму класів, за діаграмою випадків використання – діаграму функцій, за діаграмою функцій – діаграму діяльності, за діаграмами діяльності і класів – діаграму кооперації тощо). З одного боку учні будуть мати приклад (зразок) побудови однієї діаграми, а з іншого будуть самостійно працювати над створенням іншої, близької за виконуваною функцією. В такому завданні репродуктивна діяльність переходить в продуктивну, яка знаходить своє продовження, а набуті уміння і знання з основ UML технології – своє засування, в подальшій діяльності із створення власних програмних проектів

Проведення ООА на основі UML технології рекомендуємо виконувати за наступними послідовними етапами:

1. *Описання вимог до програми*, яка моделює функціонування системи. На цьому етапі описується що і для кого повинна робити система. Спочатку будується діаграма прецедентів, на яку наносяться випадки використання системи і “актори”, які користуються системою в даних випадках. Потім для кожного окремого випадку використання будується діаграма функцій, які повинні виконуватися системою в даному випадку використання.

2. *Описання схеми архітектури і основних сценаріїв роботи системи.* При використанні об’єктно-орієнтованого підходу виконання функцій системи реалізується як сумісна діяльність кількох об’єктів. На цьому етапі для кожної діаграми функцій, що створені на попередньому етапі будується діаграма об’єктів, основне призначення якої – описання об’єктів-ролей, які є

елементами системи, та зв’язків між ними. Після цього для кожної діаграми об’єктів будується діаграма послідовностей або діаграма сценаріїв взаємодії та діаграма кооперацій, на яких показується, як, за допомогою яких повідомлень, в якій послідовності створені об’єкти реалізують функції системи.

3. *Уточнення архітектури системи.* На цьому етапі будується діаграма класів, яка описує внутрішню структуру системи, класи, що беруть участь в реалізації системи, і зв’язки між ними. Окремий клас створюється для кожного об’єкта-ролі, визначаються атрибути та методи класів.

4. *Описання алгоритмів поведінки об’єктів.* На основі побудови діаграм діяльності та станів описується поведінка системи, визначаються процеси, що протікають в системі в термінах станів, подій і дій. Діаграми будуються або для кожного класу, або для прецеденту, або для системи в цілому.

5. *Описання конфігурації системи.* На цьому етапі будується діаграма модулів, в яких розміщуються реалізація виділених класів, об’єктів, їх інтерфейсів. На основі визначених модулів можна розподілити подальшу роботу над проектом між учнями групи.

Процес побудови кожної з діаграм передбачає виконання учнями відповідних інтелектуальних операцій, що доводить доцільність застосування UML технології при створенні нових умов для застосування учнями сформованих інтелектуальних умінь, для їх удосконалення, а також як засобу розв’язування задач-проектів. Залежність між діями, які виконуються учнями при побудові діаграм і інтелектуальними операціями наводиться в таблиці 2.2. Використання UML технології дозволяє розділити процес з створення проекту на окремі етапи, результати проходження яких можна враховувати при оцінюванні рівня сформованості інтелектуальних умінь учнів.

Умовні позначення для таблиці 2.2:

1- Аналіз, синтез; 2- Абстрагування; 3- Порівняння; 4- Виділення головного, суттєвого; 5- Узагальнення; 6- Класифікація; 7- Конкретизація; 8- Прогнозування; 9- Перевірка (критичність мислення).

Таблиця 2.2.

Інтелектуальні операції, які виконуються учнями при побудові UML діаграм.

Виконувані при побудові діаграм дії	Інтелектуальні операції								
	1	2	3	4	5	6	7	8	9
Діаграма прецедентів або випадків використання									
Визначення суб'єктів (акторів), яким необхідна система для виконання своїх задач., визначити ці задачі	+			+		+	+	+	
Визначити ці задачі або випадки використання системи	+			+		+	+	+	
Організація схожих акторів у групи за допомогою відношення узагальнення/спеціалізація	+		+	+	+	+			
Визначення зв'язків між акторами і випадками використання системи	+			+				+	
Діаграма функцій									
Визначення функцій і їх підфункцій для кожного випадку використання системи	+				+	+	+	+	
Діаграма об'єктів									
Визначення об'єктів, які беруть участь у виконанні окремої функції системи	+					+	+		
Розгляд, виконання в уяві сценарію при виконанні цієї окремої функції	+						+	+	
Визначення значень атрибутів об'єктів при виконанні сценарію							+		
Визначення зв'язків між об'єктами, їх взаємодії	+						+		
Діаграма послідовностей або сценарію взаємодії									
Визначення сцени взаємодії при виконанні деякого сценарію	+							+	

Продовження табл.2.2

Виконувані при побудові діаграм дії	Інтелектуальні операції								
	1	2	3	4	5	6	7	8	9
Визначення лінії життя для тих об'єктів, що створюються або вилучаються в процесі взаємодії	+			+			+	+	

Визначення і розміщення зверху вниз повідомлень, якими обмінюються об’єкти в процесі взаємодії, починаючи з повідомлення, що ініціює цю взаємодію	+			+		+	+	+	
Діаграма кооперацій									
Визначення сцени взаємодії при виконанні деякого сценарію	+							+	
Визначення початкових значень атрибутів об’єктів, що беруть участь у взаємодії							+	+	
Визначення зв’язків між об’єктами	+			+			+	+	
Виявлення повідомлень, які змінюють значення атрибутів об’єктів при взаємодії, визначення цих нових значень	+			+			+	+	
Позначення цих повідомлень певними номерами, впорядкування їх за часом виникнення						+		+	
Діаграма класів									
Розподілення об’єктів за відповідними групами, класами		+	+	+	+	+			
Визначення атрибутів і методів певних класів	+	+	+	+					
Визначення зв’язків між класами	+			+	+	+			
Визначення інтерфейсу класу для реалізації певних зв’язків	+						+	+	
Діаграма діяльності									
Виділення деякого проміжку робочого процесу	+							+	
Визначення передумов для початкового та постумов для кінцевого станів робочого процесу	+		+	+			+	+	
Зображення послідовності дій робочого процесу, починаючи з початкового стану, та переходів між ними	+				+	+		+	
Згортання множини дій, що повторюються в одну складну дію	+			+	+				
Розгортання складних дій на окремих діаграмах діяльності	+						+		

Продовження табл.2.2

Виконувані при побудові діаграм дії	Інтелектуальні операції								
	1	2	3	4	5	6	7	8	9
Діаграма станів									
Визначення початкового і кінцевого станів об’єкта	+	+	+	+			+	+	

Визначення стійких станів, тобто тих, в яких об’єкт перебуває протягом деякого часу	+	+	+	+			+	+	
Упорядкування стійких станів на протязі життєвого циклу об’єкта						+		+	
Визначення подій, що ініціюють перехід об’єкта із одного стану до іншого, зображення їх як зв’язків між станами	+			+				+	
Перевірка досяжності будь-якого стану при деякій комбінації подій, відсутності тупикових станів									+
Діаграма модулів									
Розбиття системи на окремі частини з урахуванням можливості управління конфігурацією системи та повторного використання її компонентів	+	+						+	
Визначення елементів, які будуть реалізовані в цих окремих частинах, в процесі уявного моделювання роботи системи	+		+	+	+	+			
Визначення зв’язків між модулями та інтерфейсів для їх реалізації	+					+		+	
Перевірка діаграми на відсутність циклів									+

Згідно з таблицею 2.2 для перевірки рівня сформованості уміння виконувати певні інтелектуальні операції можна використовувати ті діаграми, в побудові яких вони задіяні (позначені символом “+”). Так, уміння виконувати аналіз і синтез перевіряються при побудові всіх зазначених діаграм, а уміння виконувати абстрагування - при роботі над діаграмами: класів, станів, модулів і т.д. Самостійна побудова учнем всіх відповідних діаграм буде свідчити про високий рівень сформованого інтелектуального уміння (5 балів згідно зі шкалою вимірювання за Мар’яненко, 1.1.3). Побудова всіх необхідних діаграм при наданні підказки з боку вчителя - 4 бали. Побудова окремих діаграм або самостійно, або з незначною допомогою вчителя – 3 бали. Створення діаграм у співробітництві з вчителем, в ході евристичної бесіди – 2 бали. Невміння працювати над діаграмами – 1 бал.

Мотиваційним аспектом застосування учнями UML технології в своїй практичній діяльності при роботі над проектом є усвідомлення можливості

використання набутих умінь в подальшій професійній діяльності. UML технологія дуже поширена в CASE (Computer Aided System/Software Engineering) системах, які використовуються для автоматизації, візуалізації проектування, описання і документування при розробці програмного забезпечення. Графічні засоби CASE-систем моделювання предметної галузі дозволяють розробникам в наглядному вигляді вивчати інформаційну систему, перебудовувати її згідно з поставленою метою і наявними обмеженнями.

Існує кілька типів CASE-систем в залежності від функціональної орієнтації CASE-засобу на ті чи інші процеси життєвого циклу програмного забезпечення (аналіз і проектування, розробка додатку (генерація програмного коду), докуметування, тестування тощо):

- засоби аналізу (Upper CASE), призначені для побудови і аналізу моделей предметної галузі (Design/IDEF (Meta Software), Bpwin (Logic Works));
- засоби аналізу і проектування (Middle CASE), що використовуються для створення проектних специфікацій (Vantage Team Builder (Cayenne), Designer/2000 (ORACLE), Silverrun (CSA), PRO-IV (McDonnell Douglas), CASE.Аналитик (МакроПроджект), Rational Rose (Rational Software Corp.), ObjectiF (MicroTool), Real (Ланит-Терком), Visual UML (Visual Object Modelers)). Виходом таких засобів є специфікації компонентів і інтерфейсів системи, архітектури системи, алгоритмів і структур даних;
- засоби проектування баз даних, що забезпечують моделювання даних і генерацію схем баз даних (як правило, на мові SQL) для найбільш розповсюджених СУБД. До них відносяться ERwin (Logic Works), S-Designor (SDP) і DataBase Designer (ORACLE). Засоби проектування баз даних входять до складу наступних CASE-засобів Vantage Team Builder, Designer/2000, Silverrun и PRO-IV;
- засоби розробки додатків (Uniface (Compuware), JAM (JYACC), PowerBuilder (Sybase), Developer/2000 (ORACLE), New Era (Informix), SQL

Windows (Gupta), Delphi (Borland) тощо) та генератори кодів, що входять до складу Vantage Team Builder, PRO-IV, Silverrun, Real (Java), Visual UML (Visual Basic);

- засоби реінжинірингу, що забезпечують аналіз програмних кодів і схем баз даних, формування на їх основі різних моделей і проектних специфікацій. Засоби аналізу схем БД входять до складу Vantage Team Builder, PRO-IV, Silverrun, Designer/2000, ERwin і S-Designor. Засоби Rational Rose (Rational Software), Object Team (Cayenne) забезпечують реінжиніринг програм на мові C++. Visual UML забезпечує реінжиніринг програм з Visual Basic.

До CASE-засобів, що реалізують тільки UML технологію належать Rational Rose (Rational Software Corp.), ObjectiF (MicroTool), Visual UML (Visual Object Modelers), а до тих, що застосовують UML технологію в комбінації з іншими належать Geode (Verilog), CASE/4/0 (MicroTool), Real (Ланит-Терком). В таблиці 2.3 наводяться типи діаграм, що використовуються в цих CASE–засобах при створенні програмного забезпечення.

Таблиця 2.3.

Типи діаграм CASE–засобів.

Тип діаграм	Rational Rose	ObjectiF	CASE/4/0	Geode	Real	Visual UML
Діаграма прецедентів або випадків використання	+	+	-	-	+	+
Діаграма функцій	+	-	+	-	+	-
Діаграма потоків інформації	-	-	+	-	-	-
Діаграма об’єктів	+	+	++	+	+	+
Діаграма сценаріїв взаємодії	+	+	-	++	+	+
Діаграма діяльності (STD)	+	+	Слабо	-	+	+
Діаграма станів (SDL)	-	-	-	++	+	+
Діаграма розгортання	-	-	-	-	-	+

Умовні позначення, використані в таблиці 2.3:

“-“ – діаграма не використовується, “+” – діаграма використовується в даному CASE-засобі, “++” – елемент засобу найкраще реалізований.

Ознайомлення учнів з CASE-системами доцільно проводити, якщо сформовані на високому рівні загальні інтелектуальні уміння і уміння будувати діаграми, якість яких залежить від загального рівня інтелектуальних.

Застосування CASE-систем при проектуванні слабкими і середніми учнями ускладнює процес аналізу системи, знижує мотивацію діяльності, а тим самим не сприяє формуванню їх інтелектуальних умінь. При побудові діаграм за допомогою CASE-системи виконання загальних інтелектуальних дій залежить від умінь користуватися цією CASE-системою, які розраховані на професійних розробників програмного забезпечення, в більшості випадків мають складний інтерфейс та непрості інструменти. Сильні учні, навпаки, сприймають CASE-системи, як засіб, що автоматизує їх роботу при побудові діаграм, звільняє від необхідності вручну на папері малювати ці діаграми.

Згідно з таблицею 2.3 найповнішій комплекс діаграм UML технології надають дві CASE –системи: Real і Visual UML. Перевагою Real є можливість роботи з інтерфейсом російською мовою, недоліком будемо вважати необхідність ведення описання класів в термінах C++ та Java при умові, що вся інша робота над проектом проводиться мовою Object Pascal, як більш простої і доступної для вивчення і використання учнями. Недоліком Visual UML є англомовний інтерфейс, але надається можливість проводити описання класів, задіяних в проекті, в термінах мови Object Pascal, що спрощує подальшу роботу над реалізацією проекту в Delphi. Але існування спеціального “моста” – програми Rose Delphi Link (Ensemble Systems), що інсталюється в середовище Rational Rose і надає можливість для генерації програмного коду на Delphi та зворотного реінжирингу, робить доцільним застосування Rational Rose для проектування також. Основні правила роботи в Visual UML та в Rational Rose з Rose Delphi Link описані в додатку Ж.

2.3. Формування інтелектуальних умінь учнів у процесі вивчення програмного засобу (на прикладі середовища візуального програмування Delphi)

2.3.1. Особливості середовища візуального програмування Delphi

Об’єктно-орієнтовані середовища широко використовується не тільки для програмування, а й для створення анімаційних файлів, розробки Web-сторінок тощо. Тому важливим є ознайомлення учнів з методами дослідження елементів цих середовищ, які повніше можна показати на прикладі середовища візуального програмування Delphi.

Об’єктно-орієнованое середовище візуального програмування Delphi містить зручні для використання учнями засоби (компоненти) створення програмної реалізації проектів. Значна їх кількість (тисячі) дозволяє вчителю організувати їх вивчення учнями з метою використання у власних проектах на основі застосування дослідницького методу, методу аналогії, що створює нові умови для подальшого формування та удосконалення інтелектуальних умінь учнів.

Доцільність застосування Delphi для комп’ютерного моделювання, що здійснюється при розв’язуванні задач-проектів, визначається його належністю до систем RAD (Rapid Application Development) програмування (систем швидкої розробки прикладних програм), до яких, окрім Delphi, ще належать Visual Basic, Visual C++, Bridge View, C Builder тощо. В RAD системах програмістам надаються наочніші і інтуїтивно зрозуміліші засоби, ніж в звичайних середовищах програмування. У зв’язку з цим розробка прикладних програм значно спрощується. Нелегку працю програміста з написання програмного коду часто вдається замінити на натиснення кнопки миші, що робить процес створення складних програм більш доступним для школярів і дозволяє приділяти більше часу творчому мисленню.

Вибір середовища візуального програмування Delphi як засобу для програмної реалізації учнями власних проектів та формування їх інтелектуальних умінь обгрунтовується його переконливими перевагами та особливостями.

Delphi – це перспективна система програмування. Delphi працює в сучасному середовищі, має візуальний характер, базується на об'єктно-орієнтованому програмуванні, яке зараз переживає період швидкого зростання. При підборі програмних засобів для навчання важливою є не стільки їх сучасність, скільки перспективність. Для комп'ютерної галузі, яка розвивається швидкими темпами, що надзвичайно важливо. Перспективність Delphi в створенні Windows–орієнтованих програмних додатків спонукала її авторів (фірма Borland International, Inc (США)) до його перенесення на платформу Linux (Borland Kylix). У зв'язку з цим в майбутньому вчителі матимуть змогу використовувати розроблені компоненти методичної системи незалежно від того, на якій операційній системі (Windows чи Linux) грунтується процес навчання інформатики.

В основі Delphi лежить простіша для школярів, порівняно з C++, мова Object Pascal, і, відповідно, витримується мовна лінія Н.Вірта, дотримання якої дозволяє прищеплювати учням програмістську культуру в процесі навчання. Пітер Нортон вважає, що сам по собі Pascal кращий, в ньому менше можливостей зробити помилку. Таким чином мова Pascal найбільше підходить для цілей навчання мови програмування високого рівня, яка в той же час є достатньо потужною і використовуються професіональними програмістами. Так, наприклад, "Утілити Нортона" були спочатку написані мовою Pascal, а потім переписані на С.

Можна сказати, що синтаксис мови Visual Basic є наближеним до Pascal, тобто мова, що лежить в основі Visual Basic, така ж проста у використанні. Але згідно з визначенням об'єктно-орієнтованої мови Л.Карделлі і П.Вегнера (2.1.2.) мова програмування середовища Visual Basic є тільки об'єктною, вона використовує об'єкти і методи, але не підтримує основних концепцій об'єктно-орієнтованого програмування, таких як інкапсуляцію, успадкування і поліморфізм. Мова в основі Delphi (Object Pascal) – насправді об'єктно-орієнтована. Вона дозволяє об'єднати дані і код в один клас (інкапсуляція), створити класи-нащадки (успадкування) і використовувати класи-нащадки, як батьківські класи для інших класів. У

зв’язку з цим її доцільніше використовувати для формування загальних інтелектуальних умінь учнів (2.1.2).

Згідно з дослідженнями компанії Carnegie Technology Group, що проводились з метою оцінювання продуктивності засобів розробки клієнтських додатків, Delphi, порівняно з Visual Basic, швидше виконує операції: обробку розгалужень, роботу з цілими числами, обробку складних виразів, виклик внутрішніх процедур та процедури роботи з дисплеєм [234]. Порівняльний аналіз засобів швидкої розробки додатків Visual Basic та Delphi подано в таблицях 2.4 і 2.5 та на рис. 2.8 і 2.9.

Таблиця 2.4.

Продуктивність програмного коду (млс)

Продуктивність програмного коду	Visual Basic	Delphi
Обробка циклу	0,00102	0,00133
Обробка розгалужень	0,00156	0,00084
Робота з цілими числами	0,00148	0,00065
Робота з числами з плаваючою точкою	0,00276	0,00783
Робота з числовими функціями	0,00438	0,00757
Обробка складних виразів	0,01205	0,00896
Виклик внутрішніх процедур	0,00624	0,0042
Виклик зовнішніх процедур	0,00356	0,00466

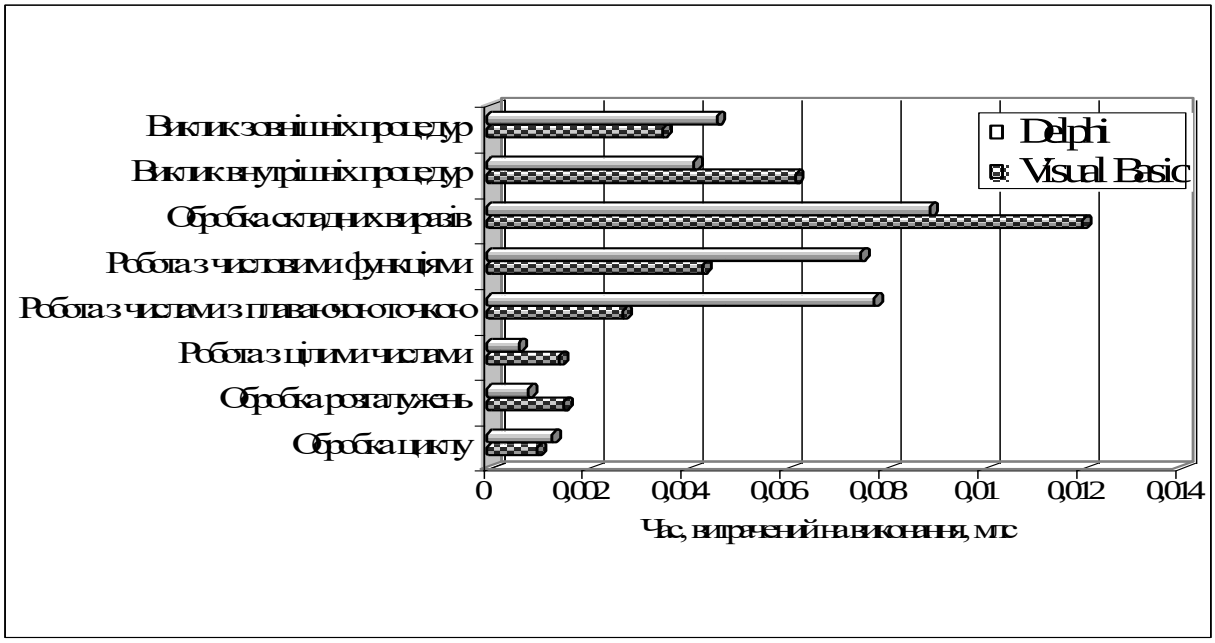


Рис.2 8. Діаграма продуктивності програмного коду Delphi і Visual Basic.

Таблиця 2.5.

Продуктивність роботи з дисплеєм (млс)

Ефективність роботи з дисплеєм	Visual Basic	Delphi
Відображення пустих вікон	179,71	158,51
Відображення текстової інформації у вікні	17,52	14,32
Відображення графічної інформації у вікні	13,84	14

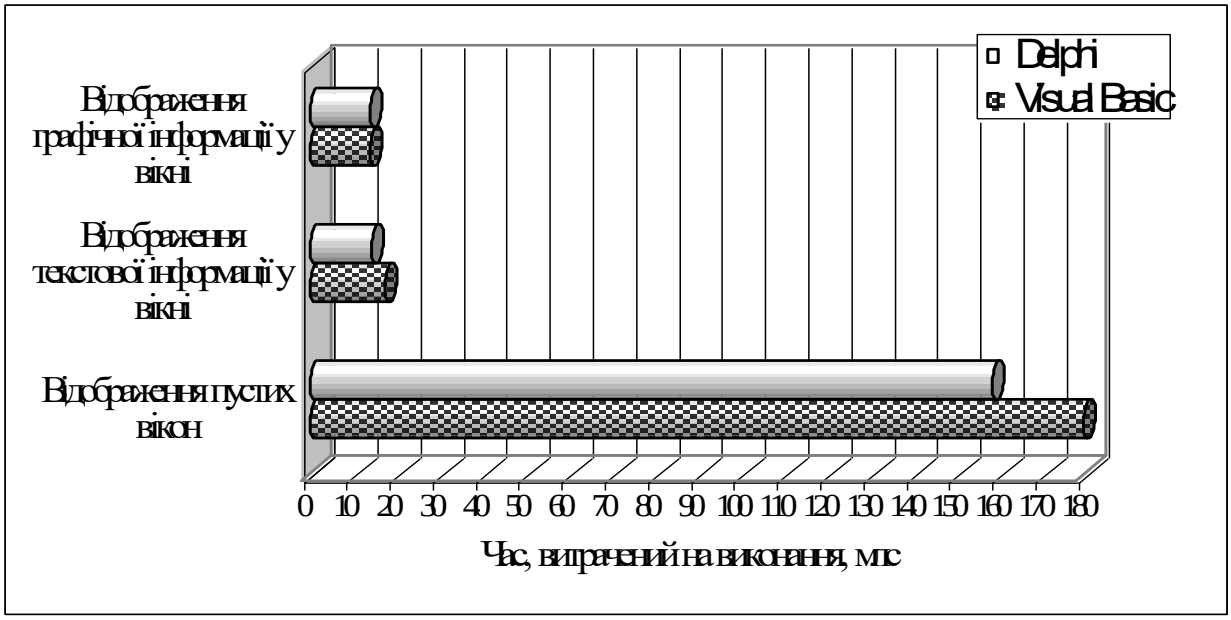


Рис.2.9. Діаграма ефективності роботи з дисплеєм Delphi і Visual Basic.

Delphi – це інтегроване середовище для розробки програм, яке об’єднує в собі засоби роботи з базами даних, електронними таблицями, графічним, текстовим редакторами, містить множину готових компонентів, зокрема для підтримки Web-програмування, тощо. Вивчення перерахованого програмного забезпечення передбачено шкільною програмою з основ інформатики та обчислювальної техніки. Тобто Delphi можна використовувати не тільки для навчання сучасних методів програмування, а й для формування умінь подання і обробки інформації, її пошуку в базі даних.

В основі створення власних проектів засобами системи візуального програмування Delphi лежить комбіноване використання різних компонент–візуальних об’єктів з власними властивостями і методами опрацювання, з яких, як будинок з цеглин, будуються проекти – користувацькі програми.

Кожна група містить в собі кілька різних компонент, призначених для виконання однакових операцій, але різними способами. Наприклад, операція вибору може здійснюватися за допомогою компонент: *ListBox, ComboBox, CheckBox, RadioGroup, Button* тощо. Побудова власного проекту передбачає аналіз властивостей і функцій компонент однієї групи з метою вибору раціональної. Творча невдоволеність стандартними компонентами задовольняється засобами побудови власних компонентів на основі принципів об'єктно-орієнтовного програмування.

Компоненти в Delphi є більш гнучкими порівняно з Visual Basic. В Visual Basic прикладний програміст програмує в середовищі мови бейсік, а компоненти йому надаються в готовому вигляді, написані програмістом-професіоналом на мові C++. Бажання створити власну компоненту вимагає освоєння відповідних деталей C++. В Delphi навпаки, візуальні компоненти створюються на Object Pascal, тобто на тій самій мові, на якій пишеться алгоритмічна частина додатку. З огляду на це візуальні компоненти Delphi, як окремі об'єкти є відкритими для надбудови і переписування.

Створення нової компоненти на основі компонент або класів, вже існуючих в Delphi, передбачає виконання, а, відповідно, і розвиток розумових операцій, характерних для конструкторської діяльності: включення до вже існуючої компоненти нової частини, яка відповідає за виконання нової функції; зміна зовнішнього вигляду компоненти; пристосування компоненти до нових умов застосування (модернізація) тощо.

Зазначається [80, с.65], що для навчання інформатики в школі Delphi цікава не тільки завдяки своїм потужнім засобам створення програмних проектів, а й своєю простотою. Delphi є системою програмування високого рівня. Вона бере на себе значну частину роботи з управління комп'ютером, що дає можливість у простих випадках обходитись без особливих знань про деталі її роботи. Вона пише "за нас" значну частину тексту програми: описання об'єктів, заголовки процедур тощо. Програмісту залишається тільки

вписувати необхідні рядки, що визначають індивідуальну поведінку програми. Але навіть тут Delphi в багатьох випадках сама вказує місце, де необхідно розмісити ці рядки, підказує, які методи необхідно застосувати до об'єктів, що використовуються.

Окрім цього, Delphi дозволяє виконувати конструювання форми без написання програмного коду: розміщувати компоненти на екрані, задавати початкові значення їх властивостей (розміри, колір, вид бордюру тощо). Для цього призначено спеціальне вікно – Інспектор об'єктів, в якому перераховані всі доступні в режимі проектування властивості виділеної компоненти і їх поточні значення. Значення будь-якої властивості при необхідності можна легко змінити, що негайно відобразиться на зовнішньому вигляді об'єкта. Наприклад, можна змінити колір, написи на об'єкті тощо. Тобто до запуску програми на екрані можна побачити, як буде виглядати форма нашого програмного проекту. Візуальне програмування підходить для навчання не тільки тому, що воно сучасне. Візуалізація процесу дозволяє значно скоріше побачити результат своїх зусиль, робить його наочним, і в той же час спирається на достатньо глибокі поняття і уміння учнів. Не останню роль при цьому відіграють емоції і естетичні почуття – прагнення гарно розташувати об'єкти, підібрати їх колір тощо. Помічено, що навіть діти, які не вміють малювати, з цікавістю і задоволенням будують зображення з готових елементів.

На Delphi створені багаточисельні додатки в усіх тих галузях людської діяльності, де застосовуються комп'ютери – від інженерних і наукових розрахунків до автоматизації діяльності управління.

Така потужна система з великими можливостями (Delphi) дозволяє легко реалізовувати навчання на різних рівнях складності:

- робота з візуальними об'єктами практично без програмування;
- використання готових компонент системи при написанні складніших програм;

- створення власних компонент і включення їх в палітру компонент Delphi як стандартних;
- розробка практично корисних закінчених додатків Windows.

Таким чином можна сказати, що Delphi місить в собі передові риси сучасних і перспективних систем програмування, і до того ж має значні переваги в порівнянні з іншими системами для реалізації в навчальному процесі. Доцільно зазначити, що наявність умінь працювати в Delphi буде корисним для майбутньої професійної діяльності учнів. Тому є доцільним вивчення Delphi в межах шкільного курсу інформатики, застосування розроблених компонентів методичної системи формування інтелектуальних учнів як у процесі навчання програмування в Delphi, так і при створенні ними власних програмних проектів.

2.3.2. Вивчення і використання засобів середовища візуального програмування Delphi на основі дослідницького методу

Delphi, як потужна сучасна система програмування, має настільки багато додатків, що вивчити всі її можливості в рамках основного і факультативного курсів інформатики неможливо. “Delphi можна порівняти з горою, шлях до вершини якої далекий і довгий” [89, с.18]. Вчитель не має можливості пояснити функціонування всіх елементів Delphi. Тому викладання правил роботи з Delphi доцільно переорієнтувати на самостійну дослідницьку діяльність учнів з вивчення окремих її компонент, тобто на дослідження, спостереження за “прекрасними квітами і диковинними метеликами на схилах цієї гори”[89, с.18]. Орієнтація на оволодіння учнями дослідницькими методами у процесі вивчення навчального матеріалу дозволяє поєднувати науково-дослідницьку і навчальну діяльності, розділення яких на думку А.М.Боровика [30, с.10] є однією з причин освітянських проблем.

Мотиваційними аспектами стимулювання учнів до вивчення Delphi можуть бути: зручність засобів для реалізації власних проектів, простота

роботи в “інтелектуальному”, дружньому середовищі, історична інформація про створення цього програмного продукту та виникнення самої його назви. Дельфи – це стародавнє грецьке місто на березі Коринфського затоки. Воно пов’язано з ім’ям бога мудрості і покровителя мистецтв Аполлона. Згідно міфу, саме в Дельфи знаходилося святилище Аполлона. Його жриці-сивілли вирікали тим, хто до них звертався, пророцтва – оракули. Розробники Delphi, підбираючи ім’я для свого нового програмного продукту, хотіли відобразити його унікальні можливості в роботі з базами даних, пов’язати його ім’я з таким відомим іменем в цій галузі, як Oracle. Тим самим, Delphi ще називають “дельфійським оракулом”.

Звісно, що дослідницький метод при вивченні Delphi носить імітаційний характер, але його доцільно застосовувати для подальшого формування та удосконалення інтелектуальних умінь. Досліджувані учнями компоненти Delphi не є новими, вони створювалися розробниками цього програмного продукту. Але вони нові для учнів, а відкриття нового відбувається через відповідну розумову діяльність першовідкривачів, яка передбачає виконання комплексу інтелектуальних операцій (спостереження, аналіз, синтез, порівняння, виділення головного, класифікація, узагальнення, передбачення тощо).

Сутність науки вбачається не в простій реєстрації фактів, а в упорядкованому порівнянні та класифікації знань. Знайдені нові деталі порівнюються і включаються в класифікацію вже існуючих знань. Згідно з діями, які виконуються при створенні програмної реалізації проекту, всі компоненти Delphi можна поділити на такі групи (за функціональним призначенням): 1) компоненти для реалізації введення даних (*Edit, Memo, OpenDialog, SpinEdit, MaskEdit* тощо); 2) компоненти для реалізації виведення результатів та повідомлень для користувача (*Edit, Memo, StaticText, Panel, Label, SaveDialog* тощо); 3) компоненти для здійснення управління процесами при роботі програмного проекту (*MainMenu, PopupMenu, Button,*

CheckBox, RadioButton, ListBox, ComboBox, Radiogroup, Timer тощо); 4)компоненти, що застосовуються для відображення стану виконуваної дії (*ScrollBar, TrackBar, ProgressBar, Gauge* тощо); 5) компоненти для естетичного оформлення вікон проекту (*Panel, Bevel, ToolBar, CoolBar, Splitter* тощо); 5) компоненти спеціального призначення (для створення графічних зображень (*Image, Shape, PaintBox* тощо), мультимедійних додатків (*MediaPlayer*), баз даних (*DataSource, Table, Query* тощо), Web-сторінок (HTNL, HTTP тощо), ділової графіки (*Chart, DBChart, QRChart, VtChart, Graph*) тощо). Знаходячи для себе якусь нову компоненту (стандартну або нестандартну), визначивши її основні функції, учень на основі відповідних порівнянь з відомими йому компонентами повинен віднести її до цієї чи іншої групи, тим самим виконавши операцію класифікацію. Окрім цього, користування систематизованим інструментарієм спрощує учню орієнтацію в наборі компонент, виконання операції вибору тієї компоненти, яка на його думку буде більше підходити для реалізації деякого фрагменту програми.

В науці дослідження проводиться згідно з наступними етапами [220]: 1)виявлення незрозумілих фактів; 2) формулювання та уточнення проблеми, збір даних; 3) висування гіпотези; 4) складання плану досліджень; 5)виконання дослідження, перевірка гіпотез; 6) формулювання результату; 7)оцінка можливостей застосування нового знання. Перенесення дій наукового дослідження на процес вивчення окремих компонент Delphi потребує внесення відповідних корективів, після застосування яких дослідження компоненти доцільно проводити згідно з такими етапами: 1)виявлення нової компоненти; 2) виявлення її властивостей, методів, подій, на які реагує компонента, батьківського класу, нащадком якого вона є; 3)висування гіпотези про призначення компоненти і виконувані нею функції; 4) складання плану дослідження, експериментальної перевірки роботи методів і властивостей; 5)проведення експериментів на простих прикладах; 6)віднесення компоненти до відповідної групи в системі компонент згідно з

отриманими результатами дослідження; 7) оцінка можливостей застосування досліджуваної компоненти.

Підгрунтям для проведення дослідження окремих компонент Delphi повинні бути знання учнів основ об'єктно-орієнтованого програмування, об'єктної моделі Delphi, наявність умінь роботи з проектом в середовищі Delphi (створення, завантаження, запис проекту, розміщення компонент на формі, робота з допомогою) та відповідних інтелектуальних умінь, що задіяні при створені об'єктів (розглянуті в 2.1.2).

Виявлення нових компонент здійснюється як в самому середовищі Delphi, так і на спеціальних дисках, на яких поставляються різноманітні додаткові нестандартні компоненти, розроблювані багатьма професійними програмістами. Дослідження додаткової компоненти з диска потребує її попередньої інсталяції до середовища (Component/Install Component). Далі інстальована, розміщена на деякій панелі компонент нестандартна компонента розміщується на формі (вікні) проекту так, як і сусідні стандартні компоненти.

На наступному етапі дослідження компоненти відбувається збір інформації про її методи, властивості, пращури. Джерелами для отримання такої інформації можуть служити: книжки з програмування на Delphi, відповідні сайти в мережі Internet (наприклад, www.borland.com, www.xs4all.nl/~dgb/delpascl.html, www.doit.com/delphi, www.torry.ru), довідкова система Delphi (тільки для стандартних компонент), підказки середовища в процесі програмування, зміст інспектора об'єктів.

Довідкова система Delphi дає повне визначення стандартних компонент: їхнє призначення, перелік властивостей з короткою характеристикою кожної, пе ма сп ий вк ий сп в др

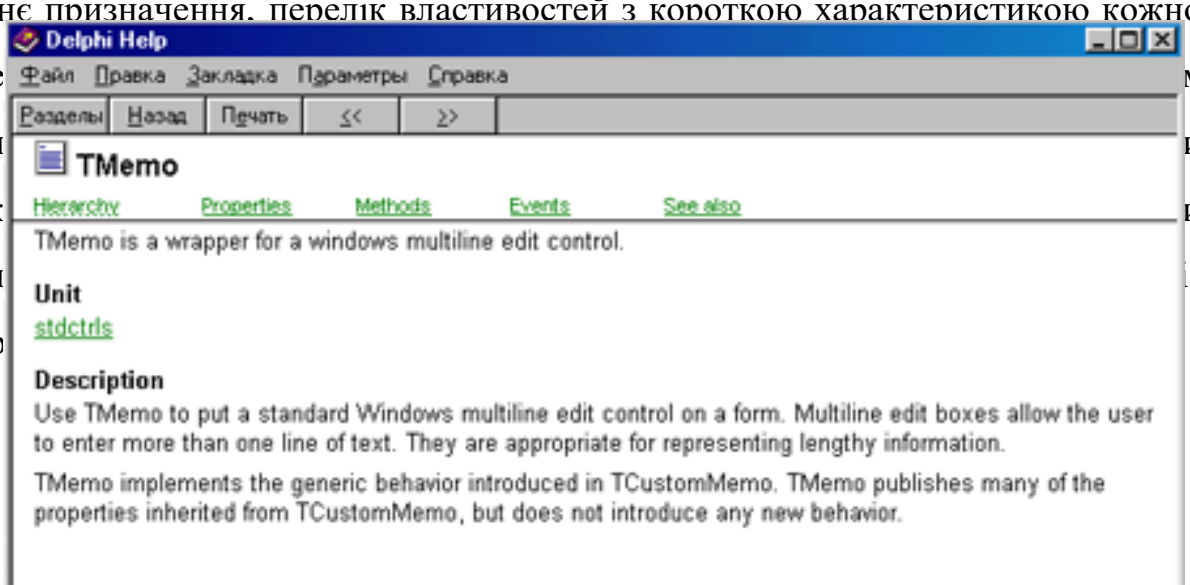
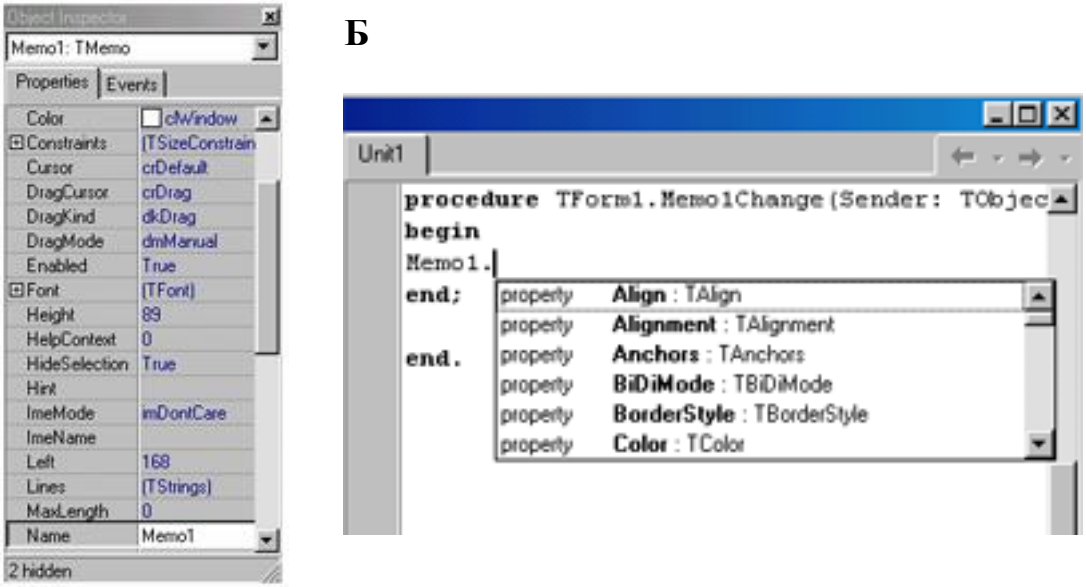


Рис.2.10. Вікно загальної довідкової інформації заданої компоненти.

У вікні загальної довідкової інформації наводиться назва компоненти (в



даному прикладі *TMemo*), ряд покажчиків: Hierarchy (в окремому вікні виводиться фрагмент ієрархічного дерева, в якому є шукана компонента. Вона належить до найнижчого елемента ієрархії і успадковує властивості і методи класів вищого рівня, які також можна передивитися), Properties (виводиться список властивостей, коротку характеристику яких можна передивитися окремо), Methods (виводиться список методів з передбаченою можливістю їх перегляду), Events (список подій, на які реагує компонента), See also (список компонент, споріднених або подібній досліджуваній), Example (якщо є, то виводиться приклад застосування компоненти), загальна інформація про призначення компоненти. Поступовий перегляд довідкової інформації допоможе учням зібрати необхідну інформацію про компоненту.

Інспектор об’єктів містить список подій і властивостей компоненти, доступних у візуальному режимі (рис.2.11. А). Зрозуміло, що на основі цього списку можна частково зробити висновок про атрибути і застосування компоненти, але її методи залишаться для дослідника невидними, що є недоліком, особливо відчутним при дослідженні нестандартних компонент, інформація про які відсутня в довіднику Delphi.

Рис.2.11. А- інспектор об’єктів, Б- приклад роботи підказки.

Побачити не тільки всі властивості, а й методи компоненти дозволяє механізм підказки середовища, який введений починаючи з третьої версії Delphi. Вікно підказки (рис.2.11. Б) з’являється після набору точки після імені компоненти або її властивості, і містить список всіх властивостей та формати методів компоненти, які в даному випадку можна використовувати. Назви властивостей і методів в більшості випадках свідчать про виконувані ними функції, а описання самого методу та його формальних параметрів дозволяє здогадатися про спосіб його використання, про те, що необхідно передавати в середину методу, передбачити результат та поведінку компоненти при його застосуванні.

Подальша експериментальна робота з вивчення функціонування компоненти передбачає складання плану та проведення експериментів на основі її застосування в простих проектах. Наявний список властивостей і методів утворюють своєрідну “схему”, “програму”, що визначає діяльність дослідника. Тактику проведення подальшої експериментальної роботи за аналогією з принципом “розділяй і володій” можна визначити як “кожний раз змінюй тільки що-небудь одне”. Незалежно від того, що є предметом дослідження, повинен бути в наявності тільки один змінний фактор. Тобто в кожному окремому експерименті необхідно змінювати тільки одну властивість або застосовувати один метод, а потім спостерігати за реакцією компоненти на цю зміну.

Виходячи з вибраної тактики дослідження, розробка і реалізація експериментів значно залежить від знаходження аналогій між новими фактами, що спостерігаються, та попереднім досвідом. Планування експерименту доцільно проводити за такими етапами: 1) спостереження факту наявності якогось елемента компоненти та формулювання ідеї його функціонування або поведінки компоненти; 2) зіставлення нового з чимось вже відомим із попереднього досвіду; згідно з методом аналогії, якщо об’єкти

подібні в ряді властивості, то вважається, що вони подібні і в інших; тобто, якщо нова компонента має ті ж самі властивості, що і вже відома, то можна передбачити, що вона буде мати ту ж саму поведінку, а, тим самим, і аналогічний спосіб використання; 3) висунення припущення, що нову компоненту можна застосовувати в тих самих випадках, що і відому компоненту; 4) виявлення відмінностей між новою компонентою та компонентою з минулого досвіду; 5) висунення припущення, що завдяки цій відмінній інформації нова компонента може знайти ширше застосування, може використовуватись в інших випадках. В ході експериментів перевіряється використання компоненти в різних випадках, в ході чого відбувається підтвердження або спростування висунутих гіпотез.

Підгрунтям для умілого виконання учнями операції аналогії є сформованість інтелектуального уміння порівняння. За допомогою аналогії подібність предметів, виявлена в результаті їх порівняння, поширюється на нові властивості (нову властивість). В навчанні, як і в науці, аналогія часто корисна тим, що вона наводить людину на здогадки, тобто є евристичним методом. В навчанні не менш важливим, ніж вчити доказувати і перевіряти, є необхідність вчити догадуватися, що треба доказувати та перевірити і як знайти це доведення і спосіб перевірки, формувати уміння прогнозувати [191, с.288].

Розглянемо приклад застосування описаної методики. Нехай компонента *Edit* (розташована на сторінці *Standart)*, яка призначена для введення та виведення одного текстового рядка, вже відома учням, входить до змісту їх минулого досвіду. Вона характеризується тим, що значення текстового рядка присвоюється її властивості *Text*: *Var S:string; Edit1:TEdit; S:=Edit1.Text*. Компонента має свої власні внутрішні команди редагування: вилучення символу з рядка, виокремлення мишею частини рядка, вставлення літери в середину рядка. Для того, щоб запобігти висвітлюванню у вікні введення рядка *‘Edit1’* в результаті запуску програми, при конструюванні в інспекторі

об’єктів значення властивості *Text* доцільно очистити. При необхідності очищення властивості *Text* у процесі виконання програми до компоненти застосовується метод-процедура *Clear: Edit1.Clear;*. У зв’язку з тим, що тип властивості *Text* – текстовий рядок *string*, при введенні числа необхідно буде в програмному коді виконати перетворення введеного рядка в число:*Var n:integer; N:=StrToInt(Edit1.Text);*. Для дійсних чисел можна використовувати функцію перетворення *StrToFloat(Edit1.Text)*. В разі використання компоненти *Edit* тільки для виведення інформації, її властивості *ReadOnly* присвоюється значення *True*, що не дозволяє користувачу змінювати значення властивості *Text* в процесі виконання програми, але не забороняє його змінювати в коді самої програми. Окрім цього, встановлення властивостей *Enabled* і *TabStop* у *False* забороняє вибір цієї компоненти за допомогою миші або табуляції з клавіатури при роботі програми.

Завдання для учнів стосується дослідження аналогічних компонент *MaskEdit (*розташована на сторінці *Additional)* та *SpinEdit (*сторінка *Samplas).* Про можливості застосування цих компонент будуть свідчити: їх назва (ім’я), зовнішній вигляд, наявні властивості і методи. Описання властивостей і методів компоненти можна або скориставшись допомогою (Help), або довідниковою літературою. Розмістивши компоненту на формі, використати в простому проекті з метою її тестування. Задавши вхідне дане окремій властивості, прослідкувати за роботою програми і зробити відповідні висновки про роль цієї властивості, а потім і про функціонування всієї компоненти в цілому. Застосувати досліджену компоненту в тому ж самому проекті, де застосовувалася компонента з минулого досвіду, намітити інші випадки її використання.

В ході проведеного дослідження стосовно компоненти *MaskEdit* учні повинні з’ясувати, що порівняно з компонентою *Edit* поряд з однаковими властивостями (*Text, ReadOnly, Enabled, TabStop* тощо) і методами (*Clear, CopyToClipbord, SelectAll, Undo* тощо) досліджувана компонента має

властивість *EditMask,* яка дозволяє задати маску (формат) для подання даних. Введення даних формату, що відрізняється від заданого, забороняється. При виборі цієї властивості завантажується редактор масок *MaskEditor*, в якому можна вибрати необхідний формат даних зі стандартного списку, наприклад: Phone-(044)555-12-12, Date-06.27.99, Long Time-09:05:15PM, Shot Time-13:45, або скориставшись клавішею завантажити один з наборів нестандартних форматів. Послуга редактора *Character for Blanks* дозволяє задати символ для маски, наприклад, "#". Але, як і в компоненті *Edit* властивість *Text* має тип *string*. Тому, наприклад, для використання числових даних з формату дати необхідно застосовувати функції вирізання фрагменту тексту і перетворення рядка в число. В проекті обчислення за введеною датою кількості днів, що залишилося до певної події (наприклад, дня народження) замість компоненти *Edit* може бути застосована досліджувана компонента *MaskEdit*.

Передбачається, що в результаті проведеного дослідження стосовно компоненти *SpinEdit* учні знайдуть ті ж самі властивості і методи, що і в компоненті *Edit,* та будуть мати наступні відомості: її властивості *MaxValue* та *MinValue* задають найбільше і найменше значення числа, а властивість *Increment* задає крок нарощування значення при натисненні відповідної з двох кнопок, не забороняється введення цілого числа без використання кнопок. На відміну від компоненти *Edit* компонента *SpinEdit* призначена для введення числової інформації. Тому вона не має властивості *Text*, а дані присвоюються властивості цілого типу *Value*. З огляду на сказане, компоненти типу *SpinEdit* зручніші для застосування в проекті створення калькулятора для знаходження суми двох цілих чисел, ніж компоненти типу *Edit*.

Подальше використання компонент *MaskEdit* та *SpinEdit* можна пов'язати із створенням форм заповнення баз даних, що надаватиме можливість запобігання помилок, зменшення їх кількості при наборі інформації з клавіатури. Так, компонента *MaskEdit* не дасть набрати більше цифр в номері телефону, ніж передбачено форматом маски, а компонента *SpinEdit* не дасть

набрати значення ваги людини, яка б виходила за межі допустимого, яке задається властивостями *MaxValue* та *MinValue*.

Досвід дослідження функціонування наданих компонент стає підгрунтям для творчої діяльності з їх створення. “Компонента (Component) – це фізична сутність системи, що реалізує деякій набір інтерфейсів” [36, с.311]. Окремий клас, що створюється в процесі об’єктно-орієнтованого програмування, становить логічну сутність системи, а елементи його структури (атрибути, властивості, методи) розподіляються на відкриту (public), закриту (private), захищену (protected) частини. Властивості і методи, що належать до відкритої частини, доступні для об’єктів інших класів в процесі їх взаємодії і становлять інтерфейс класу. Фізична модель системи (компоненти) має ту ж саму структуру, що і логічна модель, в якій елементи інтерфейсу (властивості та події) описуються в блоці published і є доступними не тільки для інших об’єктів, а й програм, наприклад, для інспектора об’єктів (Object Inspector). Структурна спорідненість компоненти з об’єктною моделлю Object Pascal забезпечує можливість для застосування загальних інтелектуальних умінь, сформованих в процесі вивчення основ об’єктно-орієнтованого програмування, в новій, але аналогічній, ситуації створення власних компонент.

З огляду на вище сказане, доцільним є застосування учнями при створенні власних компонент тих самих правил-орієнтирів виконання розумових операцій аналізу-синтезу, абстрагування, порівняння, узагальнення, що описані в 2.1.2 і використовувалися при створенні об’єктів в процесі об’єктно-орієнтованого програмування та формування загальних інтелектуальних умінь. Нові умови застосування зазначених правил-орієнтирів вимагають внесення до них певних корективів. Так, в правилі-орієнтирі розумових дій аналізу-синтезу, абстрагування разом з виділенням суттєвих з точки зору функціонування компоненти характеристик необхідно визначити, які характеристики (властивості, події) будуть становити інтерфейс

компоненти, тобто будуть доступні зовнішньому користувачу в інспекторі об’єктів. Результатом виконання операції узагальнення буде визначення батьківського класу (стандартні класи *TComponent, TGraphicControl, TWinControl, TCustomControl* тощо, класи, що відповідають проінстальованим в середовищі стандартним і нестандартним компонентам), нащадком якого повинна бути нова компонента згідно з виконуваними нею функціями і типом компоненти. Так, невізуальним компонентам достатньо бути нащадком від класу *TComponent* (батьківського класу для всіх компонентних класів), візуальні компоненти з деяким графічним зображенням (наприклад, кнопка у вигляді довільної фігури) повинні бути нащадками від компонентних класів (наприклад, *TGraphicControl*, *TWinControl*, *TCustomControl),* що мають властивість Canvas та метод Paint для малювання зображення цієї візуальної компоненти. Вибір батьківського класу для створюваної компоненти здійснюється у вікні, що викликається через пункт меню Component/New Component, в рядку Ancestor type.

Широка різноманітність наявних компонентних класів дозволяє реалізовувати методику створення однакових компонент на основі різних батьківських класів, що передбачає проведення дослідження кількох компонентних класів, виявлення їх загальних та відмінних характеристик, застосування різних механізмів, що пропонуються різними батьківськими класами для реалізації поведінки створюваної компоненти. Так, в [140, с.9-10] наводиться приклад створення візуальної управляючої компоненти, яка має вигляд круга, що змінює свій колір в результаті натиснення на ній лівою клавішею миші. Цю компоненту пропонується реалізувати двома способами: 1) у вигляді круглої кнопки (*Button*) як нащадка класу *TGraphicControl*; 2) у вигляді компоненти *Check* як нащадка класу *TCustomControl*. Обидва батьківських класів мають властивість *Canvas* і метод *Paint*, тобто малювання візуального зображення компоненти в різних її реалізаціях здійснюється однаково. Проблемна ситуація виникає при реалізації механізму управління,

тобто зміні стану компоненти при натисненні на компоненті лівою клавішею миші. Успішне вирішення цієї проблеми залежить від рівня сформованості умінь аналізувати, порівнювати, проводити аналогію між створюваною компонентою і стандартною, подібною до неї. Так, для реалізації управління в структурі нової компоненти типу кнопки (*Button*) достатньо мати атрибут стану натиснення (*FDown*) логічного типу *Boolean*, значення якого змінюється на протилежне в процедурі обробки події *OnClick* разом з перемальовуванням візуального зображення кнопки. Клас *TCustomControl* є нащадком від класу *TWinControl*, тобто успадковує від нього ширші можливості реалізації функцій управління Windows. Так, для реалізації механізму управління новою компонентою типу Check надається можливість мати атрибут стану *FState* і відповідну властивість *State* типу *TChackBoxState*, які можуть набувати тільки два значення *cbChecked* (помічений) та *cbUnChecked* (непомічений), як і в стандартній компоненті *CheckBox* (сторінка Standard). Зміна значення стану компоненти і перемальовування її зображення також здійснюється в процедурі обробки події *OnClick*.

Зрозуміло, що творча діяльність із створення власних компонент вимагає від учнів сформованості на високому рівні загальних інтелектуальних умінь, задіяних в ТООП. Але Delphi надає можливість створювати нові компоненти на більш низькому конструкторському рівні. На відміну від творчості, яка пов’язується із створенням нового, невідомого, конструювання грунтується на комбінуванні старого і відомого, тобто вже існуючих компонент. Необхідні компоненти (складові нової) розміщуються на формі, пункт меню Comonent/Create Component Template дозволяє зберегти створену комбінацію компонентів у вигляді окремої компоненти і розмістити на деякій сторінці панелі компонентів.

Не менш важливим для створення компоненти є етап її тестування, аналіз функціонування компоненти в процесі виконання проекту, що її використовує. Тестування – процес деструктивний, зворотний

конструктивному, створюючому. Як свідчать спостереження експериментальної роботи, учні більше схильні до конструктивного процесу створення об’єктів, ніж до деструктивного процесу їх тестування. Підбір відповідних тестів для аналізу правильності і ефективності функціонування компоненти сприяє розвитку критичності мислення учнів, формуванню умінню прогнозувати, тобто є таким же важливим підгрунтям для їх інтелектуального розвитку, як і процес створення власних компонент. В [148] рекомендується тести неправильних та непередбачених вхідних даних розробляти так само ретельно, як і для правильних та передбачених. Необхідно перевіряти не тільки, чи робить компонента те, для чого вона призначена, але й чи не робить вона те, що не повинна робити. Доцільно використовувати аналіз межових значень вхідних та вихідних змінних, метод припущення про помилку. Наявність уміння передбачати можливі помилки спрощує процес розробки тестів для їх виявлення та застосування механізмів захисту програмного коду від переривання програми у випадку виникнення помилок (*try ... except*).

З огляду на вище сказане, можна зазначити, що процес створення власних компонент, а в загалі об’єктів, містить в собі елементи і науки і мистецтва. “Розробка нових структур передбачає і політ фантазії, і синтез досвіду і знань: все, що необхідно художникові для реалізації свого задуму на полотні або папері. Після того, як цей задум визрів в голові програміста-художника, він обов’язково повинен бути проаналізований з точки зору застосування даного наукового методу програмістом-вченим зі всією ретельністю, що притаманна справжньому вченому” [292].

Уміння досліджувати функціонування існуючих компонент, створювати нові компоненти можна розглядати як уміння здобувати нові знання, що завжди вважалося основою для зростання професіоналізму, безперервної освіти людини в майбутньому, її загальної культури, а відповідно і її інтелекту. Отже, поряд з методом проектів дослідницький метод і метод

аналогії також можна використовувати як непрямі методи формування інтелектуальних умінь учнів.

2.4. Організація і аналіз результатів педагогічного експерименту

З метою визначення ефективності запропонованих компонент методичної системи навчання інформатики проводився педагогічний експеримент, який проходив в три етапи**:**

- констатуючий (1998 – 1999 н.р.);
- пошуковий (1999 – 2000 н.р.);
- формуючий (2000 – 2001 н.р.).

В процесі експериментальної роботи розв’язувались такі завдання:

- виявити рівень сформованості загальних інтелектуальних умінь старшокласників у процесі навчання інформатики;
- проаналізувати та виявити типи програмного забезпечення програмістського призначення для формування інтелектуальних умінь учнів у процесі навчання інформатики;
- уточнити шляхи та методичні прийоми формування інтелектуальних умінь учнів у процесі навчання інформатики;
- перевірити ефективність запропонованої методики формування інтелектуальних умінь учнів у процесі навчання інформатики.

Експериментальною базою були загальноосвітні школи м. Чернігова №8, 9, 12, гімназія гуманітарно-естетичного профілю №31, колегіум №11 м.Чернігова, Чернігівський обласний педагогічний ліцей**.**

Основними діагностичними методами були: психолого-педагогічне спостереження, індивідуальні бесіди з учнями, психологічна діагностика рівнів сформованості загальних інтелектуальних умінь учнів за відповідними психологічними методиками [5, 258-260], вивчення продуктів діяльності учнів, перевірка знань. В ході спостережень фіксувалися різні показники процесу навчання: час, витрачений вчителем на введення прийому розумової діяльності, закріплення прийому в ході навчання основних положень ТООП та візуального програмування; зміст і характер питань, які задавали учні на етапі

вивчення нового матеріалу або його закріплення; час, витрачений учнями на самостійне виконання завдань; ступінь інтелектуальної активності учнів. “Якщо мислення - це процес розв’язування задач, то інтелектуальна активність – це нестимульоване зовні продовження мислення” [26, с.24]. Ефективність і доступність розроблених компонентів методичної системи визначалась аналізом проведених уроків, з’ясовуванням причин утруднень у школярів при виконанні завдань з ТООП, розв’язуванні задач-проектів.

В ході проведення констатуючого експерименту за відповідними психологічними методиками (додаток З), кожна з яких містила по 20 завдань, для учнів визначався рівень сформованості умінь виконувати певні інтелектуальні операції: 1) аналогію (тест N1); 2) виділення істотних ознак (тест N3), що входить до складу операції абстрагування, яка, як зазначалося в 1.2.3, передбачає виділення суттєвих для спостерігача характеристик деякого об’єкта, що відрізняють його від всіх інших видів об’єктів, випускаючи ті характеристики, які на даний момент несуттєві; 3) узагальнення і класифікація понять (тест N5); 4) порівняння (порівняння двох пар термінів: перша пара - терміни відносяться до однієї категорії, друга пара - терміни відносяться до різних категорій); 5) знаходження закономірності в числовому ряді; 6) розглядання об’єкта з різних точок зору - гнучкість мислення (написати способи використання деякого предмета). Результати тестування визначалися за кількістю набраних балів з 20 (для порівняння та гнучкості – необмежена кількість балів).

За аналогією зі стандартними критеріями оцінювання навчальних досягнень учнів та методикою оцінювання умінь учнів (Л.В.Мар’яненко), описаної в 1.1.3., для оцінювання сформованості зазначених інтелектуальних умінь встановимо чотири рангові градації:

1. Початковий (уміння не сформоване) – 0–6 (порівняння <=4, гнучкість 0-1).
2. Середній (уміння помітне в діяльності учня, але проявляється не дуже якісно) – 7 – 14 (порівняння 5-10, гнучкість 2-4).

3. Достатній (уміння сформоване, але його можна вдосконалювати) – 15 – 18 (порівняння 11-16, гнучкість 5-8).
4. Високий (уміння сформоване і ефективне) – 19 – 20 (порівняння >17, гнучкість >= 9).

В ході проведеного дослідження були отримані результати, наведені в таблиці 2.6**.** Таблиця 2.6.

Розподіл учнів за рівнями сформованості загальних інтелектуальних умінь.

Інтелектуальна операція або якість мислення	Початковий рівень		Середній рівень		Достатній рівень		Високий рівень	
	К	%	К	%	К	%	К	%
Аналогія	26	12,5	124	59,6	52	25	6	2,9
Виділення істотних ознак	11	5,3	103	49,5	86	41,3	8	3,8
Узагальнення і класифікація понять	25	12	149	71,6	29	13,9	5	2,4
Порівняння	9	4,3	69	33,2	86	41,3	44	21,2
Знаходження закономірності в числовому ряді	4	1,9	85	41	105	50,5	14	6,7
Гнучкість мислення	33	15,8	96	46,2	69	33,2	10	4,9

Умовні позначення до таблиці 2.6:

К – кількість учнів, що набрали відповідну кількість балів,

% - відсоток учнів з даним рівнем сформованості інтелектуального уміння.

Як свідчать дані констатуючого експерименту, наведені в таблиці 2.6 та рис. 2.12, особливої уваги щодо формування потребували уміння виконувати інтелектуальні операції аналогію та класифікацію понять. Ці уміння набрали найбільший відсоток (59,6%, 71,6%) на середньому рівні і найменші відсотки на достатньому рівні (25%, 13,9%). Наступними за рангом необхідності розвитку є гнучкість мислення і формування уміння виділяти істотні ознаки. Високий рівень сформованості умінь порівнювати і знаходити закономірності пояснюється наявністю в учнів значного попереднього досвіду з виконання цих операцій при вивченні інших навчальних предметів та інших тем з

інформатики. Наприклад, при вивченні операторів циклу учням доводилося виконувати завдання на побудову загальної формули для числового ряду.

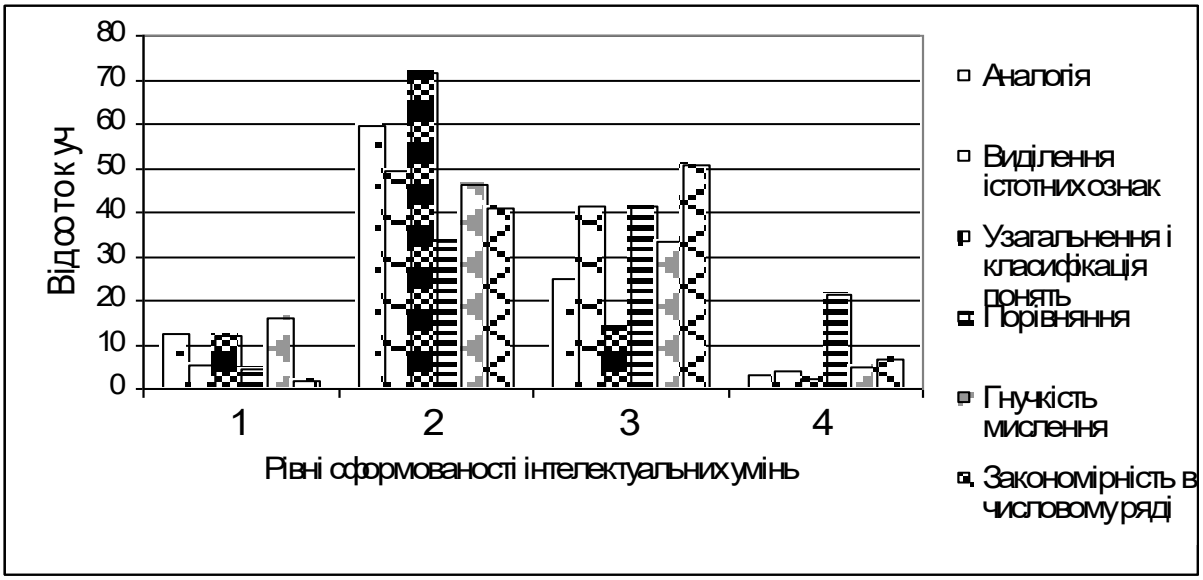


Рис.2.12. Діаграма розподілу учнів за рівнями сформованості умінь виконувати відповідні інтелектуальні операції.

Отримані в ході констатуючого експерименту дані спонукали до вибору ТООП та дослідницького методу та методу аналогії як засобів формування загальних інтелектуальних умінь: узагальнення і класифікації понять (при побудові ієрархій об’єктних типів або класів), виділення істотних ознак (при побудові абстракцій (об’єктних типів або класів)), а також як засобів розвитку гнучкості мислення (погляд на об’єкт з різних точок зору при дослідженні його функціонування), формування уміння проводити аналогію між явищами, об’єктами при дослідженні предметної галузі. Уміння порівнювати і знаходити закономірності становлять основу як для операції аналогії, так і для класифікації і виділення суттєвих ознак. За даними дослідження І.Д.Пасічника [184] операція порівняння позитивно впливає на синтез, узагальнення, конкретизацію, між ними виявлена значна кореляція (+0.24 - +0.47). Між класифікацією і порівнянням спостерігається ще тісніший зв’язок, кореляція між ними становить +0.81. Тому вище значення достатнього рівня їх сформованості є підгрунтям для успішного формування інших зазначених загальних інтелектуальних умінь, пов’язаних з ними.

На цьому етапі експерименту також проводився пошук і методичний аналіз різноманітних програмних засобів реалізації технології об’єктно-орієнтованого програмування та проектування.

В ході пошукового етапу експерименту (1999-2000 н.р.) уточнювались шляхи і методичні прийоми формування інтелектуальних умінь учнів, проводився цілеспрямований пошук та добір нового змісту навчання шкільного курсу інформатики з метою підвищення практичної значущості результатів навчання і як наслідок підвищення мотивації і прийняття мети навчання. На цьому етапі також досліджувалися можливості використання в навчальному процесі об’єктно-орієнтованої системи візуального програмування Delphi, розроблялися різні види завдань, тематика проектів, призначених для розв’язання засобами візуального програмування.

На перших двох етапах експерименту були отримані такі результати:

- недостатній рівень сформованості загальних інтелектуальних умінь;
- підтверджені висновки теоретичного аналізу про можливість використання технологій об’єктно-орієнтованого та візуального програмування як засобів формування інтелектуальних умінь учнів;
- підібрані програмні засоби для проведення формуючого експерименту;
- розроблена методична система навчання об’єктно-орієнтованого програмування і проектування та візуального програмування.

В ході пошукового і констатуючого етапів експерименту були створені необхідні передумови для проведення формуючого експерименту.

Мета формуючого етапу експерименту (2000-2001 н.р.) полягала в перевірці ефективності запропонованої методики формування інтелектуальних умінь учнів засобами об’єктно-орієнтованого програмування у процесі навчання інформатики. Формуючий експеримент проводився в звичайних умовах педагогічного процесу. До експерименту було включено 7 класів (208 особи)– контрольна група (КГ) і 7 класів (208 осіб) – експериментальна група (ЕГ). Всього в експерименті брало участь 416 учнів.

Навчання в експериментальних класах відрізнялося від навчання в контрольних класах вивченням і застосуванням засобів об’єктно-орієнтованого програмування. Всі інші фактори, які впливали на процес навчальної діяльності учнів ми намагалися вирівняти: при відборі експериментальних класів і шкіл враховувався тип шкіл, їх територіальне розміщення, оснащеність класів сучасною комп’ютерною технікою, заняття по можливості проводилися одними і тими ж вчителями.

Оцінка експериментального навчання була проведена на основі:

1) визначення рівнів сформованості загальних інтелектуальних умінь за відповідними психолого-діагностичними методиками;
2) визначення рівня інтелектуальної активності учнів при розв’язуванні конкретних задач за допомогою ПЕОМ.

За рекомендацією психологів визначення рівнів сформованості різних інтелектуальних умінь учнів на етапі формуючого експерименту проводилося за однаковими з констатуючим етапом методиками (виділення істотних ознак) або за аналогічними: аналогія – тест N2, узагальнення і класифікація понять – тест N5, порівняння – порівняти дві пари термінів, гнучкість мислення – написати способи використання іншого предмета. Результати формуючого та констатуючого експериментів представлені в таблиці 2.7 та рис.2.13-2.17.

Умовні позначення до таблиці 2.7:

К – кількість набраних балів, З – загальна кількість балів на одному рівні сформованості інтелектуального уміння, % - відсоток учнів.

Таблиця 2.7.

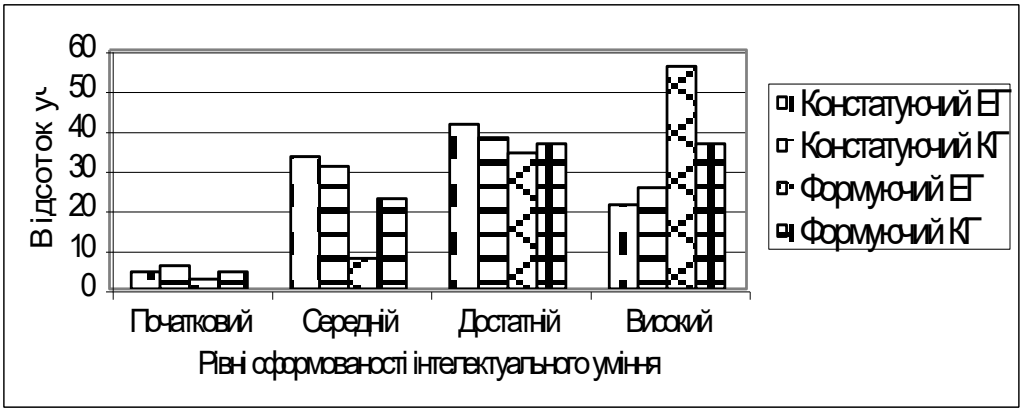
Розподіл учнів за рівнями сформованості загальних інтелектуальних умінь.

Інтелектуальна операція або якість мислення		Рівні сформованості інтелектуальних умінь		Кількість і відсоток учнів, які набрали дану кількість балів																	
				Констатуючий експеримент						Формуючий експеримент						Зміни					
				ЕГ			КГ			ЕГ			КГ			ЕГ			КГ		
				К	З	%	К	З	%	К	З	%	К	З	%	К	З	%	К	З	%
Аналогія	1	Початковий	0-6	26	26	12,5	21	21	10,1	8	8	3,8	13	13	6,3	-18	-18	-8,7	-8	-8	-3,8
	2	Середній	7-10	51	124	59,6	65	136	65,4	26	69	33,2	58	123	59,1	-25	-55	-26,4	-7	-13	-6,3
	3		11-14	73			71			43			65			-30			-6		
	4	Достатній	15-16	36	52	25	32	44	21,2	51	105	50,5	48	63	30,3	15	53	25,5	16	19	9,1
	5		17-18	16			12			54			15			38			3		
	6	Високий	19-20	6	6	2,9	7	7	3,4	26	26	12,5	9	9	4,3	20	20	9,6	2	2	1,0
Виділення істотних ознак	1	Початковий	0-6	11	11	5,3	7	7	3,4	7	7	3,4	6	6	2,9	-4	-4	-1,9	-1	-1	-0,5
	2	Середній	7-10	16	103	49,5	15	99	47,6	14	38	18,3	14	85	40,9	-2	-65	-31,3	-1	-14	-6,7
	3		11-14	87			84			24			71			-63			-13		
	4	Достатній	15-16	71	86	41,3	73	92	44,2	61	137	65,9	76	105	50,5	-10	51	24,5	3	13	6,3
	5		17-18	15			19			76			29			61			10		
	6	Високий	19-20	8	8	3,8	10	10	4,8	26	26	12,5	12	12	5,8	18	18	8,7	2	2	1,0
Класифікація понять	1	Початковий	0-6	25	25	12	30	30	14,4	8	8	3,8	23	23	11,1	-17	-17	-8,2	-7	-7	-3,4
	2	Середній	7-10	125	149	71,6	109	143	68,8	38	80	38,5	92	137	65,9	-87	-69	-33,2	-17	-6	-2,9
	3		11-14	24			34			42			45			18			11		
	4	Достатній	15-16	16	29	13,9	20	29	13,9	76	101	48,6	26	39	18,8	60	72	34,6	6	10	4,8
	5		17-18	13			9			25			13			12			4		
	6	Високий	19-20	5	5	2,4	6	6	2,9	19	19	9,1	9	9	4,3	14	14	6,7	3	3	1,4
Порівняння	1	Початковий	0-4	9	9	4,3	12	12	5,8	5	5	2,4	9	9	4,3	-4	-4	-1,9	-3	-3	-1,4
	2	Середній	5-7	18	69	33,2	29	64	30,8	7	16	7,7	19	47	22,6	-11	-53	-25,5	-10	-17	-8,2
	3		8-10	51			35			9			28			-42			-7		
	4	Достатній	11-13	41	86	41,3	38	79	38	14	71	34,1	31	76	36,5	-27	-15	-7,2	-7	-3	-1,4
	5		14-16	45			41			57			45			12			4		
	6	Високий	17-	44	44	21,2	53	53	25,5	116	116	55,8	76	76	36,5	72	72	34,6	23	23	11,1
Гнучкість мислення	1	Початковий	1	33	33	15,9	39	39	18,8	10	10	4,8	27	27	13,0	-23	-23	-11,1	-12	-12	-5,8
	2	Середній	2-3	42	96	46,2	49	106	51	17	63	30,3	36	98	47,1	-25	-33	-15,9	-13	-8	-3,8
	3		4	54			57			46			62			-8			5		
	4	Достатній	5-6	42	69	33,2	41	54	26	74	112	53,8	48	67	32,2	32	43	20,7	7	13	6,3
	5		7-8	27			13			38			19			11			6		
	6	Високий	9-	10	10	4,8	9	9	4,3	23	23	11,1	16	16	7,7	13	13	6,3	7	7	3,4

Рис.2.13. Діаграма розподілу учнів за рівнями сформованості уміння виконувати аналогію.

Рис.2.14. Діаграма розподілу учнів за рівнями сформованості уміння виділяти істотні ознаки.

Рис.2.15. Діаграми розподілу учнів за рівнями сформованості уміння класифікувати об’єкти.



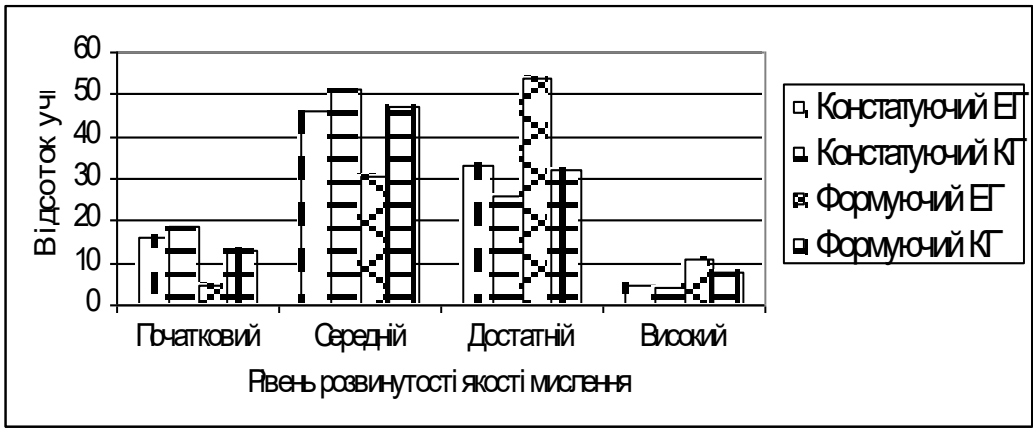


Рис.2.16. Діаграми розподілу учнів за рівнями сформованості уміння порівнювати.

Рис.2.17. Діаграми розподілу учнів за рівнями розвитку гнучкості мислення.

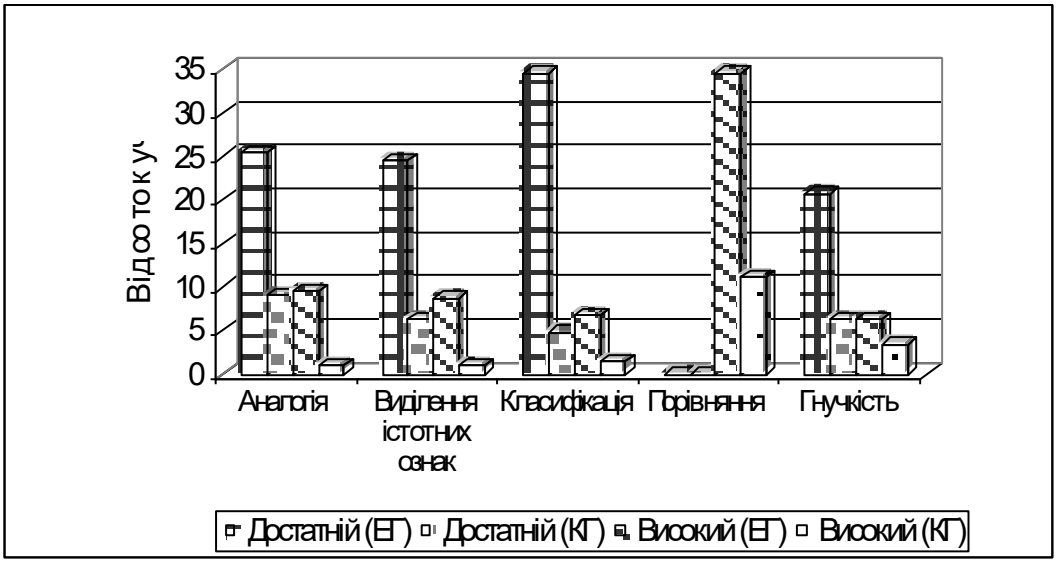


Рис.2.18 Збільшення кількості учнів на достатньому і високому рівнях сформованості інтелектуальних умінь.

Згідно з отриманими даними, порівняно з учнями контрольної групи в учнів експериментальної групи спостерігалося збільшення кількості учнів з достатнім і високим рівнем розвитку інтелектуальних умінь (рис.2.18): аналогія - 16,4%, 8,6% відповідно, виділення істотних ознак - 18,2%, 7,7%, класифікація - 29,8%, 5,3%, порівняння - 23,5% (високий рівень), гнучкість мислення 14,4%, 2,9%; зменшення кількості учнів з середнім рівнем сформованості інтелектуальних умінь: аналогія – 20,1 %, виділення істотних ознак – 24,5%, класифікація - 30,3%, порівняння – 17,3%, гнучкість мислення– 12,1%. У зв'язку з наявністю малої кількості осіб, що мають початковий рівень сформованості інтелектуальних умінь, в цій групі спостерігалися незначні в процентному відношенні зміни.

Оцінка ефективності запропонованої методики проводилася в таких напрямках:

1. Виявлення статистично значущих відмінностей в рівнях сформованості для окремих інтелектуальних умінь учнів експериментальних і контрольних класів.
2. Виявлення впливу запропонованої методики на рівні сформованості інтелектуальних умінь.

Для виявлення статистично значущих відмінностей в рівнях сформованості окремих інтелектуальних умінь учнів контрольної і експериментальної вибірок нами використано метод перевірки статистичних гіпотез.

Подані в таблиці 2.7. результати експерименту використовувалися нами для перевірки нульової і альтернативної гіпотез за критерієм Пірсона (χ^2), оскільки виконані всі відповідні для цього умови:

1) обидві вибірки випадкові;
2) вибірки незалежні і члени кожної з вибірок незалежні між собою;
3) шкала вимірів для кожного уміння є шкалою найменувань з 6 категоріями.

Сформулюємо основну H_0 та альтернативну H_1 гіпотези. Нульова гіпотеза: ймовірність попадання учнів контрольної і експериментальної вибірки (n_1=208, n_2=208) в кожну з i (i=1,2,3,…,6) категорій для окремого уміння рівні, тобто H_0: $P_{1i}=P_{2i}$ і високий рівень сформованості інтелектуального уміння в ЕГ пояснюється випадковими факторами.

Альтернативна гіпотеза H_1: $P_{1i}\neq P_{2i}$ хоча б для однієї з i=6 категорій, тобто цей вищий рівень пояснюється результатом запропонованої методики.

Скориставшись двостороннім критерієм χ^2 (критерієм Пірсона) [68, с.96-106], враховуючи C=6 – кількість категорій, для перевірки гіпотези знаходимо значення T досліджуваної випадкової величини:

$$T = \frac{1}{n_1 \cdot n_2} \sum_{i=1}^{C} \frac{\left(n_1 \cdot O_{2i} - n_2 \cdot O_{1i}\right)^2}{O_{1i} + O_{2i}} \quad (2.1),$$

де O_{1i} і O_{2i} (i=1,2,3, …) – число учнів експериментальних і контрольних класів відповідно, які попали в i-ту категорію.

За формулою (2.1) підраховувалося значення статистики критерію Т для експериментальної і контрольної вибірок до і після формуючого експерименту та для контрольної вибірки до експерименту і після його проведення. Результати обчислень представлені в таблиці 2.8.

За таблицею Г [68, с.130] для числа ступенів вільності ν=C-1=6-1=5 [68, с.101] і рівня значущості α=0,05 знаходимо критичне значення величини Т: $T_{кр}$=11,07.

Згідно з даними таблиці 2.8 для експериментальної і контрольної вибірок до експерименту ((1)-(2)) значення статистики критерію Т менше критичного значення для наведених інтелектуальних умінь, що є підставою для прийняття нульової гіпотези. Тобто до експерименту вибірки не мають статистично значущих відмінностей при 5% рівні значущості.

Таблиця 2.8.

Значення статистики критерію Т для окремих умінь.

Інтелектуальна операція або якість мислення	Кількість набраних учнями балів				Значення статистики критерію Т		
	Констатуючий		Формуючий		(3)-(4)	(1)-(2)	(2)-(4)
	ЕГ (1)	КГ (2)	ЕГ (3)	КГ(4)			
Аналогія	26	21	8	13	60,35	3,92	7,75
	51	65	26	58			
	73	71	43	65			
	36	32	51	48			
	16	12	54	15			
	6	7	26	9			
Виділення істотних ознак	11	7	7	6	63,9	2,12	4,32
	16	15	14	14			
	87	84	24	71			
	71	73	61	76			
	15	19	76	29			
	8	10	26	12			
Класифікація	25	30	8	23	77,12	5,67	7,35
	125	109	38	92			
	24	34	42	45			
	16	20	76	26			
	13	9	25	13			
	5	6	19	9			
Порівняння	9	12	5	9	39,35	8,9	10,14
	18	29	7	19			
	51	35	9	28			
	41	38	14	31			
	45	41	57	45			
	44	53	116	76			
Гнучкість мислення	33	39	10	27	37,67	7,61	9,81
	42	49	17	36			
	54	57	46	62			
	42	41	74	48			
	27	13	38	19			
	10	9	23	16			

Контрольна вибірка до експерименту і після його проведення також не має статистично значущих відмінностей ($T < T_{кр}$, таблиця 2.8. (2)-(4)). Це означає, що зміни, які відбулися в контрольній вибірці за час експерименту зумовлені природним розвитком інтелекту учнів і не є істотними.

Експериментальна і контрольна вибірки після проведення експерименту мають статистично значущі відмінності, оскільки $T > T_{кр}$ (таблиця 2.8. (3)-(4)). Це дає підставу відхилення нульової гіпотези і прийняття альтернативної.

У ході формуючого експерименту в експериментальних класах застосовувались компоненти методичної системи на основі засобів

об’єктно-орієнтованого програмування, спрямовані на формування інтелектуальних умінь в процесі навчання інформатики. Для виявлення впливу запропонованої методики на рівень сформованості інтелектуальних умінь застосуємо критерій Колмогорова-Смирнова, оскільки виконані для цього умови:

1) обидві вибірки випадкові;
2) вибірки незалежні і члени кожної з вибірок незалежні між собою;
3) шкала вимірювань не нижче порядкової [68].

Таблиця 2.9.

Результати обробки експериментальних даних для застосування критерію Колмогорова-Смирнова.

	Шкала балів	Абсолютна частота		Накопичена частота		Відносна накопичена частота		S2-S1	S1-S2	T1	T2	T3
		ЕГ	КГ	ЕГ	КГ	ЕГ(S1)	КГ(S2)					
Аналогія	0 – 6	8	13	8	13	0,038462	0,0625	0,024038	-0,02885	0,284	0	0,284
	7-10	26	58	34	71	0,163462	0,341346	0,177885	-0,17788			
	11-14	43	65	77	136	0,370192	0,653846	0,283654	-0,28365			
	15-16	51	48	128	184	0,615385	0,884615	0,269231	-0,26923			
	17-18	54	15	182	199	0,875	0,956731	0,081731	-0,08173			
	19-20	26	9	208	208	1	1	0	0			
Виділення істотних ознак	0 – 6	7	6	7	6	0,033654	0,028846	-0,00481	0,004808	0,293	0,005	0,293
	7-10	14	14	21	20	0,100962	0,096154	-0,00481	0,004808			
	11-14	24	71	45	91	0,216346	0,4375	0,221154	-0,22115			
	15-16	61	76	106	167	0,509615	0,802885	0,293269	-0,29327			
	17-18	76	29	182	196	0,875	0,942308	0,067308	-0,06731			
	19-20	26	12	208	208	1	1	0	0			
Класифікація	0 – 6	8	23	8	23	0,038462	0,110577	0,072115	-0,07212	0,346	0	0,346
	7-10	38	92	46	115	0,221154	0,552885	0,331731	-0,33173			
	11-14	42	45	88	160	0,423077	0,769231	0,346154	-0,34615			
	15-16	76	26	164	186	0,788462	0,894231	0,105769	-0,10577			
	17-18	25	13	189	199	0,908654	0,956731	0,048077	-0,04808			
	19-20	19	9	208	208	1	1	0	0			
Порівняння	0 – 4	5	9	5	9	0,024038	0,043269	0,019231	-0,01923	0,25	0	0,25
	5-7	7	19	12	28	0,057692	0,134615	0,076923	-0,07692			
	8-10	9	28	21	56	0,100962	0,269231	0,168269	-0,16827			
	11-13	14	31	35	87	0,168269	0,418269	0,25	-0,25			
	14-16	57	45	92	132	0,442308	0,634615	0,192308	-0,19231			
	17 –	116	76	208	208	1	1	0	0			
Гнучкість мислення	0-6	10	27	10	27	0,048077	0,129808	0,081731	-0,08173	0,25	0	0,25
	2-3	17	36	27	63	0,129808	0,302885	0,173077	-0,17308			
	4	46	62	73	125	0,350962	0,600962	0,25	-0,25			
	5-6	74	48	147	173	0,706731	0,831731	0,125	-0,125			
	7-8	38	19	185	192	0,889423	0,923077	0,033654	-0,03365			
	9-	23	16	208	208	1	1	0	0			

В таблиці 2.9: T1=max |S1-S2|, T2=max (S1-S2), T3= max (S2-S1).

Для визначення критичного значення статистики для рівня значущості 0,05 при великих обсягах вибірок використовуємо наближену формулу:

$$W_{1-\alpha} \approx \lambda_{\alpha} \sqrt{\frac{n_1 + n_2}{n_1 \cdot n_2}} \quad (2.2),$$

де λ_{α} - квантиль функції Колмогорова для вибраного рівня значущості α=0,05, значення якого: λ_{α}=1,36.

При n_1=208, n_2=208 маємо:

$$W_{1-\alpha} \approx 1{,}36 \cdot \sqrt{\frac{208+208}{208 \cdot 208}} \approx 0{,}13 \quad (2.3).$$

Для досліджуваних інтелектуальних умінь за даними таблиці 2.9 T1>$W_{1-\alpha}$, отже у відповідності з правилом прийняття рішення для двостороннього критерію Колмогорова-Смирнова нульова гіпотеза H_0: $F(x)=G(x)$ для довільного значення x, де $F(x)$ і $G(x)$ – невідомі функції розподілу ймовірностей рівня сформованості інтелектуального уміння в експериментальних і контрольних класах, відхиляється і приймається альтернативна гіпотеза H_1: $F(x) \neq G(x)$ хоча б для одного значення x. Це означає, що існує відмінність розподілу ймовірностей рівня сформованості інтелектуального уміння учнів, які навчалися за традиційною і експериментальною методиками. Якщо врахувати, що T3>$W_{1-\alpha}$ і застосувати односторонній критерій Колмогорова-Смирнова, можна уточнити зроблений висновок: учні, які навчались в експериментальних класах мали вищий рівень сформованості інтелектуальних умінь. Приймаючи до уваги, що в експериментальних класах був введений змінний фактор – методичні прийоми і засоби навчання інформатики на основі об’єктно-орієнтованого програмування, спрямовані на формування інтелектуальних умінь учнів, можна припустити, що саме це і дало можливість досягти кращих результатів.

Входячи до змістовно-операційного компонента пізнавальної діяльності [64, 99, 150], визначаючи широту і глибину пізнавального інтересу, інтелектуальні уміння становлять основу інтелектуальної активності, опосередковано виявляються в ній, “переломлюючись через мотиваційну структуру особистості” [26, с.24].

Використовувані тести ставлять учня в такі умови, в яких він “видає” стільки інтелектуальних умінь і знань, скільки вимагає задача. Проектний підхід, який передбачає як творчу, так і дослідницьку діяльність, дозволяє виявити, чи здатний учень на більше, ніж написати програму, що виконувала б функції, задані умовою задачі, визначити міру активності його інтелекту. Передумовою застосування проектного підходу для визначення рівня інтелектуальної активності учнів є відносна відсутність обмежень у часі процесу самостійного виконання проекту. “Учню необхідно дати час для виявлення творчої ініціативи, а у всіх він різний” [26, с.53]. Завдання з створення проекту, як і будь-яке інше, має деяку “стелю”, яка дозволяє визначити, чи справився учень із задачею, але не обмежує його діяльність: “воно будується як необмежене нічим поле діяльності” [26, с.54]. Інтелектуальна активність виявляється в ситуації подолання і виходу за межі заданої в умові “стелі”. Тобто мається на увазі два типи діяльності. Перший тип – задана умовою задачі діяльність, і другий – подальший аналіз матеріалу, удосконалення розв’язку, що “не диктується “утилітарною” потребою виконати вимогу задачі” [26, с.55]. Інтелектуальна активність учня виявляється при переході до другого типу діяльності після необхідного розв’язку задачі за ініціативою самого суб’єкту, при відсутності зовнішнього стимулу діяльності.

В [26, с.73-79] визначаються три рівні інтелектуальної активності: “стимульно-продуктивний”, “евристичний”, “креативний”. Інтелектуальна активність відноситься до стимульно-продуктивного рівня, якщо учень

реалізував проект в межах поставленого завдання. Евристичний рівень інтелектуальної активності передбачає, що знаючи, як вирішувати поставлену проблему, в ході аналізу предметної галузі учень виявляє якісь нові закономірності, властивості об'єкта, не передбачені умовою завдання, або новий спосіб розв'язування. Це відкриття оцінюється як "власне" доповнення або новий спосіб, що застосовується для розв'язування поставленої задачі. "Звідси – межа інтелектуальної активності евриста". На вищому креативному рівні інтелектуальної активності знайдена учнем закономірність є для нього самостійною проблемою, заради розв'язування якої він готовий відмовитись від попередньо поставленої задачі.

Розглянемо приклад завдання-проекту створення бази даних рослин. "Створити базу даних рослин, в якій передбачити можливості для збереження і вибору наступної інформації: назва рослини, тип стебла, листа, квітки, плоду, життєва форма, район походження". Якщо учень виявляє, що від значень перерахованих в завданні ознак залежить назва родини, роду, порядку, відділу рослини, і включить цю інформацію до створюваної бази даних, то його інтелектуальна активність належатиме до евристичного рівня. Якщо ж учня настільки зацікавлять виявлені закономірності, що він захоче створити електронний визначник хоча б для деякої родини рослин і приєднати його до бази даних, то він виявить креативний рівень інтелектуальної активності.

Визначення рівнів інтелектуальної активності учнів здійснювалося в ході виконання ними завдань-проектів та аналізу результатів роботи. Учні ЕГ мали можливість використовувати в своїй діяльності середовище візуального програмування Delphi, яке дозволяє об'єднувати в одній програмі програмний код на Object Pascal, наприклад, з механізмом розробки баз даних. Учні КГ для реалізації проектів користувалися мовою програмування TURBO Pascal, інструментарієм Microsoft Office.

Таблиця 2.10.

Оцінка інтелектуальної активності учнів в процесі роботи над задачею-проектом.

Групи	Рівні інтелектуальної активності учнів		
	Стимульно-продуктивний	Евристичний	Креативний
ЕГ n_1=208	O_{11}=129	O_{12}=61	O_{13}=18
КГ n_2=208	O_{21}=156	O_{22}=42	O_{23}=10

Позначення до таблиці 2.10:

n_1 – кількість учнів в експериментальній групі;

n_2 – кількість учнів в контрольній групі;

O_{1i} – кількість учнів в експериментальній групі, які мають стимульно-продуктивний (i=1), евристичний (i=2), креативний (i=3) рівні інтелектуальної активності;

O_{2i} – кількість учнів в контрольній групі, які мають стимульно-продуктивний (i=1), евристичний (i=2), креативний (i=3) рівні інтелектуальної активності.

Для виявлення статистично значущих відмінностей в рівнях інтелектуальної активності учнів експериментальних і контрольних груп, визначених в процесі роботи над задачею-проектом та її результатами, сформулюємо нульову гіпотезу H_0: вищий рівень інтелектуальної активності учнів в експериментальній групі пояснюється випадковими факторами, тобто статистично КГ і ЕГ однакові. Альтернативна гіпотеза H_1: цей вищий рівень є результатом використання запропонованої методики навчання. Вибірки незалежні, вимірювана властивість (рівень інтелектуальної активності учнів) виміряна за шкалою порядку, що має три

категорії: стимульно-продуктивний, евристичний, креативний. Скориставшись двостороннім критерієм χ^2, враховуючи, що експериментальні дані подані у формі таблиці 2xC, де C=3 – кількість категорій, для перевірки гіпотези знаходимо значення статистики критерію T досліджуваної випадкової величини за формулою (2.1).

В результаті отримуємо T=6,714. Для числа ступенів вільності ν=C-1=2 та рівня значущості α=0,05 знаходимо за таблицею Г [68, с.130] критичне значення статистики $T_{кр}$=5,991. Оскільки $T>T_{кр}$ отримані результати дають підставу для відхилення нульової гіпотези і прийняття альтернативної, тобто вищий рівень інтелектуальної активності учнів (кількість учнів евристичного та креативного рівнів інтелектуальної активності в експериментальній групі порівняно з контрольною вища на 13%) в процесі розв'язування задач-проектів є результатом запропонованої методики.

В ході навчального процесу серед учнів спостерігалося посилення мотивації навчання інформатики: підвищився інтерес до різних видів навчальної діяльності, зокрема до самостійної роботи. Особливо це спостерігалося в процесі роботи над задачами-проектами в середовищі візуального програмування Delphi. Практична значущість навчального матеріалу з візуального програмування стимулювала учнів до поглибленого вивчення теми, звернення до додаткової літератури. Необхідність застосування у власних програмах графіки створювала мотивацію опанування учнями графічних пакетів Adobe Photoshop 5.5, Corel Draw 8.0. Не меншу зацікавленість учнів з достатнім і високим рівнем сформованості загальних інтелектуальних умінь викликала робота з Case засобами: Visual UML, Rose Delphi Link. Зручні і прості у використанні засоби візуального програмування Delphi дозволяли справлятися з завданням навіть тим учням, що мають нижчий рівень сформованості інтелектуальних умінь.

У процесі проведення експерименту спостерігалося перенесення знань з інформатики та сформованих інтелектуальних умінь на матеріал інших шкільних дисциплін. Кількість учнів експериментальних класів, що досягли високого рівня навчальних досягнень (мають середній бал 10-12), на 16% більша ніж у контрольних класах. Особливо успішно здійснювалося перенесення інтелектуальних умінь, сформованих у процесі вивчення об’єктно-орієнтованого та візуального програмування, на такі навчальні дисципліни, як математика, біологія, що частково було обумовлено тематикою задач, зокрема при створенні експертних систем в DESS, які надавалися учням для розв’язування на уроках інформатики, у процесі роботи над якими їм доводилося самостійно здобувати знання із зазначених навчальних предметів.

Підсумки експерименту показують, що цілеспрямоване використання засобів об’єктно-орієнтованого, візуального програмування у процесі навчання інформатики позитивно впливає на процес формування інтелектуальних умінь учнів. Застосування запропонованої методичної системи, зокрема дослідницького методу та методу аналогії сприяло переходу учнів від репродуктивного рівня засвоєння навчальної інформації до активного її здобуття та логічного осмислення, підвищенню рівня їх інтелектуальної активності та загальної успішності в навчанні. Отже, можна зробити висновок, що педагогічний експеримент підтвердив гіпотезу нашого дослідження.

ВИСНОВКИ

1. З’ясовано, що недостатня практична предметна реалізація сучасних досягнень психології і педагогіки з питання формування інтелектуальних умінь учнів, неповномірне застосування вчителями спеціальних методик не дозволяє забезпечити на належному рівні формування в учнів інтелектуальних умінь, що стосуються таких загальних розумових дій: аналогії, класифікації понять, виділення істотних ознак, варіативності напрямку думки у процесі розв’язування задач.

2. Розроблено компоненти методичної системи формування інтелектуальних умінь старшокласників (цілі, зміст, методи, засоби навчання) у процесі навчання інформатики на трьох рівнях навчально-пізнавальної діяльності учнів (репродуктивному, творчо-репродуктивному, творчо-пошуковому) на основі технологій об’єктно-орієнтованого та візуального програмування з використанням активних методів навчання (методу проектів, дослідницького методу, методу аналогії), зокрема:

- запропоновано структуру і зміст тем: “Основи об’єктно-орієнтованого програмування”, “Основи об’єктно-орієнтованого аналізу та проектування”, “Основи візуального програмування”, які можна вивчати в певних обсягах в залежності від профілю школи;
- сформульовані правила-орієнтири виконання дій у процесі об’єктно-орієнтованого аналізу, проектування і програмування як інваріанти правил виконання загальних розумових дій: аналогії, порівняння, класифікації, виділення істотних ознак;
- розроблені діагностичні, пропедевтичні, тренувальні, контролюючі вправи та тематика задач-проектів;
- розроблено педагогічний програмний засіб та відібрані з відомих програмні засоби для реалізації комп’ютерно-орієнтованої методики

формування інтелектуальних умінь учнів у процесі навчання інформатики.

3. Обгрунтовано, що поєднання цілеспрямованого управління розумовою діяльністю учнів з активними методами навчання забезпечує необхідний рівень сформованості інтелектуальних умінь учнів, реалізацію аксіологічної, когнітивної, діяльнісно-творчої компонент особистісно-зорієнтованої моделі навчання. У процесі застосування активних методів навчання запропоновано використовувати прояв інтелектуальної активності учнів на різних рівнях (стимульно-продуктивному, евристичному, креативному) для оцінювання якості сформованих інтелектуальних умінь.

4. З'ясовано, що формування інтелектуальних умінь та засвоєння учнями теоретичного матеріалу у процесі навчання об'єктно-орієнтованого і візуального програмування ефективне, якщо включає такі етапи: накопичення фонду знань, діагностика, мотивація, рефлексія, тренування, узагальнення та перенесення, контроль і корекція.

5. Для формування інтелектуальних умінь учнів розроблено експертну оболонку DESS (Diagnostic Expert Systems Shell), яку доцільно також використовувати у процесі узагальнення і систематизації знань учнів з інформатики та інших предметів, реалізуючи парадигму навчального процесу "учіння шляхом навчання" ("навчаючи" комп'ютер виконувати діагностику на основі розробленої учнем бази знань експертної системи).

6. Експериментально встановлено, що застосування розробленої методики сприяє формуванню та вдосконаленню умінь виконувати порівняння, класифікацію, виділення істотних ознак, аналогію, варіацію напрямку думки при розв'язуванні задач (розумові дії перераховані за спаданням різниць показників в ЕГ і КГ).

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Абрамов С.А., Гнездилова Г.Г., Капустина Е.Н., Селюн М.И. Задачи по программированию. – М.: Наука. Гл. ред. физ.-мат. лит., 1988. –234 с.
2. Агафонов В.Н. Объектно-ориентированное программирование и абстрактные типы данных // Программирование. -№6, 1990. – С.27-32.
3. Адлер Г. Техника развития интеллекта. – Спб: Питер, 2001. – 192 с.
4. Айзенк Г.О. Проверьте свои способности / Пер. с англ. А.Н.Лука, И.С.Хорола. – М.: Педагогика: Пресс, 1992. – 176 с.
5. Альманах психологических тестов. – М.: КСП, 1996. – 400 с.
6. Амонашвили Ш.А. Гуманно – личностный подход к детям. – М. – Воронеж, 1998. – 544 с. - /”Психология отечества”/.
7. Андреев А.М, Березкин Д.Б., Кантолистов Ю.А. Среда и хранилище: ООБД, обзор по ООБД, включающим средства разработки // Мир ПК. -№4, 1998. – С.74-81.
8. Ананьев Б.Г., Степанова Е.И. Развитие психофизиологических функций взрослых людей (средняя взрослость). – М.: Педагогика, 1977. – 198 с.
9. Апатова Н.В. Влияние информационных технологий на содержание и методы обучения в средней школе: Дис…докт. пед. наук: 13.00.02.- М., 1994.- 348 с.
10. Антипов И.Н. Основы информатики и вычислительной техники: Метод. пособие для преподавателей техникумов. – М.: Высш.шк., 1991. – 246 с.
11. Архангельский С.И. Учебный процесс в высшей школе и его закономерные основы и методы. – М.: Высшая школа, 1980. – 368 с.
12. Бабанский Ю.К. Оптимизация процесса обучения. – М.:Просвещение, 1977. – С.79.

13. Бабанский Ю.К. Рациональная организация учебной деятельности. – М.: Знание, 1981. – 96 с.

14. Бабанский Ю.К. Оптимизация учебно – познавательного процесса: методические основы. – М.: Просвещение, 1982, - 192 с.

15. Бава А.А. Конфігураційно-частотний аналіз у психологічних дослідженнях // Практична психологія та соціальна робота. –№8, 1998.- С.42-45.

16. Бадд Т. Объектно-ориентированное программирование в действии / Перев. с англ. – СПб.: Питер, 1997. – 464 с.

17. Балик Н.Р. Методика вивчення експертних систем у курсі інформатики та обчислювальної техніки: 13.00.02. Дис...канд. пед. наук. – К., 1995. – 192с.

18. Беляева И.Н. Перспективы и возможности курса информатики на современном этапе // Информатика и образование. - №4, 1996. – С.24-28.

19. Беспалько В.П. Слагаемые педагогической технологии. – М.: Педагогика, 1989. – 192 с.

20. Берулава Г.А. Психодиагностика умственного развития учащихся. – Новосибирск: Изд-во НГПИ, 1990.

21. Биков В.Ю., Руденко В.Д. Системи управління інформаційними базами даних в освіті: Навч. посібник для студ. вищ. навч. закладів/АПН України, Інститут педагогіки. – К.: Рад. школа, 1979. – 118 с.

22. Білоконна Н.І. Формування інтелектуальних умінь молодших школярів у процесі навчання: 13.00.01. Дис...канд.пед.наук. – Кривий Ріг, 1998. – 139с.

23. Білоусова Л.І., Гризун Л.Е. Програмно-методичний комплекс “Програмування на Visual Basic”// Комп’ютер у школі та сім’ї. – 1998. - №1. – С. 27-28.

24. Блонский П.П. Избранные педагогические и психологические сочинения. – Т.2. – М., 1979, - С. 118-341.

25. Бобровский С. Delphi: учебный курс – СПб: Питер, 2001. – 640 с.

26. Богоявленская Д.Б. Интеллектуальная активность как проблема творчества. – Ростов: Изд-во Ростов. унив-та, 1983. – 183 с.

27. Богоявленский Д.Н. Прийомы умственной деятельности и их формирование у школьников // Вопросы психологии. - №2, 1969. – С.25 – 38.

28. Бондарев В.М., Рублинецкий В.И., Клачко Е.Г. Основы программирования. – Харьков: Фолио; Ростов на/Д: Феникс., 1997. – 368 с.

29. Бондаревская Е.В. Гуманистическая парадигма личностно ориентированного образования // Педагогика. - №4, 1997. – С.11-17.

30. Боровик А.М. Реформування системи освіти в Україні – об’єктивна необхідність. Педагогічні і психологічні проблеми підготовки вчителів: Матеріали ювілейної наукової конференції, присвяченої 80-річчю ЧДПІ ім.Т.Г.Шевченка. – Чернігів, 1996. – С.10-12.

31. Брудно А.Л., Каплан Л.И. Московские олимпиады по программированию / Под ред. Б.Н. Наумова. – 2-е изд., доп. и перераб. – М.: Наука. Гл. ред. физ.-мат. літ., 1990. – 208 с.

32. Бруновт Е.П., Бровкина Е.Т. Формирование приемов умственной деятельности учащихся. На материале учебного предмета биологии. – М.: Педагогика, 1981. – 70 с.

33. Бугрим М.С., Кузнецов В.И. Введение в современную точную методологию науки: Структуры систем знания. – М.: АО “Аспект Пресс”, 1994. – 304 с.

34. Бурда М.І., Мацько Н.Д., Хмара Т.Н. Особливості організації уроків математики // Радянська школа, 1989.- N 3.- с.60-64.

35. Буч Г. Объектно-ориентированный анализ и проектирование с примерами приложений на C++, 2-е изд./ Пер. с англ. – М.: «Издательство Бином», СПб.: «Невский диалект», 1999. –560 с.

36. Буч Г., Рамбо Д., Джекобсон А. Язык UML. Руководство пользователя: Пер. с англ. – М.: ДМК, 2000. – 432 с.

37. Василенко І.М. Розв'язування задач як засіб розвитку інтелектуальних умінь учнів на уроках інформатики. Комп'ютерно-орієнтовані системи навчання: Зб. наук. праць/ Редкол. Випуск 2. –К.: НПУ ім. М.П. Драгоманова. –2000. - С. 213-223.

38. Василенко І.М. Настенко Л.Г. Формування інтелектуальних умінь в контексті розвитку особистісної культури. Наука і сучасність: Збірник наукових праць Національного педагогічного університету імені М.П.Драгоманова. – К.: Логос, 1999. Ч.ІІ. – С. 23 – 30.

39. Василенко І.М. Реалізація методики розвиваючого навчання в процесі вивчення інформатики. Сучасні інформаційні технології в навчальному процесі: Зб. наук. праць/ Редкол. – К.: НПУ. – 1997. – С. 215 – 223.

40. Василенко І.М. Формування інтелектуальних якостей майбутнього вчителя на уроках інформатики. Профорієнтація та довузівська підготовка майбутніх спеціалістів: проблеми, досвід, перспективи: Матеріали Всеукраїнської науково-методичної конференції/ Редкол. – Чернігів: ЧДПУ імені Т.Г.Шевченка. – 1999. – С. 113 – 117.

41. Василенко І.М. Формування прийомів продуктивного мислення на уроках інформатики. Вісник ЧДПУ імені Т.Г. Шевченка. Випуск 4. Серія: педагогічні науки: Збірник. – Чернігів: ЧДПУ, 2001. - №4. – С.27-30.

42. Васильев В. Объектно-ориентированная БД: взгляд изнутри // Компьютеры + программы. -№3, 1997. -С.45-49.

43. Венгер Л.А. Диагностика умственного развития дошкольников. – М.: Педагогика, 1978.

44. Верлань А.Ф., Апатова Н.В. Інформатика. Підручник для учнів 10-11 класів середньої школи. – К.: Квазар-Мікро, 1998. – 200 с.

45. Вернадский В.И. Размышление натуралиста. Научная мысль как планетарное явление. – М.: Наука, 1984. – 191 с.

46. Вертгеймер М. Продуктивное мышление. – М.: Прогресс, 1987. – 336с.

47. Возрастная и педагогическая психология / Под ред. Петровского А.В. – М.: Педагогика, 1973. – 288 с.

48. Видинеев Н.В. Природа интеллектуальных способностей человека. – М.: Мысль, 1989. – 173 с.

49. Вітюк О.В. Засоби новітніх інформаційних технологій навчання при вивченні стереометрії. Комп'ютерно-орієнтовані системи навчання: Зб. наук. праць/ Редкол. Випуск 2. –К.: НПУ ім. М.П. Драгоманова. –2000.- С.224-232.

50. Воробьев А.Н. Тренинг интеллекта. – М.: Лесная промышленность, 1989.- 175 с.

51. Выготский Л.С. Мышление и речь // Собр. Соч. – Т.2. – М.: Педагогика, 1982. – С. 5-361.

52. Вьюнкова Ю.Н. Проблемы коррекционно-развивающего обучения // Педагогика. - №1, 1999. – С.50-56.

53. Гаврилова Т.А., Хорошевский В.Ф. Базы знаний интеллектуальных систем.– СПб.: Питер, 2001. – 384 с.

54. Гальперин П.Я. Опыт изучения формирования умственных действий. В кн. Доклады на совещании по вопросам психологии. М.: Изд-во АПН РСФСР, 1974, С. 188-201.

55. Гальперин П.Я. Введение в психологию. – М., 1976, - 119 с.

56. Гальперин П.Я., Талызина Н.Ф. Современное состояние теории поэтапного формирования умственных действий. – Вестник МГУ. Серия 14. Психология. – М., 1979. – С.54-63.

57. Гальперин П.Я. Общий взгляд на учение о так называемом поэтапном формировании умственных действий, представлений и понятий // Вестник Московского университета. Серия 14. Психология. - №2, 1998. – С.3-7.

58. Гальтон Ф. Наследственность таланта: Законы и последствия: Пер. с англ. – М.: Мысль, 1996. – 272 с.

59. Гильбух Ю.З. Темперамент и познавательные способности школьника: Психология, диагностика, педагогика /АПН Украины. Ин-т психологии. К., 1993. – 261 с.

60. Гильбух Ю.З. Внимание: одаренные дети. – М.: Знание, 1991. – 80 с.

61. Гилфорд Дж. Три стороні интеллекта // Психология мышления / Под ред. А.М. Матюшкина. – М.: Прогресс, 1965.

62. Гин А.А. Приемі педагогической техники: Свобода выбора. Открытость. Деятельность. Обратная связь. Идеальность: Пособие для учителей. – Гомель: ИПП “Сож”, - 1999. – 88с.

63. Гнатюк Н.П. Нет предела совершенству … Кн. для учащихся. – М.: Просвещение, 1989. – 208 с.

64. Головань М.С. Розвиток пізнавальної активності учнів в процесі навчання алгебри і початків аналізу на основі НІТ : 13.00.02. Дис...канд. пед. наук. – К., 1997, - 177 с.

65. Гончаренко С.А., Мартынова Л.М. Комплексная диагностика интеллектуального развития учащихся 7-11 кл. // Практична психологія та соціальна робота. –№4, №5,1998. – С. 24-28, С.14-18.

66. Гончаренко С.У. Український педагогічний словник. – К.: Либідь, 1997. –374с.

67. Горошко Ю.В. Вплив нової інформаційної технології на практичну значимість результатів навчання математики в старших класах середньої школи: 13.00.02.Дис...канд.пед.наук.- К., 1993. – 103 с.

68. Грабарь М.И., Краснянська К.А. Применение математической статистики в педагогических исследованиях. – М.: Педагогика, 1977. – 136 с.

69. Граник Г.Г. Психология модель процесса формирования умения // Вопросы психологии. – 1979, - №3. – С. 56-65.

70. Грузман М.З. Обучение учащихся средней школы программированию на основе структурного подхода: 13.00.02. Дис…канд.пед.наук. – К., 1986. – 133с.

71. Грузман М.З. Эвристика в информатике. Винница: Арбат, 1998.- 305 с.

72. Губенко О.В. Проблема вивчення та розвитку мислення: деякі методичні аспекти// Практична психологія та соціальна робота. –№6-7, 1998.- С.19-20.

73. Гуржій А.М. та ін. Засоби навчання: Навч. посібник для студ. вузів та слухачів підвищ. кваліфікації/ МО України. ІЗМН. АПН України. Інститут педагогіки. – К., 1997. – 208 с.

74. Гуржій А.М., Жук Ю.О. Інформатика і школа: проблеми і перспективи // Комп'ютер у школі та сім'ї. – 1998. - №1. – С. 8-10.

75. Давыдов В.В. Проблемы развивающего обучения. – М.: Педагогика, 1986. – 240 с.

76. Дидактика современной школы: Пособие для учителя / Под ред. В.А.Онищука. – К.: Рад. шк., 1987. – 351с.

77. Єгорова М.Е. Створення умов особистісного зростання та розвитку учнів на уроках інформатики в умовах Довгінцівського ліцею на підставі використання особистісно зорієнтованої педагогічної парадигми через запровадження творчо-ігрових елементів // Інформатика та комп'ютерно-орієнтовані технології навчання: Збірник наукових праць Всеукраїнської науково-практичної конференції (м. Хмельницький, 16-18 травня 2001 року)/ Редкол. – К.: Педагогічна Думка. – 2001. С. 88-90.

78. Егорова М.Э. Сборник задач по информатике с решениями. – Кривой рог., 1997. – 261 с.

79. Егорова М.Э. Основы программирования TURBO Pascal.: Практические работы .- Днепропетровск: Упр. образ. Днепр. обл. админ.; Днепр. ин-т образ., 1999. – 69 с.

80. Еремин Е.А. Почему система Delhi интересна для образования? // Информатика и образование. - №1, 1997. – С.65-69.

81. Ерецкий М.И., Пороцкий Э.С. Проверка знаний, умений и навыков учащихся техникумов. – М., 1978, - С. 11-12.

82. Ермолаев О.Ю., Марютина Т.М. Индивидуальность школьника и компьютеры. – М.: Знание, 1990. – 80 с.

83. Ершов А.П., Звенигородский Г.А., Первин Ю.А. Школьная информатика (концепция, состояние, перспективы). – Новосибирск: Препринт ВЦСО АН СССР, - № 152, 1979. – С. 4-29.

84. Ефимов Е.И. Решатели интеллектуальных задач. – М.: Наука, 1982. – 316с.

85. Жалдак М.І. Подготовка учителя к использованию информационной технологии в учебном процессе. Автореф. дисс. … докт. пед. наук. – М.: 1991.– 48 с.

86. Жалдак М.І., Морзе Н.В., Рамський Ю.С. “Основи інформатики” як одна з вагомих складових систем навчальних предметів загальноосвітньої школи // Сучасні інформаційні технології в навчальному процесі. – К.: НПУ.- 1997.- С.3-21.

87. Жалдак М.І. Комп’ютер на уроках математики: Посібник для вчителів. – К.: Техніка, 1997. – 303 с.

88. Жильцов О.Б. Розвиток розумової діяльності учнів 7 класів середньої школи при вивченні математики з використанням НІТ:13.00.02. Дис...канд. пед.наук. – К., 1994. – 227 с.

89. Жуков А. Изучаем Delphi – СПб: Питер, 2000. – 352 с.

90. Жук Ю.А. Решение исследовательских задач по физике с использованием новых информационных технологий: 13.00.02. Дис…канд.пед.наук – К., 1995.– 217 с.

91. Забокрицкая Е.И. Виды познавательных заданий для лабораторных и практических работ (в обучении естественным предметам в 7-9 классах общеобразовательной школы): 13.00.01. Дис...канд.пед.наук. – К., 1984. – 229с.

92. Завада Т.А. Система заданий учебника как средство формирования интеллектуальных умений у школьников (на материале предметов гуманитарного цикла в 4-6 классах): 13.00.01. Дис...канд.пед.наук. – К., 1989.- 205 с.

93. Завада Т.О. Підручник як засіб активізації пізнавальної діяльності учнів. Педагогічні і психологічні проблеми підготовки вчителів: Матеріали ювілейної конференції, присвяченої 80-річчю ЧДПУ імені Т.Г.Шевченка. – Чернігів, 1996. – С.85-87

94. Зак А.З. Как определить уровень развития мышления школьника. – М.: Знание, 1989. – 96 с.

95. Занков Л.В. Избранные педагогические труды. – М.: Педагогика, 1990. – 424 с.

96. Зарецька І.Т., Колодяжний Б.Г., Гуржій А.М., Соколов М.О. Інформатика: Навч. посібник для 10 – 11 кл. загальноосвіт. шк. – Х.: Факт; К.: Навчальна книга, 2002. – 496 с.

97. Зверева И.М. Активизация мышления учащихся на уроках физики: Из опыта работы: Пособие для учителей. – М.: Просвещение, 1980. – 112 с.

98. Зеленяк О.П. Программирование в среде Turbo Pascal. -Александрия: ОАО “Александрийская фабрика диаграммных бумаг”, 1999. – 308 с.

99. Іваськів І.С. Активізація навчально-пізнавальної діяльності учнів на основі систем штучного інтелекту при навчанні інформатики в старшій школі: 13.00.02. Дис...канд.пед.наук. – К., 2000. - 250 с.

100. Ігнатенко М.Я. Методологічні та методичні основи активізації навчально-пізнавальної діяльності учнів старших класів при вивченні математики: 13.00.02. Дис...докт.пед.наук. – К., 1997. – 335с.

101. Інформаційні технології в аналітичній геометрії: Навч. посіб. для студ. мат. спец. ун-тів/ С.А. Раков, В.П. Горох, Т.О. Олійник та ін. – Х.: ХДПУ, 2000. – 190 с.

102. Изучение языков программирования в школе: Пособие для учителя/ Н.И.Шкиль, М.И.Жалдак, Н.В.Морзе, Ю.С.Рамський. – К.: Рад.шк., 1988. – 272 с.

103. Кабанова-Меллер Е.Н. Психология формирования знаний и навыков у школьников. – М.: Из-во АПН РСФСР, 1962. – 356 с.

104. Кабанова-Меллер Е.Н. Формирование приёмов умственной деятельности и умственное развитие учащихся. – М.: Просвещение, 1968. – 288 с.

105. Калмыкова З. И. Продуктивное мышление как основа обучаемости. – М.: Педагогика, 1981. – С. 17-25.

106. Калмыкова З.И. Психологические принципы развивающего обучения. – М.: Знание, 1979.- 48 с.

107. Калмыкова З.И. Педагогика гуманизма. – М.: Знание, 1990. – 80 с.

108. Калмыкова З.И. Процессы анализа и синтеза при решении задач // Известия РСФСР. – 1955.-№7.

109. Караванова Т.П. Елементи розвивального навчання у викладанні інформатики // Комп'ютер у школі та сім'ї. – 2000. - №3. – С. 16-17.

110. Карпов Ю.А., Талызина Н.Ф. Критерии интеллектуального развития детей // Вопр.псих. – 1985.- №2.- С.52-59.

111. Касяненко О.Ю. Психологічна структура дослідницької установки особистості // Практична психологія та соціальна робота. - №5, 1998.- С.7-20.

112. Келер В. Некоторые задачи гельштат-психологии // Хрестоматия по истории психологии. – М.: Изд-во Моск. ун-та, 1980. - С.102-120.

113. Клочко В.І. Нові інформаційні технології навчання математики в технічній вищій школі: 13.00.02 . Дис...докт.пед.наук. – Вінниця, 1998. – 396с.

114. Колин К.К. Информатика сегодня и завтра // Информационные технологии. – М.: Машиностроение. - №1, 2000. – С.2-8.

115. Коновалец Л.С. Познавательная самостоятельность учащихся в условиях компьютерного обучения // Педагогика. -№2, 1999. – С.46-50.

116. Конопка Р. Создание оригинальных компонентов в среде Delphi. К.: НИПФ-Диа Софт Лтд.,1996. – 512 с.

117. Коржуев А.В. О классификациях, используемых в дидактических исследованиях // Педагогика. - №1, 1997. – С.33-37.

118. Коротяев Б.И. Методы учебно–познавательной деятельности учащихся: 13.730. Автор…докт.пед.наук. – М., 1971. – 37 с.

119. Копаєв О.В. Вплив сучасних інформаційних технологій на вивчення основ алгоритмізації в школі // Комп'ютер у школі та сім'ї. - №2, 2000. – С.24–27.

120. Костюк Г.С. Навчально-виховний процес і психологічний розвиток особистості. – К.: Радянська школа, 1989. – 608 с.

121. Кравец Г.П. Самостоятельная работа по информатике. Выпуск 1. – Донецк, 1998. - 12 с.

122. Крамаренко В.Ю. Интеллект и уровни его развития. Дис. канд. наук. – М.: Моск. ун-т, 1983.

123. Краус И., Линерт Г.А. Основы конфигурационно-частотного анализа (КЧА) и его модификации // Психологический журнал. – Том 5.-№1, 1984.-С.26-34.

124. Крутецкий В.А. Основы педагогической психологии. – М.: Просвещение, 1972. – 255 с.

125. Крутецкий В.А. Психология математических способностей школьников.– М.: Просвещение, 1968. – 432 с.

126. Кудрявцев В.Т. Проблемное обучение: истоки, сущность, перспективы. – М.: Знание, 1991. – 80 с.

127. Кузнецов А.А. Школьная информатика:что дальше? // Информатика и образование. - №7, 1997. – С. 14-16.

128. Кузнецов А.Б. Программа курса “Основы объектно-ориентированного программирования”// Информатика и образование. - № 7, 1998. – С.17-24.

129. Курнос Д.И. Индивидуальность и творческое мышление. – М.: Мысль, 1992. – 171 с.

130. Лазаревский С.В. Формирование общеучебных интеллектуальных умений у старшеклассников (на материале дисциплин естественнонаучного цикла): 13.00.01. Автореф…канд.пед.наук. – К., 1990. – 24 с.

131. Лейкина Н.Ю. Самостоятельная работа на уроке как фактор активизации учебно-познавательной деятельности школьников: 13.00.01. Дис...канд.пед.наук. –Л., 1984. – 264 с.

132. Лельчук Т.И. Параллельные возможности объектно-ориентированных языков // Программирование. -№6, 1990. – С.33-43.

133. Леонтьев А.М. Деятельность. Сознание. Личность. – М.: Политиздат, 1977. –304 с.

134. Леонтьев А.М. Проблема развития психики. – М.: Педагогика, 1965.-506с.

135. Леонтьев А.М. Психология образа // Вестник Московского ун-та. Серия 14. Психология. - №2.- 1979. – С.3-13.

136. Леонтьев А.Н. Избранные психологические произведени: В 2т., – Т.2. – М.: Педагогика, 1983. – С.79-92.

137. Лернер И.Я. Дидактические основы методов обучения. – М.: Педагогика, 1981. – 186 с.

138. Лесневский А.С. Практикум по объектно-ориентированному проектированию и программированию // Информатика и образование. -№5, 1998. – С. 114-121.

139. Лозница В.С. Формирование стратегии поиска аналогов как метод развития творческого мышления старшеклассников: 19.00.01. Дис…канд.псих.наук. – К., 1983. – 142 с.

140. Лукаш І.М. Об’єктно-орієнтоване програмування як засіб формування інтелектуальних умінь учнів у процесі навчання інформатики // Комп’ютер у школі та сім’ї. - №2 (14), 2001. – С.5 –11.

141. Лукаш І.М. Об’єктно-орієнтована технологія створення програмних проектів як засіб розвитку інтелектуальних умінь учнів // Інформатика та комп’ютерно орієнтовані технології навчання: Зб. наук. Праць Всеукраїнської науково-практичної конференції (м. Хмельницький, 16-18 травня 2001 року)/ Редкол. – К.: Педагогічна думка. – 2001. – С.110-112.

142. Лукаш І.М. Середовище візуального програмування Delphi як засіб формування інтелектуальних умінь учнів при навчанні інформатики // Педагогіка математики і природознавства. IV Всеукраїнські читання, присвячені пам’яті М.В. Остроградського, 4-5 жовтня 2000 р. Збірник статей.– Полтава: ПОІПОПП, 2000. – С.124-126.

143. Лукаш І.М. Порівняльний аналіз при поданні навчального матеріалу як засіб інтелектуального розвитку учнів (на прикладі порівняння об’єктних моделей Turbo Pascal 7.0 і Delphi). Комп’ютерно-орієнтовані системи

навчання: Зб. наук. праць/ Редкол. Випуск 3. –К.: НПУ ім. М.П. Драгоманова.–2001. - С. 236-246.

144. Лукаш І.М. Застосування Delphi для реалізації методу проектів при навчанні інформатики. Комп'ютерно-орієнтовані системи навчання: Зб. наук. праць/ Редкол. – К.: НПУ ім.М.П.Драгоманова. – Випуск 5. – 2002. – С. 313-322.

145. Лутай В.С. Філософія сучасної освіти: Навчальний посібник. – К.: Центр "Магістр – S" творчої спілки вчителів України, 1996. – 256 с.

146. Люблинская А.А. Детская психология. – М.: Просвещение, 1971. – 356с.

147. Ляшенко О.І. Формування фізичного знання в учнів середньої школи: Логіко-дидактичні основи. – К.: Генеза, 1996. – 128 с.

148. Майерс Г. Надежность программного обеспечения. – М.: Мир, 1980. – 360с.

149. Мартинюк М.Т. Науково- методичні засади навчання фізики в основній школі**:** Дис... д-ра пед. наук: 13.00.02 / Інститут педагогіки АПН України. — К., 1998. — 441с.

150. Мар'янеко Л.В. Психологічні умови формування пізнавальної активності слабовстигаючих старшокласників: 19.00.07. Дис…канд. психол. наук. – К., 1992. – 201 с.

151. Маслоу А. Новые рубежи человеческой природы / Под общ. Ред. Г.А.Балла, А.Н. Киричука, Д.А.Леонтьва. – М.: Смысл, 1999. – 425с.

152. Математическая энциклопедия. – М.: Советская энциклопедия, 1982. – Т.3. – 1183 с.

153. Матюшкин А.М. Проблемные ситуации в мышлении и обучении. – М.: Педагогика, 1972. – 208 с.

154. Махмутов М.И. Теория и практика проблемного обучения. – Казань: Татарское книжное издательство, 1972. –551 с.

155. Машбиц Е.И. Психологические основы управления учебной деятельностью.- К.: Вища школа, 1987.- 224 с.

156. Мельникова Е. Россия – США: технологии формирования образовательного потенциала. // Alma mater (Вестник высшей школы). – 1998.- №6. – С.37-44.

157. Менчинская Н.А. Педагогические проблемы активности личности в обучении // Проблемы социалистической педагогики. – М.: Педагогика, 1973.– С.146-158.

158. Менчинская Н.А. Проблемы учения и умственного развития школьника.– М.: Педагогика, 1989. – 219 с.

159. Мир детства: Подросток / Под ред. Хриптовой; Отв. ред. Г.Н. Филонов.– 2-е изд., доп. – М.: Педагогика, 1989. – 288 с.

160. Миллс Р., Лингер Р. Теория и практика структурного программирования.– М.: Мир, 1982. – 358 с.

161. Михайлов Ф.Т. Философия образования: ее реальность и перспективы // Вопросы философии. - №8, 1999. – С.92-118.

162. Михайлович Т.С. Формирование логических умений у младших школьников в процессе решения задач: 13.00.01. Дис…канд.пед.наук. – К., 1990.

163. Могилев А.В., Хеннер Е.К. О понятии “Информационное моделирование” // Информатика и образование. - №8, 1997. – С.3-7.

164. Моляко В.А. Психология решения школьниками творческих задач. – К.: Рад. Школа, 1983. – 94 с.

165. Монахов В.М. Программирование. Факультативный курс. – М.: Просвещение, 1974. – 159 с.

166. Морзе Н.В. Практичні роботи з основ алгоритмізації та програмування мовами Basic та Pascal. Частина 2. – К., 1996. – 76 с.

167. Морзе Н.В. Прийоми розумової діяльності учнів при вивченні ОІОТ // Рад.школа. – 1986 - №8. –С.24-32.

168. Немов Р.С. Общие основы психологии // Психология. В трех кн. Книга1.– М.: Просвещение, Владос, 1995. – 576 с.

169. Ніколаєнко О.Ю. Вивчення елементів візуального програмування в педагогічному вузі. Комп'ютерно-орієнтовані системи навчання: Зб. наук. праць/ Редкол. Випуск 2. –К.: НПУ ім. М.П. Драгоманова. –2000. - С. 195-202.

170. Никольская А.А. Методы и формы обучения в историческом ракурсе // Педагогика. – №5, 1999. – С. 52 – 56.

171. Нильсон Н. Искусственный интеллект. – М.: Мир, 1973. – 270 с.

172. Озолиныш Р.П. Обучение учащихся VII-VIII классов интеллектуальным умениям самостоятельной познавательной деятельности в процессе решения учебных задач: 13.00.01. Дис...канд.пед.наук. – Рига, 1984, -201 с.

173. Околелов О.П. Системный подход к построению электронного курса для дистанционного обучения // Педагогика. – №6, 1999. – С. 50 – 56.

174. Орлов В.И. Знания, умения и навыки учащихся // Педагогика. – 1997, - №1. – С. 33-39.

175. Освіта України. (газета) - №6, 7 лютого, 2001. – С.9-10.

176. Островская Е.М. Моделирование на компьютере // Информатика и образование. - №7, 1998. – С.64-70.

177. Остроградський М.В., Бум І.А. Роздуми про викладання // Постметодика.– 1996. - №2 (12). – С.44-54.

178. Павская Л.М. Дидактические основы построения системы учебных задач: 13.00.01. Автореф…канд.пед.наук. – К., 1985. – 22 с.

179. Паламарчук В.Ф., Орлов С.И. НОТ школьника – путь к творчеству. – К.: Рад. школа, 1988, - 134 с.

180. Паламарчук В.Ф. Школа учит мыслить. – 2-е изд. доп. и перераб. – М.: Просвещение, 1987. – 208 с.

181. Паламарчук В.Ф. Дидактические основы формирования мышления учащихся в процессе обучения: 13.00.01. Дис…д-ра пед.наук.-К.,1983. – 393 с.

182. Паламарчук В.Ф. Як виростити інтелектуала?: Посіб. для вчителів.- К.: Знання, 1999.- 112 с.

183. Паламарчук В.Ф. Техне інтелектус (технологія інтелектуальної діяльності учнів). – Суми: ВВП “Мрія-1” ЛТД, 1999. – 92 с.

184. Пасічник І.Д. Психологія операційних структур мислительної діяльності (генезис дії систематизації на математичному матеріалі): 19.00.07. Дис...докт. психол. наук. – Рівне, 1993. – 261 с.

185. Пеньков А.В. Использование новой информационной технологии при преподавании математики в старших классах средней школы: 13.00.02. Дис.… канд. пед. наук. – К., 1992. – 171 с.

186. Пиаже Ж. Психология интеллекта // Избранные психологические труды.– М.: Просвещение, 1969. – С.206.

187. Підгорна Т.В. Деякі питання використання інформаційної технології для підвищення пізнавальної активності учнів. Сучасні інформаційні технології в навчальному процесі: Зб. наук. праць/ Редкол. – К.: НПУ. – 1997. – С.170-176.

188. Пильщиков В.Н. Сборник упражнений по языку Паскаль. – М.: Наука. Гл. ред. физ.-мат. лит., 1989. – 160 с.

189. Пиявский С.А., Кадочкин Д.Е. Программное обеспечение образовательной технологии развития одаренности // Программные продукты и системы. - №2, 1998. – С. 28-32.

190. Пойа Д. Как решать задачу /Пер. с англ. – М.: Учпедгиз, 1961. – 205 с.

191. Пойа Д. Математическое открытие /Пер. с англ. – М.: Наука, 1976. – 448с.

192. Платонов К.К. Психология. – М., 1997, - С. 80-82.

193. Плотников В.А. Объектно-ориентированное программирование при реализации метода граничных елементов в задачах упругой динамики: 05.02.07. Дис…канд. техн.наук. – Запорожье, 1995. – 105 с.

194. Поддубная Т.Н., Фукс И.Л. Информатика в задачах и упражнениях. – Томск: МП «Раско», 1992. – 128 с.

195. Пономарев Я.А. Знания, мышление и умственное развитие. – М., 1967. – 264 с.

196. Приходченко К.І. Розвиток творчих здібностей учнів 9 – 11 класів на уроках української літератури. 13.00.01. дис. … канд. пед. наук. – Донецк, 1997. – 150 с.

197. Процесс обучения // Основы дидактики / Под ред. Б.П. Есипова. – М., 1967, С. 197-198.

198. Психология / Под ред. А.А. Смирнова, А.Н. Леонтьева и др. – М., 1962, - 559 с.

199. Психология / Под ред. А.В. Петровского. – М., 1986, - С. 107-116.

200. Психология деятельности // Общая психология / Под ред. В.В.Богоявленского и др. – М., 1981, - С. 138-140.

201. Психологічний словник / За ред. Войтка В.І. – К.: Вища школа, 1982, - 215с.

202. Психологические тесты для деловых людей / Сост. Н.А. Литвинцева – М.: ЗАО “Бизнес-школа “Интел-синтез”, 1998,- 528 с.

203. Психологические тесты / Сост. Ахмеджанов Э.Р.- М., 1996.-320 с.

204. Ракитина Е.А., Лыскова В.Ю. Информационные поля в учебной деятельности // Информатика и образование. - №1, 1999. – С.19-25

205. Рамський Ю.С., Балик Н.Р. Методичні основи вивчення експертних систем у школі. – К.: Логос, 1997. – 114с.

206. Рамський Ю.С., Лукаш І.М. Методика навчання основ об'єктно-орієнтованого програмування// Комп'ютер у школі та сім'ї - №1-6, 2002.

207. Рахманина Н.Н. Формирование межсистемных знаний и умений учащихся средствами проблемного обучения: 13.00.01. Дис...канд.пед.наук. – М., 1973. – 252 с.

208. Решетова З.А. Структура ориентировочной деятельности и ее особенности при формировании теоретического мышления // Вестник Московского университета. Серия 14. Психология. - №2, 1998. – С.14-20.

209. Романець В.А. Психологія творчості. – К.: Вища школа, 1971. – 245 с.

210. Ротенберг В.С., Бондаренко С.М. Мозг. Обучение. Здоровье: Кн. Для учителя. – М.: Просвещение, 1989. –239с.

211. Рубинштейн С.Л. Основы общей психологии. – М.: Учпедгиз, 1946. – 704с.

212. Рубинштейн С.Л. О мышлении и путях его исследования. – М.: Изд-во АН СССР, 1958. – 146 с.

213. Рубинштейн С.Л. Проблемы общей психологии. – М.: Педагогика, 1973.– 423 с.

214. Руденко В.Д., Макарчук О.М., Патланжоглу М.О. Практичний курс інформатики / За ред. Мадзігона В.М. – К.: Фенікс, 1997. – 304 с.

215. Савченко В.С. Разработка алгоритмов: от простого к сложному. – Донецк, 1996. – 320 с.

216. Самарин Ю.А. Воспитание способностей у детей // Стенограмма публичной лекции. – Ленинград, 1954. – 40 с.

217. Самарин Ю.А. Очерки по методике преподавания психологии в средней школе. – М.: Изд-во АПН РСФСР, 1950. – 176 с.

218. Саранцев Г.И. Метод обучения как категория методики преподавания // Педагогика. – №1, 1998. – С. 28 – 34.

219. Саранцев Г.И. Теория, методика и технология обучения // Педагогика. – №1, 1999. – С. 19 – 24.

220. Селье Г. От мечты к открытию: Как стать ученым / Пер. с англ. Войкунской; Общ. ред. М.Н. Кондрашовой, И.С. Хорола. – М.: Прогресс, 1987. – 366 с.

221. Семенець С.П. Розвиток продуктивного мислення учнів при вивченні алгебри і початків аналізу: 13.00.02. Автореф...канд.пед.наук. – К., 1998. – 19с.

222. Семеріков С.О. Активізація пізнавальної діяльності студентів при вивченні чисельних методів у об’єктно-орієнтованій технології програмування: 13.00.02. Дис...канд.пед.наук. – К., 2001. – 255 с.

223. Сергєєв О.В. та ін. Міжпредметні зв’язки під час вивчення фізики в середній школі: Посібник для вчителів. – К.: Рад. школа, 1979. – 118 с.

224. Сергиевский М.В., Шалашов А.В. Турбо Паскаль 7.0: Язык, среда программирования. – М.: Машиностроение, 1994.- 254 с.

225. Сисоєва С.О. Підготовка вчителя до формування творчої особистості учня. – К.: Полиграф-книга, 1996. –406 с.

226. Скаткин М.Н. Совершенствование процесса обучения. – М.: Педагогика, 1971. – 206 с.

227. Слепкань З.И. Психолого-педагогические основы обучения математике: Метод. Пособие. – К.: Рад. школа, 1983. – 192 с.

228. Слепкань З.И. Методическая система реализации развивающей функции обучения математике в средней школе: Дис… докт. пед. наук. – М., 1987. – 47с.

229. Слэйг Дж. Искусственный интеллект. – М.: Мир, 1973. – 319 с.

230. Смирнова Е.Н. Развитие важнейших компонентов интеллекта на основе комплексного использования НИТ при обучении математике в старшей школе: 13.00.02. Дис...канд.пед.наук.- К., 1997. – 258 с.

231. Смульсон М.Л. Психологія розвитку інтелекту: Монографія. – К., 2001. – 276 с.

232. Соколов В.В. Основи програмування в Delphi // Комп'ютер у школі та сім'ї. - №3, 2001. – С.21 –25.

233. Сохор А.М. Логическая структура учебного материала. Вопросы дидактического анализа. – М.: Педагогика, 1974. – 192 с.

234. Сравнительный анализ средств быстрой разработки приложений Visual Basic, Delphi, Power Object // Argc&Argv: журнал для профессиональных программистов. - №3(15), 1998. – С. 12-22.

235. Степанова М.А. Представление о параметрах умственных действий в психологическом учении П.Я. Гальперина // Вестник Московского университета. Серия 14. Психология. - №3, 1998. – С.95-103.

236. Сухомлинский В.А. Об умственном воспитании. – К.: Рад. школа, 1883, - 206 с.

237. Сухарев В.О. Психологія інтелекту. – Донецьк: Сталкер, 1997. – 416 с.

238. Сыромолотов Е. Семь законов оценки знаний // Alma mater (Вестник высшей школы). - №1. – 1998. – С.10-14.

239. Талызина Н.Ф. Формирование познавательной деятельности учащихся. – М.: Знание, 1983. – 96 с.

240. Талызина Н.Ф. Управление процессом усвоения знаний. – М.: Изд-во МГУ, 1975. – 345 с.

241. Талызина Н.Ф. Новые подходы к психодиагностике интеллекта // Вестник Московского университета. Серия 14. Психология. - №2, 1998. – С.8-13.

242. Теплицький І.О. Розвиток творчих здібностей школярів засобами комп'ютерного моделювання: 13.00.02. Дис...канд.пед.наук. – К., 2001. – 227с.

243. Тихомиров О.К. Структура мыслительной деятельности человека. – М.: Изд-во Моск. ун-та, 1969. – 304 с.

244. Токарь С. Объектно-ориентированные базы данных (ООБД) // Компьютерное обозрение. – Декабрь, 1993. – С.11.

245. Токарь С., Штонда В. Объектно-ориентированный анализ для программистов // Компьютерное обозрение. – Октябрь, 1993. – С. 3-8.

246. Триус Ю.В., Бакланова М.Л. Проблеми вивчення математичних дисциплін у коледжах та шляхи їх подолання.Комп'ютерно-орієнтовані системи навчання: Зб.наук.праць/ Редкол. – К.: НПУ імені М.П.Драгоманова. – Випуск 6. - 2003. – С.118-137.

247. Тыугу Э.Х. Объектно-ориентированное программирование // Программирование. - №6, 1990. – С. 16-26.

248. Україна XXI століття. Державна національна програма "Освіта" / Міністерство освіти України. – К.: Компас, 1992. – 70 с.

249. Унт И.Э. Индивидуализация и дифференциация обучения. – М.: Педагогика, 1990. – 192 с.

250. Усова А.В., Бобров А.А. Формирование у учащихся учебных умений. – М.: Знание, 1987. – 80 с.

251. Фейгенбаум Э., Фельдман Дж. Искусственный разум. Введение. – В сб. Вычислительные машины и мышление. – М.: Мир, 1967. – С. 23-32.

252. Формирование учебной деятельности / Под ред. Давыдова В.В., МарковаА.К., Лапнер И. – М.: Педагогика, 1982. – 216 с.

253. Фридман Л.М., Волков К.И. Психологическая наука – учителю. – М.: Просвещение, 1985. – 224 с.

254. Фридман Л.М. Психолого-педагогические основы обучения математике в школе. – М.: Просвещение, 1983. – 158 с.

255. Фридман Л.М. Логико-психологический анализ школьных учебных задач. – М.: Педагогика, 1977.- 201 с.

256. Фурман А.В. Психолого-педагогічна теорія навчальних проблемних ситуацій: 19.00.07. Дис...док.псих.наук. – К., 1993. – 449 с.

257. Холодная М.А. Структурная организация индивидуального интеллекта. Дис...докт.психол. наук - М., 1990.

258. Холодная М.А. Психология интеллекта: парадоксы исследования. – Томск: Изд-во Том. ун-та. М.: Изд-во “Барс”, 1997. – 329 с.

259. Холодная М.А., Гельфман Э.Г. Интеллектуальное воспитание личности. // Педагогика. – 1998. - №1. – С.54-60.

260. Хомич Л.А. Система межпредметных заданий как средство формирования научного мировоззрения школьников (9-10 классы общеобразовательной школы): 13.00.01. Дис...канд.пед.наук. – К., 1986. – 209с.

261. Хорошевский В. и др. Представление знаний в человеко-машинных и робототехнических системах = knowledge representation in man-machine systems and robotics: [Отчет в 4-х т.] – М., 1984.

262. Худик В.А. Исследования интеллекта личности: Образование аналогий. – К.: Здоровье, 1995.- 42 с.

263. Худик В.А. Исследования интеллекта личности: Таблицы Шульте. – К.: Здоровье, 1994.- 20 с.

264. Худик В.А. Исследования интеллекта личности: Исключение предметов. – К.: Здоровье, 1994.- 28 с.

265. Хуторской А.В. Эвристический тип образования: результаты научно-практического исследования // Педагогика. – №7, 1999. – С.15 – 22.

266. Цибко Г.Ю. Підвищення рівня теоретичної підготовки з інформатики на фізико-математичних факультетах педагогічних вузів: 13.00.02. Дис...канд.пед.наук. – К., 1998. – 200 с.

267. Цибко Г.Ю. Роль і місце поняття “база даних” у теоретичній і прикладній інформатиці. Комп’ютерно-орієнтовані системи навчання: Зб. наук. праць/ Редкол. Випуск 2. –К.: НПУ ім. М.П. Драгоманова. –2000. – С. 98 – 104.

268. Циганок М.М. Розв’язування фізичних задач з динамічною структурою змісту в сучасній загальноосвітній школі: 13.00.02. Дис...канд.пед.наук. – К., 2001. – 222 с.

269. Человеческий интеллект и его измерение. Теория и практика / Под ред. Ю. З. Гильбуха. – К.:РОВО “Укрвузполиграф”, 1992. – 133с.

270. Чередов И.М. Система форм организации обучения в советской общеобразовательной школе. – М.: Просвещение, 1987, 157 с.

271. Черняхівський В.В. Збірник задач з основ алгоритмізації. – Львів: Вид.-во наук.-техн. літ., 1997. – 195 с.

272. Чепрасова Т.І. Підвищення практичної значущості результатів навчання інформатики в старших класах середньої школи в умовах НІТН: 13.00.02. Дис…канд.пед.наук. – К., 1998., - 235 с.

273. Чудовский В.Э. Одаренность: дар или испытание. – М.: Знание, 1990. – 80с.

274. Шапіро С.І. Від алгоритмів до суджень. Експеримент по навчанню елементів математичного мислення. – М.: Сов. радио, 1973. – 16 с.

275. Шадриков В.Д. Психология деятельности и способности человека: Учебное пособие. 2-е изд., перераб. и доп. – М.: Издательская корпорация “Логос”, 1996. – 320с.

276. Шамова Т.И. Активизация учения школьников. – М.: Педагогика, 1982. – 208 с.

277. Школьный тест умственного развития (ШТУР): Форма А. – Харьков: РА, 1997. – 39 с.

278. Шлеер С., Меллор С. Объектно-ориентированный анализ: моделирование мира в состояниях. – К.: Диалектика, 1993. – 240 с.

279. Шлеер С., Меллор С. Нотация для объектно-ориентированного проектирования (OOD), независимая от языка программирования // Компьютерное обозрение. – Декабрь, 1993. – С. 3-9.

280. Щукина Г.И. Роль деятельности в учебном процессе: книга для учителя. – М.: Просвещение, 1986. – 144 с.

281. Эльконин Д.Б. Психология игры. – М.: Педагогика, 1978. – 301 с.

282. Эльконин Д.Б. Избранные психологические труды / Под. ред. В.В.Давыдова, В.П. Зинченко, - М.: Педагогика, 1989. – 560 с.

283. Юдаков С.Г. Формирование информационных умений и развитие творческих способностей учащихся // Информатика и образование. - №6, 2000.- С.70-73.

284. Якиманская И.С. Развивающее обучение. –М.: Педагогика, 1979. - 144 с.

285. Яковлев Н.М., Сохор А.М. Методика и техника урока в школе: В помощь начинающему учителю. – 3-е изд., перераб. И доп. – М.: Просвещение, 1985.– 208 с.

286. Ясінський А.М. Формування основ інформаційної культури школярів засобами інтегрованих завдань з інформатики: 13.00.02. Автореф…канд.пед.наук. – К., 2000. – 22 с.

287. Staats A.W., Burns G.L. Intelligence and child development: what intelligence is and how it is learned and functions. Genetic Psichol. Monograph. V.104. (1981). P.237-301.

288. Ftuerstein R. The theory of structural cognitive modifialibity, In: Presseisen B.Z. (Ed.). Learning and thinking styles: Classroom interfction. Washington. D.C.: Nat. Educat. Association. (1990). P.68-134.

289. Sternberg R.J. Inside Intelligence. Amer. Scientist. V.74(2). P.137-143. (1986).

290. Meili R. Structur der Intelligenz. Bern: Huber. (1981).

291. Guilford J.P. The nature of human intelligence. N.Y.: MC. Graw Hill. (1967).

292. Petroski H. To Engineer Is Human. St Martin’s Press: New York, 1985, p.40.

293. Thurstone L.L. The nature of intelligence. N.Y.: Harcourt. Brace and Company, Inc. (1924).

294. Sternberg R. J. Procedures for identifuing intellectual potential in the gifted: A perspective on alternative "Metaphors of Mind". In: Heller K.A. et al. (Eds.). International handbook of research and development of giftedness and talent. Oxford: Pergamon. (1993). P. 185-207.

295. Rentsch T. September 1982. Object-Oriented Programming. SIGPLAN Notices vol. 17(12), p.51.

296. Cardelly L., Wegner P. On Undestanding Types, Data Abstraction, and Polimophism. December 1985. ACM Computing Surveys vol.17(4), p.481.

297. Schmucker K. August 1986. Object-Oriented Languages for the Macintosh. Byte vol. 11(8), p.179.

Д О Д А Т К И

Додаток А

Програмна реалізація семантичної мережі задачі 2.8 засобами об’єктно-орієнтованого програмування на основі Object Pascal Delphi:

```
program Project1;
Uses Math;
Type
                TCosAl=class {Оголошення об’єкта S1}
FB,FC,FAl:real;
Constructor Create;
Procedure DoIt;
Property B:real read FB write FB;
Property C:real read FC write FC;
Property Al:real read FAl write FAl;
end;
TA=class {Оголошення об’єкта S2}
FA,FB,FAl:real;
Constructor Create;
Procedure DoIt;
Property A:real read FA write FA;
Property B:real read FB write FB;
Property Al:real read FAl write FAl;
end;
TCC=class {Оголошення об’єкта S3}
FA,FB,FC:real;
Constructor Create;
Procedure DoIt;
Property A:real read FA write FA;
Property B:real read FB write FB;
Property C:real read FC write FC;
end;
TBt=class {Оголошення об’єкта S4}
FAl,FBt:real;
Constructor Create;
Procedure DoIt;
Property Al:real read FAl write FAl;
Property Bt:real read FBt write FBt;
end;
TR=class {Оголошення об’єкта S5}
FC,FR:real;
Constructor Create;
Procedure DoIt;
Property C:real read FC write FC;
Property R:real read FR write FR;
End;
      TP=class {Оголошення об’єкта S6}
FA,FB,FC,FP:real;
Constructor Create;
Procedure DoIt;
Property A:real read FA write FA;
Property B:real read FB write FB;
Property C:real read FC write FC;
Property P:real read FP write FP;
End;
TSS1=class {Оголошення об’єкта S7}
FA,FB,FS:real;
Constructor Create;
Procedure DoIt;
Property A:real read FA write FA;
Property B:real read FB write FB;
Property S:real read FS write FS;
```

```
End;
TSS2=class {Оголошення об’єкта S8}
FS,FP,FRR:real;
Constructor Create;
Procedure DoIt;
Property S:real read FS write FS;
Property P:real read FP write FP;
Property RR:real read FRR write FRR;
End;
{Описання екземплярів об’єктів}
Var S1:TCosAl;S2:TA;S3:TCC;S4:TBt;S5:TR;S6:TP;S7:TSS1;S8:TSS2;
{Описання методів об’єктів}
Constructor TCosAl.Create; {Реалізація методів об’єкта S1}
Begin
Inherited Create;
FB:=0;FC:=0;FAl:=0;
End;
Procedure TCosAl.DoIt;
Begin
if (FB<>0)and (FC<>0)and(FAl=0)then FAl:=ArcCos(Fb/Fc);
if (FB<>0)and(FAl<>0)and(FC=0)then FC:=Fb/Cos(FAl);
if (Fc<>0)and(FAl<>0)and(FB=0)then FB:=Fc*Cos(Fal);
S2.B:=FB;S2.Al:=FAl;S3.C:=FC;S4.Al:=FAl;S5.C:=FC;
S6.C:=FC;S7.B:=FB;
End;
Constructor TA.Create; {Реалізація методів об’єкта S2}
Begin
Inherited Create;
FA:=0;FB:=0;FAl:=0;
End;
Procedure TA.DoIt;
Begin
if (FA<>0)and(FB<>0)and(FAl=0)then FAl:=ArcTan2(Fa,Fb);
if (FA<>0)and(FAl<>0)and(FB=0)then FB:=Fa/Tan(FAl);
if (FB<>0)and(FAl<>0)and(FA=0)then FA:=Fb*Tan(FAl);
S3.A:=FA;S6.A:=FA;S7.A:=FA;S1.b:=FB;S3.B:=FB;S6.B:=FB;S7.B:=FB;
S1.Al:=FAl;S4.Al:=FAl;
End;
Constructor TCC.Create; {Реалізація методів об’єкта S3}
Begin
Inherited Create;
FA:=0;FB:=0;FC:=0;
End;
Procedure TCC.DoIt;
Begin
if (FA<>0)and(FB<>0)and(FC=0)then FC:=Sqrt(Sqr(FA)+Sqr(FB));
if (FA<>0)and(FC<>0)and(FB=0)then FB:=Sqrt(Sqr(FC)-Sqr(FA));
if (FB<>0)and(FC<>0)and(FA=0)then FA:=Sqrt(Sqr(FC)-Sqr(FB));
S2.A:=FA;S6.A:=FA;S7.A:=FA;S1.B:=FB;S2.B:=FB;S6.B:=FB;S7.B:=FB;
S1.C:=FC;S5.C:=FC;S6.C:=FC;
End;
Constructor TBt.Create; {Реалізація методів об’єкта S4}
Begin
Inherited Create;
FAl:=0;FBt:=0;
End;
Procedure TBt.DoIt;
Begin
If (FAl<>0)and(FBt=0) then FBt:=pi/2-FAl;
If (FBt<>0)and(FAl=0) then FAl:=pi/2-FBt;
S1.Al:=FAl;S2.Al:=FAl;
End;
Constructor TR.Create; {Реалізація методів об’єкта S5}
Begin
```

```
Inherited Create;
FC:=0;FR:=0;
End;
Procedure TR.DoIt;
Begin
If (FC<>0)and(FR=0)then FR:=FC/2;
If (FR<>0)and(FC=0)then FC:=FR*2;
S1.C:=FC;S3.C:=FC;S6.C:=FC;
End;
Constructor TP.Create; {Реалізація методів об’єкта S6}
Begin
Inherited Create;
FA:=0;FB:=0;FC:=0;FP:=0;
End;
Procedure TP.DoIt;
Begin
If (FA<>0)and(FB<>0)and(FC<>0)and(FP=0)then FP:=(FA+FB+FC)/2;
If (FA<>0)and(FB<>0)and(FP<>0)and(FC=0)then FC:=FP*2-(FA+FB);
If (FA<>0)and(FC<>0)and(FP<>0)and(FB=0)then FB:=FP*2-(FA+FC);
If (FB<>0)and(FC<>0)and(FP<>0)and(FA=0)then FA:=FP*2-(FC+FB);
S2.A:=FA;S3.A:=FA;S7.A:=FA;S1.B:=FB;S2.B:=FB;S3.B:=FB;s7.B:=FB;
S1.C:=FC;S3.FC:=FC;S5.C:=FC;S6.P:=FP;S8.P:=FP;
End;
Constructor TSS1.Create; {Реалізація методів об’єкта S7}
Begin
Inherited Create;
FA:=0;FB:=0;FS:=0;
End;
Procedure TSS1.DoIt;
Begin
If (FA<>0)and(FB<>0)and(FS=0)then FS:=FA*FB/2;
If (FA<>0)and(FS<>0)and(FB=0)then FB:=FS*2/FA;
If (FB<>0)and(FS<>0)and(FA=0)then FA:=FS*2/FA;
S8.S:=FS;S2.A:=FA;S3.A:=FA;S6.A:=FA;S1.B:=FB;S2.B:=FB;
S3.B:=FB;S6.B:=FB;
End;
Constructor TSS2.Create; {Реалізація методів об’єкта S8}
Begin
Inherited Create;
FS:=0;FP:=0;FRR:=0;
End;
Procedure TSS2.DoIt;
      Begin
If (FS<>0)and(FP<>0)and(FRR=0)then FRR:=FS/FP;
If (FS<>0)and(FRR<>0)and(FP=0)then FP:=FS/FRR;
If (FRR<>0)and(FP<>0)and(FS=0)then FS:=FP*FRR;
S7.S:=FS;S6.P:=FP;
End;
Begin {Головна програма}
S1:=TCosAl.Create;S2:=TA.Create;S3:=TCC.Create;S4:=TBt.Create;
S5:=TR.Create;S6:=TP.Create;S7:=TSS1.Create;S8:=TSS2.Create;
S2.A:=10;S2.B:=6;
While S8.RR=0 do
Begin
S2.DoIt;S1.DoIt;S3.DoIt;S4.DoIt;S5.DoIt;S6.DoIt;S7.DoIt;S8.DoIt;
End;
Writeln(S8.RR:7:3);
S1.Free;S2.Free;S3.Free;S4.Free;S5.Free;S6.Free;S7.Free;S8.Free;
end.
```

Додаток Б

Опрацювання стандартних компонент Delphi у процесі створення навчального проекту “Політ бджоли”

1. Заготовки вчителя

Для кращого візуального спостереження за процесами, що відбуватимуться, коли будуть траплятися запропоновані події, доцільно, відтворювати переміщення попередньо створених графічних об’єктів. Ми пропонуємо використовувати різні зображення бджоли, які відповідають основним напрямкам її руху на фоні лучного краєвиду. Учням для подальшого опрацювання надається початкова структура проекту, яка включає форму Form1 з розміщеними на ній компонентами (рис.Б.1): PaintBox1, Button1 та Button2. Об’єкт PaintBox1 використовується для створення зображень моделювання переміщення бджоли. Кнопка Button1 (має Caption=’Begin’) призначена для виведення на PaintBox1 зображення фону, кнопка Button2 (Caption=’End’) використовується для переривання виконання проекту (Pro1). Код програмного модуля D1.pas, який відповідає формі Form1:

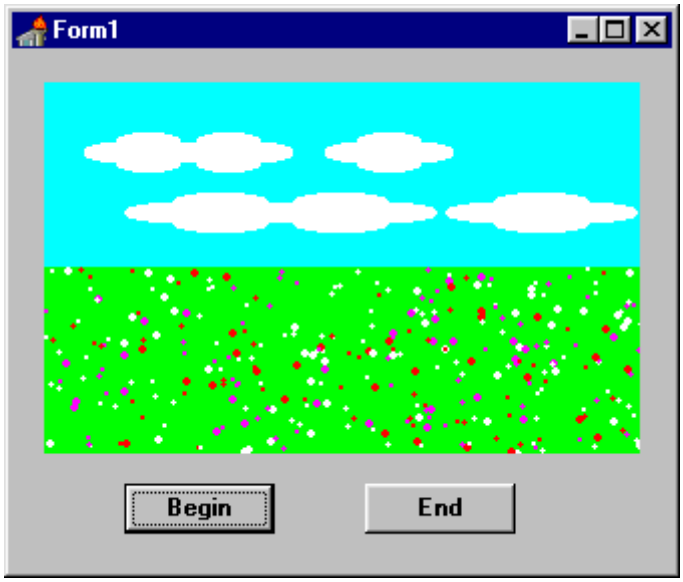


Рис.Б.1. Зображення форми Form1.

```
interface
uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms,
Dialogs,
  StdCtrls, ExtCtrls;
```

```
type
  TForm1 = class(TForm)
    PaintBox1: TPaintBox;
    Button1: TButton;
    Button2: TButton;
    procedure Button1Click(Sender: TObject);
    procedure Button2Click(Sender: TObject);
  private
  procedure Begining;
    { Private declarations }
  public
    { Public declarations }
  end;
var
  Form1: TForm1;
  Bt1,bt2:TBitmap;
  xx,yy:integer;
implementation

{$R *.DFM}
{$I pchela.pas}
procedure TForm1.Button1Click(Sender: TObject);
begin
Begining;
end;
procedure TForm1.Button2Click(Sender: TObject);
begin
Bt1.Free;
Application.Terminate;
end;
end.
```

Текст програмного модуля містить три процедури. Процедура Begining призначена для створення зображення фону на канві бітової матриці Bt1 та його подальшого виведення в компоненті PaintBox1, Button1Click(Sender: TObject) – для активізації наведеного вище зображення, при одноразовому короткочасному натисненні на об’єкті Button1 лівою клавішею миші. В процедурі Button2Click(Sender: TObject) для переривання виконання програми використовується метод Application.Terminate, а для звільнення пам’яті, що була задіяна для побудови графічного зображення на бітовій матриці Bt1 – метод Bt1.Free.

Докладніше з методами створення графічних зображень учні будуть ознайомлені пізніше. На початку створення проекту можна надати учням тільки мінімально необхідну інформацію про задані процедури побудови графічних зображень, які знаходяться у файлі pchela.pas. Цей файл як

фрагмент програмного коду підключається до модуля D1.pas у вигляді рядка {$I pchela.pas}.

Зміст файлу pchela.pas:

```
Procedure Oblako(x,y,k:integer);
begin
Bt1.Canvas.Pen.Color:=ClWhite;
Bt1.Canvas.Brush.Color:=clWhite;
Bt1.Canvas.Ellipse(x,y,x+k*16,y+10);
Bt1.Canvas.Ellipse(x+k*16,y,x+k*32,y+10);Bt1.Canvas.Ellipse(x+k*8,y-
5,x+k*24,y+5);
Bt1.Canvas.Ellipse(x+k*8,y+5,x+k*24,y+15); end;
procedure nebo(xx,yy:integer);
var i,x1,y1,h1,h2,k,h,hh:integer;
begin
x1:=xx; y1:=yy div 2;
Bt1.Canvas.Pen.Color:=ClAqua;
Bt1.Canvas.Brush.Color:=clAqua;
Bt1.Canvas.Rectangle(0,0,xx,yy div 2);
h1:=20;h2:=30;h:=30;hh:=20;
x1:=0;y1:=0;k:=2;
for i:=1 to xx div 80 do
begin
Oblako(x1+h1,y1+h2,k);
k:=k+1;h1:=h1+i*hh;h2:=h2+h;
Oblako(x1+h1,y1+h2,k);
k:=k-1;h1:=h1+i*hh;h2:=h2-h;
end; end;
procedure pole(xx,yy:integer);
var i,x,y,c,l,h:integer;
begin
Bt1.Canvas.Pen.Color:=ClLime;
Bt1.Canvas.Brush.Color:=clLime;
Bt1.Canvas.Rectangle(0,yy div 2,xx,yy);
     Randomize;
for i:=1 to xx*2  do
begin
x:=Random(xx);y:=yy div 2+Random(yy div 2);
l:=Random(5);h:=random(50)mod 3;
case h of
0: c:=clWhite;
1: c:=clred;
2: c:=clfuchsia;
end;
Bt1.Canvas.Pen.Color:=c;
Bt1.Canvas.Brush.Color:=c;
Bt1.Canvas.Ellipse(x,y,x+l,y+l);
end;
end;
procedure TForm1.Begining;
Begin
xx:=PaintBox1.Width; yy:=PaintBox1.Height;
Bt1:=TBitmap.Create;
Bt1.Height:=yy;Bt1.Width:= xx;
Bt1.Canvas.Pen.Color:=ClBtnFace;
Bt1.Canvas.Brush.Color:=clBtnFace;
Bt1.Canvas.Rectangle(0,0,xx,yy);
nebo(xx,yy); pole(xx,yy);
PaintBox1.Canvas.CopyRect(Rect(0,0,xx,yy),Bt1.Canvas,Rect(0,0,xx,yy));
Bt2:=Bt1;
end;
Procedure Golova(x,y:integer);
Begin
```

```
Bt1.Canvas.Pen.Color:=clBlack;
Bt1.Canvas.Brush.Color:=clMaroon;
Bt1.Canvas.Ellipse(x,y,x+25,y+25);
Bt1.Canvas.Brush.Color:=clWhite;
Bt1.Canvas.Ellipse(x+2,y+5,x+12,y+20);
Bt1.Canvas.Ellipse(x+13,y+5,x+23,y+20);
Bt1.Canvas.Brush.Color:=clBlack;
Bt1.Canvas.Ellipse(x+5,y+11,x+9,y+17);
Bt1.Canvas.Ellipse(x+16,y+11,x+20,y+17);Bt1.Canvas.Pen.Width:=3;
Bt1.Canvas.MoveTo(x+12,y+22);
Bt1.Canvas.LineTo(x+12,y+27);
Bt1.Canvas.Pen.Width:=1;
Bt1.Canvas.MoveTo(x+12,y+3);
Bt1.Canvas.LineTo(x+9,y-5);
Bt1.Canvas.MoveTo(x+14,y+3);
Bt1.Canvas.LineTo(x+17,y-5);
Bt1.Canvas.Ellipse(x+5,y-7,x+9,y-2);
Bt1.Canvas.Ellipse(x+17,y-7,x+21,y-2);
     end;
procedure Pchelka(x,y,k,f:integer);
{1-right,2-left,3-up-right,4-up-left,5-down-right,6-down-left}
var x1,y1,h,j,i,j1,h1,h2,h3,j3:integer;
begin
Bt1.Canvas.Pen.Color:=clBlack;
case k of
1: begin x1:=x-25; j:=1; j1:=0; y1:=y;       h1:=0; h2:=0;end;
2: begin x1:=x+30; j:=-1; j1:=0; y1:=y; h1:=23; h2:=0;end;
3: begin x1:=x-22; j:=1; j1:=1; y1:=y+20; h1:=-9; h2:=-13;end;
4: begin x1:=x+27; j:=-1; j1:=1; y1:=y+20; h1:=33; h2:=-13;end;
5: begin x1:=x-20; j:=1; j1:=-1; y1:=y-20; h1:=10; h2:=-8;end;
6: begin x1:=x+20; j:=-1; j1:=-1; y1:=y-20; h1:=15; h2:=-8;end;
end;
h:=0;
for i:=1 to 2 do
begin
Bt1.Canvas.Brush.Color:=clMaroon;
Bt1.Canvas.Ellipse(x1+j*h,y1-j1*h, x1+20+j*h, y1+20-j1*h);
Bt1.Canvas.Brush.Color:=clOlive;
Bt1.Canvas.Ellipse(x1+j*(h+5), y1-j1*(h+5), x1+20+j*(h+5),y1+20-j1*(h+5));
h:=h+12;
     end;
     h:=0;
     For i:=1 to 4 do
begin
Bt1.Canvas.MoveTo(x1+j*h+5,y1-j1*h+15);
Bt1.Canvas.LineTo(x1+j*h+5,y1-j1*h+25);
h:=h+5; end;
case f of
0: begin
Bt1.Canvas.Brush.Color:=clWhite;
Bt1.Canvas.Ellipse(x-j*8+h1,y-18-  j1*h2,x+j*7+h1,y+3-j1*h2);
Bt1.Canvas.Brush.Color:=clAqua;
Bt1.Canvas.Ellipse(x-j*13+h1,y-20-j1*h2,x+j*2+h1,y+3-j1*h2);   end;
1: begin
Bt1.Canvas.Brush.Color:=clAqua;
Bt1.Canvas.Ellipse(x-j*8+h1,y-18-j1*h2,x+j*7+h1,y+3-j1*h2);
Bt1.Canvas.Brush.Color:=clWhite;
Bt1.Canvas.Ellipse(x-j*13+h1,y-20-j1*h2,x+j*2+h1,y+3-j1*h2);
end; end;
Golova(x,y); end;
procedure PchelkaRight(x,y,f:integer);
var i:integer;
begin Pchelka(x,y,1,f);end;
procedure PchelkaLeft(x,y,f:integer);
```

```
begin Pchelka(x,y,2,f); end;
procedure PchelkaUpRight(x,y,f:integer);
begin Pchelka(x,y,3,f); end;
procedure PchelkaUpLeft(x,y,f:integer);
begin Pchelka(x,y,4,f); end;
procedurePchelkaDounRight(x,y,f:integer);
begin Pchelka(x,y,5,f);end;
procedure PchelkaDounLeft(x,y,f:integer);
begin Pchelka(x,y,6,f);end;
```

До основних процедур файлу pchela.pas належать наступні: nebo(xx,yy:integer), pole(xx,yy:integer) призначені для створення зображення лучного краєвиду, їх вхідні параметри xx, yy позначають розміри графічного об’єкту PaintBox1 на формі Form1; PchelkaRight(x,y,f:integer), PchelkaLeft(x,y,f:integer) - для створення різних зображень бджоли у процесі її польоту. Параметри x,y – координати розташування бджоли на канві графічного об’єкту, змінна f приймає значення 0 або 1, використовуватиметься для імітації її польоту.

Спочатку можна надати учням можливість завантажити проект Pro1 і зробити зміни в зовнішньому вигляді розміщених на заданій формі компонент: поміняти їх розміри, місце розташування, колір (Color), заголовок (Caption), шрифт (Font) тощо. Незмінними бажано залишити імена об’єктів (Name). Від цього буде залежати успіх подальшої колективної роботи над проектом.

2. Робота з формами

Форму Form1, призначену для виведення результуючих зображень доповнювати новими компонентами ми не будемо. Для цього використаємо іншу, робочу форму. А, щоб дві форми одночасно співіснували на екрані не перекриваючись, вертикально розмістимо їх в межах однієї батьківської як зображено на малюнку (рис.Б.2).

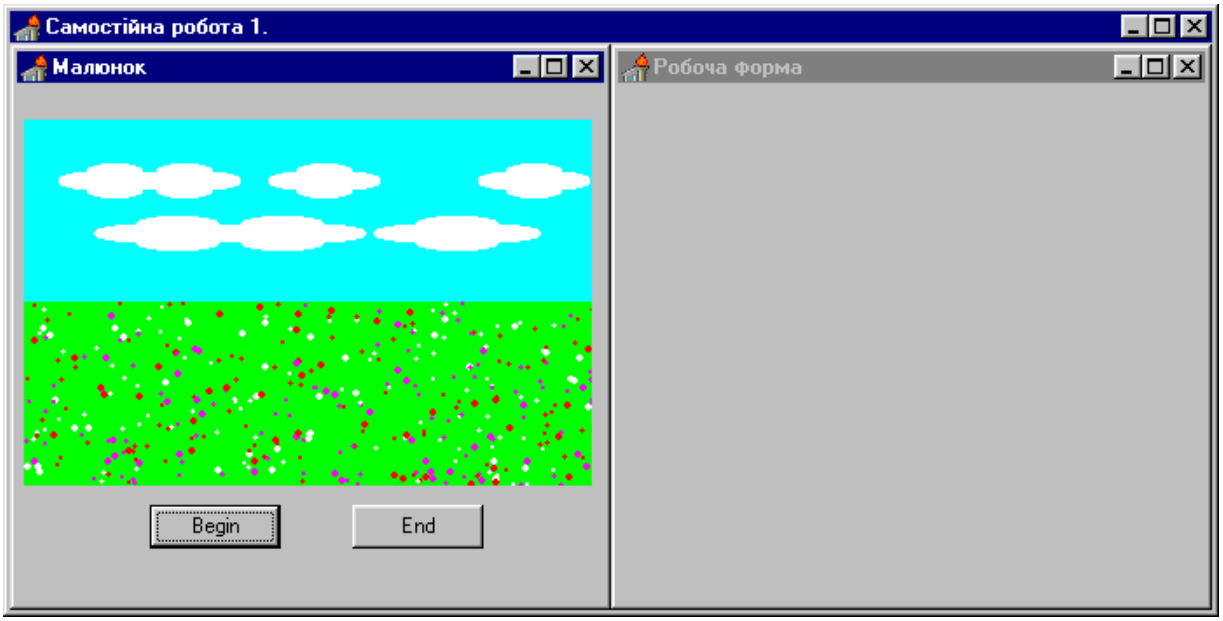


Рис.Б.2. Створення MDI форми.

Для цього необхідно викликати дві нові форми (File ⇒ New Form) Form2 та Form3. Форму Form2 зробити батьківською для заданої форми Form1 та нової Form3. У зв'язку з цим властивості форми Form2 FormStyle як батьківській в інспекторі об'єктів присвоїти значення fsMDIForm, а для форм Form1 та Form3 – fsMDIChild. При реалізації процедури висвітлення батьківської форми (procedure FormShow(Sender: TObject)) в її програмному модулі MainForm застосувати метод Tile, який розміщує дочірні форми за принципом, заданим властивістю TileMode (tbVertical або tbHorizontal). Текст програмного коду:

```
unit MainForm;
interface
uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms,
Dialogs;
type
  TForm2 = class(TForm)
    procedure FormShow(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;
var
  Form2: TForm2;
implementation
{$R *.DFM}
procedure TForm2.FormShow(Sender: TObject);
begin
```

```
TileMode:=tbVertical;
Tile;
end;
end.
```

При додаванні нових форм до проекту, хоч його програмний код доповнюється автоматично, реалізація завдання потребує втручання програміста у написання програмного коду проекту. Треба пам’ятати, що батьківська форма повинна створюватись перед дочірніми, спочатку створюється ліва форма, а потім права (якщо використовуються тільки дві дочірні форми). Тому в програмному коді проекту необхідно виконати перестановки відповідних рядків створення окремих форм. В результаті зміст файлу проекту набуде вигляду**:**

```
program Pro2;
uses
  Forms,
  D1 in 'D1.pas' {Form1},
  MainForm in 'MainForm.pas' {Form2},
  WorkForm in 'WorkForm.pas' {Form3};
{$R *.RES}
begin
  Application.Initialize;
  Application.CreateForm(TForm2, Form2);
  Application.CreateForm(TForm3, Form3);
  Application.CreateForm(TForm1, Form1);
  Application.Run;
                                  end.
```

При необхідності учні мають змогу завантажити проміжну реалізацію проекту Pro2 і з її допомогою дослідити інші властивості і методи опрацювання форм (Caption, Color, BorderStyle, WindowState, Cursor тощо). Результати дослідження обговорюються в класі (передбачаємо, наприклад, учні помітять, що дочірні форми не можна створити без бордюру, а також зроблять висновки стосовно розміщення на формі піктограм, які відповідають за можливість роботи з формою тощо).

3. Організація вибору

Для реалізації операції вибору середовище візуального програмування Delphi надає кілька засобів, які слід розглянути, а також

оцінити раціональність їх використання в проекті. Серед таких засобів назвемо такі:

1. Компонента ListBox – стандартне вікно списку Windows, яке дозволяє користувачу вибрати необхідний пункт. Окремий елемент (Items) заноситься до списку методом Add. Фрагмент програмного коду реалізації зазначеної дії: ListBox1.Items.Add('Ноий елемент'). Ініціювати вибір можна за допомогою вже відомих подій (OnClick, OnDblClick тощо). Про результат зробленого вибору свідчить значення номеру відповідного пункту (ListBox1.ItemIndex). Слід зауважити, що нумерація пунктів списку починається з 0.
2. Компонента ComboBox поєднує функції ListBox та Edit. Користувач може або ввести назву пункту з клавіатури, або вибрати його з списку. Методи роботи з цим компонентом аналогічні тим, що розглядалися вище для ListBox.
3. Компоненти RadioButton надають користувачу набір альтернатив, з яких вибирається єдина. Невеликий за кількістю пунктів список можна реалізувати набором відповідної кількості радіокнопок. Якщо деяка радіокнопка вибрана, її властивість RadioButton.Checked приймає значення true. Якщо ми передбачаємо, що в якості початкової умови, деяка радіокнопка повинна бути вибраною, то її властивості Checked необхідно присвоїти true.
4. Компоненти CheckBox дозволяють організувати вибір кількох альтернативних варіантів. Про результат вибору повідомляє значення властивості State (CheckBox1.State=cbChecked), в протилежному випадку CheckBox1.State=cbUnChecked або CheckBox1.State=cbGrayed.
5. Компонента RadioGroup використовується для створення єдиної групи радіокнопок. У зв'язку з цим зникає необхідність в перевірці властивості Checked кожної кнопки. Про здійснений вибір свідчить значення

властивості RadioGroup1.ItemIndex – номер відповідної кнопки в групі. Якщо деяку радіокнопку треба зробити поміченою за замовчання при запуску програми, в інспекторі об’єктів властивості ItemIndex компоненти RadioGroup присвоїти значення номеру зазначеної кнопки. Для формування радіокнопок в компоненті RadioGroup використовується текстовий редактор, в який можна ввійти за вибором властивості TString.

Розглянемо приклад реалізації операції вибору в проекті Pro3 (рис.Б.3). В наведеному проекті робоча форма містить компоненти:

ListBox - для організації вибору зображення бджоли в залежності від напрямку її руху. Вводити текст з клавіатури доцільніше при великій кількості пунктів списку, чого не передбачається в нашому проекті. Тому доцільно спочатку зупинитись на використанні саме цієї компоненти.

Image – для виведення зображень можливих рухів бджоли на робочій формі.

Button – для виведення вибраного зображення бджоли на фоні лучного краєвиду на компоненті PaintBox1 лівої форми.

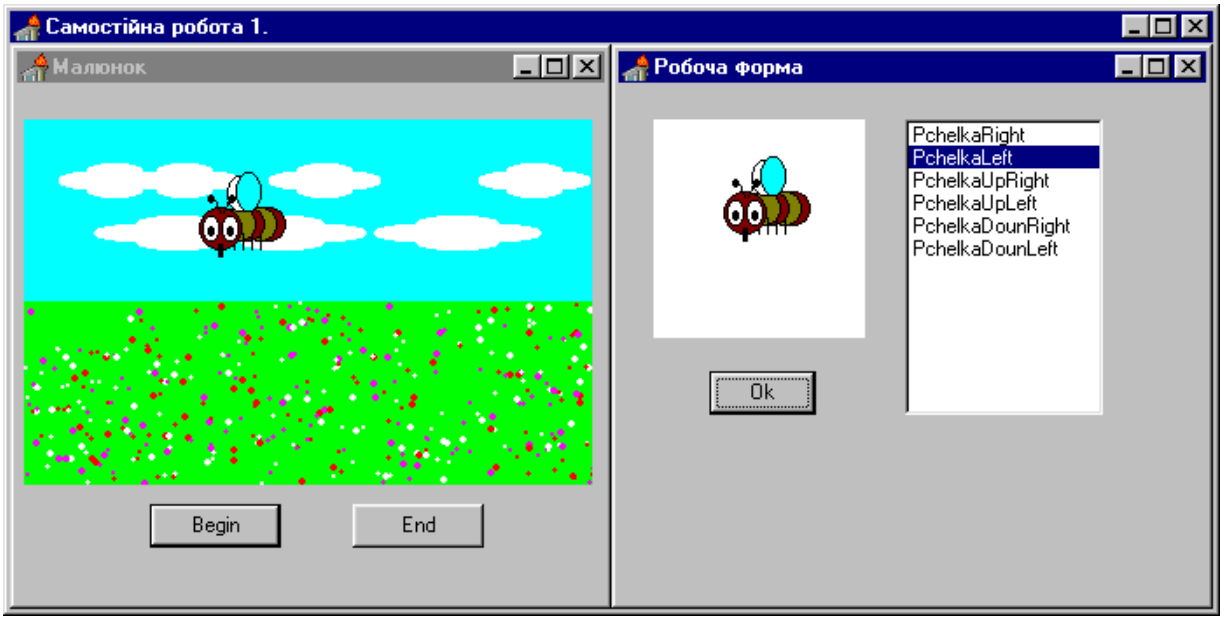


Рис.Б.3. Реалізація проекту Pro3.

Зміст програмного модуля робочої форми:

```
unit WorkForm;interface
uses
Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
StdCtrls, ExtCtrls;
type
  TForm3 = class(TForm)
    Image1: TImage;
    ListBox1: TListBox;
    Button1: TButton;
    procedure ListBox1DblClick(Sender: TObject);
    procedure FormShow (Sender: TObject);
    procedure Button1Click(Sender: TObject);
  private     { Private declarations }
  public     { Public declarations }
  end;
var
  Form3: TForm3;
  Bt1:TBitmap;
  xx1,yy1,h:integer;
implementation
uses D1;
{$R *.DFM}
{$I pchela.pas}
procedure Fon;
begin
xx1:=Form1.PaintBox1.Width;
yy1:=Form1.PaintBox1.Height;
Bt1:=TBitmap.Create;
Bt1.Height:=yy1;
Bt1.Width:= xx1;
Bt1.Canvas.Pen.Color:=ClBtnFace;
Bt1.Canvas.Brush.Color:=clBtnFace;
Bt1.Canvas.Rectangle(0,0,xx1,yy1);
nebo(xx1,yy1);
pole(xx1,yy1);
Form1.PaintBox1.Canvas.CopyRect(Rect(0,0,xx1,yy1),Bt1.Canvas,
Rect(0,0,xx1,yy1));
end;
procedure LitlPicture(k:integer);
var xx,yy,x,y:integer;
begin
xx:=Form3.Image1.Width;yy:=Form3.Image1.Height;
Bt1:=TBitmap.Create;
Bt1.Height:=yy;
Bt1.Width:= xx;
Bt1.Canvas.Pen.Color:=ClWhite;
Bt1.Canvas.Brush.Color:=clWhite;
Bt1.Canvas.Rectangle(0,0,xx,yy);
case k of
1: begin x:=xx div 2;y:=yy div 3; PchelkaRight(x,y,0);end;
2: begin x:=xx div 3;y:=yy div 3; PchelkaLeft(x,y,0);end;
3: begin x:=xx div 2;y:=yy div 3; PchelkaUpRight(x,y,0);end;
4: begin x:=xx div 3;y:=yy div 3; PchelkaUpLeft(x,y,0);end;
5: begin x:=xx div 2;y:=yy div 2; PchelkaDounRight(x,y,0);end;
6: begin x:=xx div 3;y:=yy div 2; PchelkaDounLeft(x,y,0);end;
end;
end;
procedure TForm3.ListBox1DblClick (Sender: TObject);
var i:integer;
begin
i:=ListBox1.ItemIndex+1;
case i of
1: LitlPicture(i);
```

```
2: LitlPicture(i);
3: LitlPicture(i);
4: LitlPicture(i);
5: LitlPicture(i);
6: LitlPicture(i);
end;
Image1.Picture.Bitmap:=Bt1;
Bt1.Free;
h:=i;
end;
procedure TForm3.FormShow(Sender: TObject);
begin
{Занесення списка для вибору до компонента ListBox1}
ListBox1.Items.Add('PchelkaRight');
ListBox1.Items.Add('PchelkaLeft');
ListBox1.Items.Add('PchelkaUpRight');
ListBox1.Items.Add('PchelkaUpLeft');
ListBox1.Items.Add('PchelkaDownRight');
ListBox1.Items.Add('PchelkaDownLeft');
end;
procedure TForm3.Button1Click (Sender: TObject);
var x,y:integer;
begin
Fon;
x:=100;y:=50;
case h of
1: PchelkaRight(x,y,0);
2: PchelkaLeft(x,y,0);
3: PchelkaUpRight(x,y,0);
4: PchelkaUpLeft(x,y,0);
5: PchelkaDounRight(x,y,0);
6: PchelkaDounLeft(x,y,0);
end;
Form1.PaintBox1.Canvas.CopyRect (Rect(0,0,xx1,yy1), Bt1.Canvas, Rect (0,0,
xx1,yy1));
Bt1.Free;
end;
end.
```

До програмного коду включені процедури:

Fon – для зображення лучного краєвиду на компоненті PaintBox1 лівої форми. Змінні xx1, yy1 використовуються для позначення розмірів малюнка, який формується на канві бітової матриці Bt1. Вони відповідають розмірам компонента PaintBox1 лівої форми.

LitlPicture(k:integer) – формує зображення бджоли в залежності від її руху для виведення в компоненті Image. Змінні xx, yy використовуються для позначення розмірів малюнка і дорівнюють розмірам компонента Image. Для формування зображень використовуються відповідні процедури (PchelkaRight(x,y,0); PchelkaLeft(x,y,0); PchelkaUpRight(x,y,0) тощо), які знаходяться в файлі pchela.pas.

Наявність процедури TForm3.ListBox1DblClick (Sender: TObject) свідчить про те, що вибір необхідного пункту здійснюється за короткочасним подвійним натисненням лівої кнопки миші. За значенням властивості ListBox1.ItemIndex, яке передається як значення вхідного параметру відбувається звернення до вище зазначеної процедури LitlPicture(k:integer).

Заповнення списку компонента ListBox відбувається при появі робочої форми на екрані. Здійснення цієї операції відбувається завдяки процедурі TForm3.FormShow(Sender: TObject).

Перенесення зображення бджоли на малюнок з фоном у визначені координати (100,50) компоненти PaintBox лівої форми здійснюється за натисненням кнопки “OK”, про що свідчить наявність в програмному коді процедура TForm3.Button1Click (Sender: TObject).

Цей програмний код проекту Pro3 надається учням для ознайомлення, після чого їм пропонується реалізувати операцію вибору за допомогою інших розглянутих засобів. Таке завдання потребує початкового аналізу раціональності в написанні програмного коду при використанні наведених компонент реалізації вибору.

Як вже зазначалося вище, використання компоненти ComboBox доцільне при значній кількості компонент списку, чого не передбачається в нашому проекті, а методи роботи аналогічні тим, які застосовувались при опрацюванні компоненти ListBox. Тому доцільно проаналізувати наступні три компоненти.

Застосування в проекті окремих радіокнопок (компонента RadioButton) передбачає наявність в програмному коді кількох (в нашому прикладі шести) умовних операторів перевірки активності відповідної кнопки:

```
If RadioButton1.Checked then ..........................;
If RadioButton2.Checked then ..........................;
.....................................................
If RadioButton6.Checked then ..........................;
```

Використання з метою вибору єдиної альтернативи компоненти CheckBox потребує аналогічного підходу, але завдання ускладнюється формуванням складної умови в умовному операторі**:**

```
     If (CheckBox1.State = cbChecked) and (CheckBox2.State = cbUnChecked) and
... and     CheckBox6.State=cbUnChecked) then .........;
     ..........................................................................
      If (CheckBox1.State = cbUnChecked) and (CheckBox2.State = cbUnChecked)
and ... and     CheckBox6.State=cbUnChecked) then .........;
```

Як бачимо, наведені підходи призводять до значного нагромадження програмного коду. Раціональнішим в цьому випадку є використання компоненти RadioGroup. Реалізація опрацювання операції вибору в програмному коді аналогічна проведеній за допомогою компоненти ListBox в попередньому проекті за допомогою оператора переключення (case of) за значенням властивості RadioGroup1.ItemIndex.

У випадку, коли хтось з учнів не зміг виконати завдання із заміни однієї компоненти іншою, йому доцільно продемонструвати роботу і реалізацію проекту Pro4a, зовнішній вигляд форми якого наводить на малюнку (рис.Б.4).

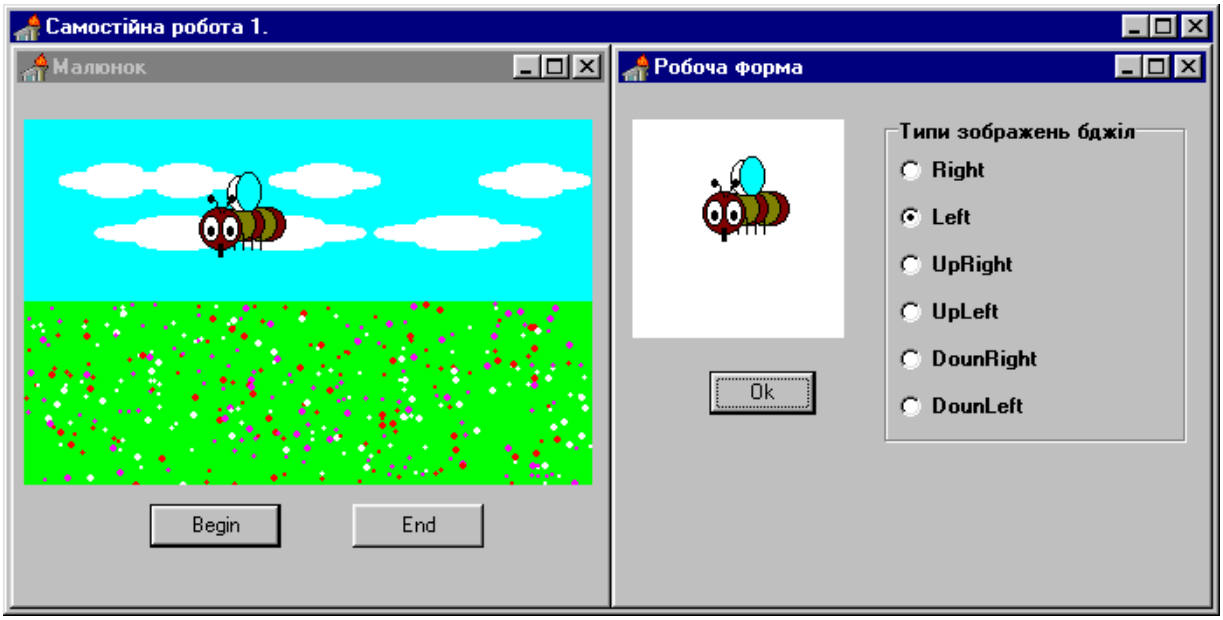


Рис.Б.4. Реалізація проекту Pro4a.

Повністю зміст нового програмного модуля робочої форми ми наводити не будемо. Він відрізняється від попереднього тільки наявністю однією процедури: `procedure TForm3.RadioGroup1Click(Sender: TObject);`

```
var i:integer;
begin
i:=RadioGroup1.ItemIndex+1;
case i of
1: LitlPicture(i);
2: LitlPicture(i);
3: LitlPicture(i);
4: LitlPicture(i);
5: LitlPicture(i);
6: LitlPicture(i);
end;
Image1.Picture.Bitmap:=Bt1;
Bt1.Free;
h:=i;
end;
```

Приклад застосування двох компонент CheckBox наведений в проекті Pro5 для вибору типу зображення в компоненті Image: білий квадрат – при відключенні обох компонент, зображення бджоли – при включенні першої і відключенні другої, квітки – при відключенні першого та включенні другого, бджоли на фоні квітки - при включенні обох компонент. Один з варіантів стану компонент CheckBox наведений на малюнку (рис.Б.5):

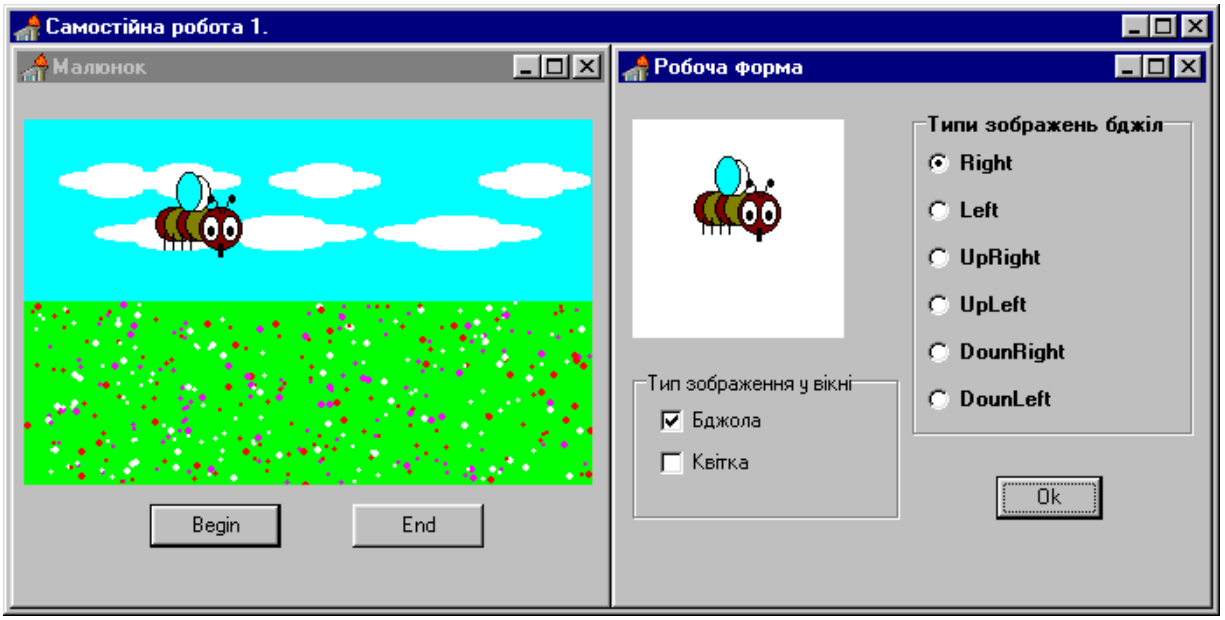


Рис.Б.5. Реалізація проекту Pro5.

Зміст програмного модуля робочої форми значно збільшується за рахунок процедур реалізації додаткових зображень та опрацювання компонент CheckBox, тому наводимо його повністю: unit WorkForm;

```
interface
uses   Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms,
Dialogs,   StdCtrls, ExtCtrls;
type
TForm3 = class(TForm)
Image1: TImage;
Button1: TButton;
RadioGroup1: TRadioGroup;
GroupBox1: TGroupBox;
CheckBox1: TCheckBox;
CheckBox2: TCheckBox;
procedure Button1Click(Sender: TObject);
procedure RadioGroup1Click (Sender: TObject);
procedure FormShow(Sender: TObject);
private     { Private declarations }
public     { Public declarations }
 end;
var
           Form3: TForm3;
       Bt1,Bt2:TBitmap;
       xx1,yy1,h,n,xx,yy,x,y:integer;
       x0,y0,xx0,yy0,l:integer;
       fl:boolean;
     implementation

     uses D1;
     {$R *.DFM}
     {$I pchela.pas}
     procedure Fon;
     begin
     xx1:=Form1.PaintBox1.Width;yy1:=Form1.PaintBox1.Height;
     Bt1:=TBitmap.Create;
     Bt1.Height:=yy1;
     Bt1.Width:= xx1;
     Bt1.Canvas.Pen.Color:=ClBtnFace;
     Bt1.Canvas.Brush.Color:=clBtnFace;
     Bt1.Canvas.Rectangle(0,0,xx1,yy1);
     nebo(xx1,yy1);
     pole(xx1,yy1);
     Form1.PaintBox1.Canvas.CopyRect(Rect(0,0,xx1,yy1),Bt1.Canvas,
     Rect(0,0,xx1,yy1));
     end;
     procedure Init;
     begin
     xx:=Form3.Image1.Width; yy:=Form3.Image1.Height;
     Bt1:=TBitmap.Create;
     Bt1.Height:=yy;
     Bt1.Width:= xx;
     Bt1.Canvas.Pen.Color:=ClWhite;
     Bt1.Canvas.Brush.Color:=clWhite;
     Bt1.Canvas.Rectangle(0,0,xx,yy);
     end;
     procedure LitlPicture(k:integer);
     begin
     Init;
     case k of
     1: begin x:=xx div 2;y:=yy div 3;PchelkaRight(x,y,0);end;
```

```
     2: begin x:=xx div 3;y:=yy div 3;PchelkaLeft(x,y,0);end;
     3: begin x:=xx div 2;y:=yy div 3;PchelkaUpRight(x,y,0);end;
     4: begin x:=xx div 3;y:=yy div 3;PchelkaUpLeft(x,y,0);end;
     5: begin x:=xx div 2;y:=yy div 2;PchelkaDounRight(x,y,0);end;
     6: begin x:=xx div 3;y:=yy div 2;PchelkaDounLeft(x,y,0);end;
     end;
     end;
     procedure Clear;
     begin
     Init;
        Form3.Image1.Picture.Bitmap:=Bt1;
             Bt1.Free;
     end;
     procedure Kvit;
     begin xx0:=Form3.Image1.Width;yy0:=Form3.Image1.Height;
      x0:=xx0 div 2{-10}; y0:=yy0 div 2;
          Init;
Bt1.Canvas.Pen.Color:=ClBlack;
Bt1.Canvas.Brush.Color:=clYellow;
Bt1.Canvas.Ellipse(x0,y0,x0+20,y0+20);Bt1.Canvas.Pen.Color:=ClBlack;
Bt1.Canvas.Brush.Color:=clFuchsia;
Bt1.Canvas.Ellipse(x0-5,y0-30,x0+25,y0);
Bt1.Canvas.Ellipse(x0+20, y0-5, x0+50, y0+25);
Bt1.Canvas.Ellipse(x0-30,y0-5,x0,y0+25);
Bt1.Canvas.Ellipse(x0-5,y0+20, x0+25, y0+50);
Form3.Image1.Picture.Bitmap:=Bt1;         Bt1.Free;
end;
procedure TForm3.RadioGroup1Click (Sender: TObject);
     var i:integer;
     begin
i:=RadioGroup1.ItemIndex+1;
if (CheckBox1.State = cbChecked) and (CheckBox2.State = cbUnChecked) then n:=1
else
     if (CheckBox2.State = cbChecked) and (CheckBox1.State = cbUnChecked) then
n:=2 else
     if (CheckBox2.State = cbChecked) and (CheckBox1.State = cbChecked) then
n:=3 else  n:= 0;
        case n of
     0: Clear;
     1: begin fl:=true;
             case i of
             1: LitlPicture(i);
             2: LitlPicture(i);
             3: LitlPicture(i);
             4: LitlPicture(i);
             5: LitlPicture(i);
             6: LitlPicture(i);
             end;
             Image1.Picture.Bitmap:=Bt1;
             Bt1.Free; end;
     2:  Kvit;
     3: begin fl:=true; Init;
Bt1.Canvas.Pen.Color:=ClBlack;
Bt1.Canvas.Brush.Color:=clYellow;
Bt1.Canvas.Ellipse(x0,y0,x0+20,y0+20);
Bt1.Canvas.Pen.Color:=ClBlack;
Bt1.Canvas.Brush.Color:=clFuchsia;
Bt1.Canvas.Ellipse(x0-5,y0-30,x0+25,y0);
Bt1.Canvas.Ellipse(x0+20,y0-5,x0+50, y0+25);
Bt1.Canvas.Ellipse(x0-30,y0-5,x0,y0+25);
Bt1.Canvas.Ellipse(x0-5,y0+20,x0+25, y0+50);
             case i of
1: begin x:=xx div 2;y:=yy div 3; PchelkaRight(x,y,0);end;
2: begin x:=xx div 3;y:=yy div 3; PchelkaLeft(x,y,0);end;
```

```
3: begin x:=xx div 2;y:=yy div 3; PchelkaUpRight(x,y,0);end;
4: begin x:=xx div 3;y:=yy div 3; PchelkaUpLeft(x,y,0);end;
5: begin x:=xx div 2;y:=yy div 2; PchelkaDounRight(x,y,0);end;
6: begin x:=xx div 3;y:=yy div 2; PchelkaDounLeft(x,y,0);end;end;
        Form3.Image1.Picture.Bitmap:=Bt1;
              Bt1.Free;
        end;       end;
     h:=i; end;
procedure TForm3.Button1Click  (Sender: TObject);
     var x,y:integer;
     begin
     if fl then begin Fon;
     x:=100;y:=50;
     case h of
     1: PchelkaRight(x,y,0);
     2: PchelkaLeft(x,y,0);
     3: PchelkaUpRight(x,y,0);
     4: PchelkaUpLeft(x,y,0);
     5: PchelkaDounRight(x,y,0);
     6: PchelkaDounLeft(x,y,0);
     end;
     Form1.PaintBox1.Canvas.CopyRect(Rect(0,0,xx1,yy1),Bt1.Canvas,Rect(0,0
     ,xx1,yy1));
     Bt1.Free;
     end;
     end;
procedure TForm3.FormShow(Sender: TObject);
     begin
     fl:=false;
     end;
     end.
```

Розглянемо призначення нових наведених процедур:

Процедура Init малює замальований прямокутник білим кольором за розмірами (xx, yy) на канві бітової матриці Bt1.

Процедура Clear виводить зображення білого замальованого прямокутника на формі в компоненті Image.

Процедура Kvit малює квітку як альтернативне попередньому зображенню. Змінні xx0, yy0 теж позначають розміри поля компоненти Image.

Процедура TForm3.RadioGroup1Click (Sender: TObject) доповнена опрацюванням можливих станів компоненти CheckBox.

В ході роботи над проектом було наведено багато прикладів створення графічних зображень. Тому вчитель може побудувати наступне завдання таким чином: доповнити компоненту вибору власно створеним зображенням бджоли.

Для виконання цього завдання учнів доцільно попередньо ознайомити з різними методами створення графічних зображень, які нададуть їм можливість вибору найприйнятнішого.

4. Методи створення графічних зображень

Графічні зображення в Delphi можна виводити прямо на форму, в компоненту PaintBox (сторінка System), або Image (сторінка Additional). Для створення графічних зображень можна використовувати властивість Canvas форми, компоненти PaintBox, бітової матриці Bitmap або компоненту Shape (малювання квадратів, кіл тощо). Компонент Bitmap не заданий на сторінках компонентів, але може бути створений програмним шляхом за допомогою методу Create. В нашому проекті вже зустрічався фрагмент програмного коду, за яким створюється бітова матриця Bt1 розмірами xx, yy:

```
Bt1:=TBitmap.Create;
Bt1.Height:=yy;
Bt1.Width:= xx;
```

Оператор Bt1.Free (метод Free) вилучає бітову матрицю з пам’яті.

Delphi надає кілька можливостей формування графічних зображень: програмний спосіб, використання вбудованого графічного редактора Image Editor, компонента створення простих фігур Shape.

Програмний спосіб передбачає створення графічного зображення на канві (при наявності властивості Canvas) вище перерахованих компонентів (Form.Canvas.....: PaintBox.Canvas.....; Bitmap.Canvas.....) за допомогою наступних методів:

Canvas.Pixels(x,y) – ставить точку в координатах (x,y) відносно графічного об’єкту.

Canvas.MoveTo(x,y) – переміщення пера в точку (x,y) без малювання.

Canvas.LineTo(x,y) – малює пряму лінію від біжучої точки до нової (x,y).

Canvas.PolyLine([Point(x1,y1),Point(x2,y2),...,Point(xn,yn),Point(x1,y1)])– формує замкнутий багатокутник (функція Point створює точку (x,y)). В цьому

методі перша та остання точки автоматично не поєднуються, тому в кінці необхідно знов вказати координати першої точки. Наприклад, трикутник буде намальований завдяки програмному коду:

```
Canvas.PolyLine([Point(10,10),Point(100,100),Point(50,75),Point(10,10)]);
```

Canvas.FillRect(Rect(x1,y1,x2,y2)), Canvas.Rectangle(x1,y1,x2,y2) – створення зображення замальованого прямокутника, x1,y1 – координати лівого верхнього кута, x2,y2 – правого нижнього кута, функція Rect формує прямокутну область.

Canvas.Ellipse(x1,y1,x2,y2) – формує зображення еліпса, в разі необхідності – кола, координати x1,y1,x2,y2 – позначають прямокутну область малювання еліпса. Для малювання частини еліпса необхідно 8 параметрів. Перші 4 такі ж, як при малюванні повного еліпса, а інші 4 – точки, які визначають частину еліпса, що буде показана.

Canvas.Pie(x1,y1,x2,y2,x3,y3,x4,y4) – малює частину кола, x1,y1,x2,y2 – визначення прямокутної області для малювання кола, x3,y3,x4,y4 – визначення частини кола, що має бути зображена. Наприклад,

Canvas.Pie (100,100,200,200,100,100,100,200).

Canvas.Arc(x1,y1,x2,y2,x3,y3,x4,y4) – малювання дуги, призначення 8-ми координат аналогічне наведеному у попередньому методі.

Перераховані методи створюють графічні зображення визначеним кольором, типом замальовування, товщиною лінії. У зв’язку з цим необхідно зробити відповідні присвоювання властивостям пера (Pen) та пензля (Brush). Властивості Pen.Color можна присвоювати значення двома способами: визначеною в Delphi константою (наприклад, Canvas.Pen.Color:=clBlue), функцією RGB(x,y,z), яка задає значення кольору комбінації. Накладання числових значень зеленого, червоного та синього. (Наприклад, RGB(255,0,0) – червоний колір, RGB(255,255,0) – жовтого тощо). Властивість Pen.Width визначає товщину лінії в пікселях. Властивість стилю Pen.Style задає тип лінії (psSolid, psDash, psDot, psDashDot, psClear, psInSideFrame). Властивість Pen.Mode

встановлює режими взаємодії з середовищем малюнка. Може приймати значення:

pmCopy – використання визначеного кольору;

pmNot – перо малює кольором, інверсним кольору фону;

pmXor, pmNotXor – малювання об’єкту без заміни заднього плану.

Пензель (Brush) має аналогічні властивості Color, Style і додаткову Bitmap (бітова матриця 8*8 пікселів), яка визначає зразок замальовування об’єктів, створюється у вбудованому графічному редакторі Image Editor.

В нашому проекті ми створювали графічні зображення на бітовій матриці, яку потім виводили у розміщенні на формі компоненти відповідними методами.

1. Для виведення бітової матриці на канву форми використовуються методи Draw(x,y,Bitmap1), StretchDraw(Rect(x1,y1,x2,y2),Bitmap1).
2. Для копіювання зображення на канву об’єкту PaintBox використовується метод CopyRect. Наприклад, PaintBox1.Canvas.CopyRect(Rect(0,0,200,200),Bitmap1.Canvas.Rect(0,0,200,200)).
3. Для виведення графічного зображення в компоненті Image, його властивості Picture.Bitmap присвоїти значення створеної бітової матриці:

Image1.Picture.Bitmap:=Bt1.

Як зазначалося вище, другий спосіб створення малюнків – використання вбудованого графічного редактора. Увійти до нього можна через меню Tools ⇒ Image Editor, вибрати тип зображення, його розміри, режим роботи, створити малюнок, який потім зберегти у файлі. Створену картинку можна вивести в зазначені компоненти методом LoadFromFile(‘NemeFile.bmp’):

Image1.Picture.LoadFromFile(‘NameFile.bmp’); Bitmap1.LoadFromFile(‘NameFile.bmp’).

Використання стандартного для Windows вікна діалогу (якщо невізуальна компонента OpenDialog присутня на формі) дозволяє вибрати

необхідний файл із записаних на диску: Image1.Picture.LoadFromFile(OpenDialog1.FileName). Вікно для запису створеного програмним шляхом зображення до файлу, здійснюється методом SaveToFile: Bitmap1.SaveToFile('NameFile.bmp').

Третій спосіб створення графічних зображень – використання компоненти TShape, яка формує елементи простих зображень (коло, еліпс, прямокутник тощо). При цьому програмний код маніпулювання зображенням (перемальовування) прихований в самій компоненті. При переміщенні графічного об'єкта достатньо змінювати тільки координати його розміщення на формі. Наприклад, Shape1.Left:=x, Shape1.Top:=y.

Після ознайомлення з наведеними методами створення графічних зображень, доцільно надати учням можливість виявити свої творчі здібності у створенні власного малюнка, який би доповнював список вже існуючих зображень бджіл в компоненті ListBox або RadioGroup.

5.**Організація введення текстової інформації**

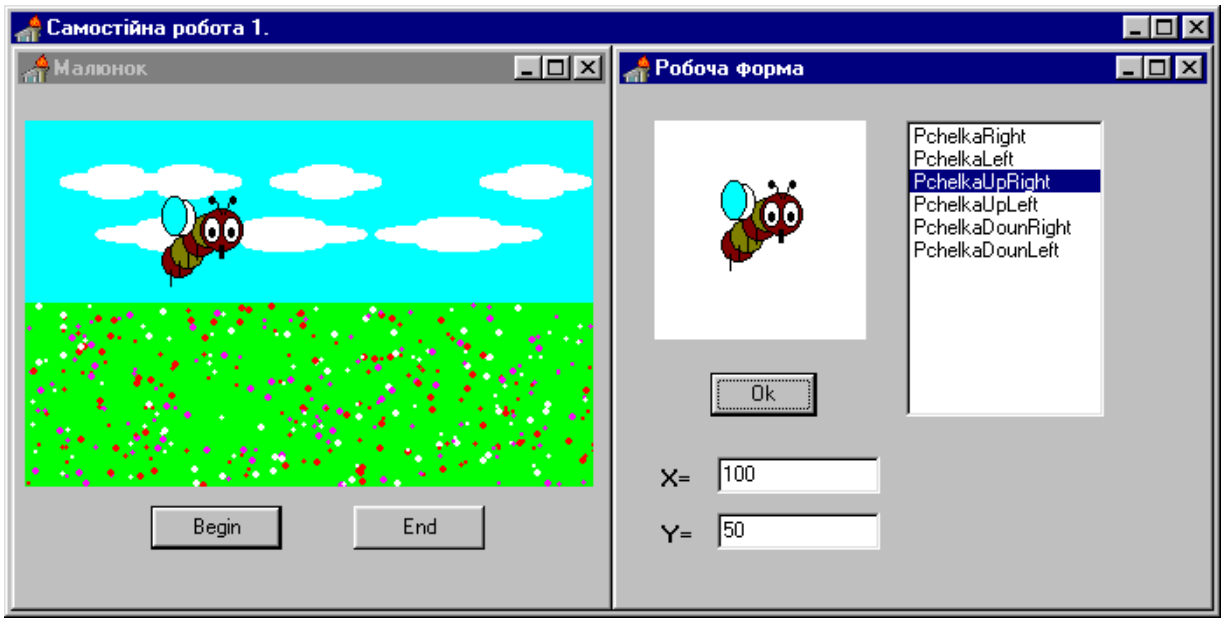
В попередніх проміжних реалізаціях проекту на малюнку лучного краєвиду форми Form1 зображення бджоли виводилось в зазначених програмним кодом координатах (x:=100, y:=50) відносно лівого верхнього кута компоненти PaintBox. Далі доповнимо проект відповідними засобами для введення значень початкових координат бджоли з клавіатури.

Для реалізації зазначеної дії Delphi надає компоненти Edit і Memo (сторінка Standard). Компонента Edit використовується для введення однорядкових текстів. На відміну від Edit, компонент Memo використовується для введення та відображення багаторядкових текстів. В програмному коді для занесення рядків використовується метод Add (Memo1.Lines.Add(E)), кількість рядків в Memo дає метод Count (k:=Memo1.Lines.Count), доступ до окремого рядка схожий на доступ до окремого (i – го) елемента в масиві

(Memo1.Lines[i], індекс першого рядка – 0), очищує компонент Memo метод Clear (Memo1.Clear). В інспекторі об’єктів до компоненти Memo можна заносити текстові рядки, для чого необхідно скористатися властивістю TString, за вибором якої з’явиться вікно вбудованого текстового редактора для набору рядків, теж саме, яке вже використовували при формуванні компоненти RadioGroup.

В проекті передбачається введення всього двох чисел (x,y) – координат місця знаходження бджоли. Тому в цьому випадку використання компоненти Memo не є раціональним. Доречніше застосувати дві компоненти Edit для введення відповідних координат, як це показано на малюнку (рис.Б.6).

Рис.Б.6. Реалізація проекту із застосуванням засобів введення інформації з клавіатури.



Але, як вже зазначалось, за допомогою компоненти Edit можна вводити тільки текстову інформацію. Тому в програмному коді для присвоєння значень цілим змінним x,y необхідно використати функцію StrToInt(Edit1.Text). Зазначеній дії відповідає фрагмент програмного коду в проекті Pro4, в якому додатково реалізована операція перевірки входження їх

значень в діапазон розмірів малюнка, тобто розмірів компоненти PaintBox1 лівої форми.

```
procedure TForm3.Button1Click(Sender: TObject);
var x,y,x1,y1:integer;sx1,sy1,s:string;
begin
Fon;
x1:=Form1.PaintBox1.Width;y1:=Form1.PaintBox1.Height;
sx1:=IntToStr(x1);sy1:=IntToStr(y1);
s:='Значення координат 0<=x<='+sx1+', 0<=y<='+sy1;
x:=StrToInt(Edit1.text);y:=StrToInt(Edit2.text);
if (x<=x1)and(x>=0)and(y<=y1)and(y>=0) then begin
case h of
1: PchelkaRight(x,y,0);
2: PchelkaLeft(x,y,0);
3: PchelkaUpRight(x,y,0);
4: PchelkaUpLeft(x,y,0);
5: PchelkaDounRight(x,y,0);
6: PchelkaDounLeft(x,y,0);
end;
```

Компоненту Memo доцільніше використовувати при зчитуванні та введенні текстової інформації у файл. При цьому необхідно на форму додатково розмістити невізуальні компоненти діалогу: OpenDialog (для відкриття файлу) та SaveDialog (для збереження файлу), які вікликають стандартні діалогові вікна Windows.

Програмний код для зчитування інформації з файлу в компоненту Memo**:**

```
If OpenDialog1.Execute then
 begin
Fnam:=OpenDialog1.FileName;
AssignFile(f,Fname);{f: file of.............}
If FileExists (Fname) then
     begin Reset(f);
             if not eof(f) then begin Read(f,E); Memo1.Lines.Add(E) end;
       end;
end;
```

Програмний код для запису інформації в файл:

```
If SaveDialog.Execute then
begin
Fname:=SaveDialog1.FileName;
AssignFile(F,Fname); Rewrite(f);
For i:=0 to Memo1.Lines.Count-1
Writeln(f, Memo1.Lines[i]);
End;
```

Надалі учням можна надати завдання в окремій формі реалізувати розглянуті операції опрацювання файлів з використанням подій, що

виникають при натисненні відповідних кнопок або при виборі пунктів меню. Самостійно реалізувати операцію доповнення файлу, яка передбачає комбінацію вище розглянутих дій (відкрити файл через SaveDialog, за процедурою Reset(f), після зчитування даних в компонент Memo вказівник переміщується на останній запис у файлі).

6. Використання кнопок різних типів

В середовищі візуального програмування Delphi програмісту надається для користування кілька типів кнопок.

Компонента Button (сторінка Standard) застосовується для формування звичайних кнопок. На самій кнопці виводиться значення властивості Caption (назва кнопки). Цей тип кнопок ми вже використовували в нашому проекті для виведення зображення в компоненту PaintBox лівої форми. Розглянемо способи створення і застосування кнопок інших типів.

Компонента BitBtn використовується для створення кнопок, на яких розміщується бітова матриця.

Компонента SpeedButton –кнопка, яка застосовується для створення спеціальних наборів кнопок, в яких одна з кнопок має знаходитись в постійно натисненому стані.

Загальним для цих кнопок є те, що для кожної з них необхідно створити бітову матрицю за допомогою вбудованого графічного редактора, а потім за допомогою властивості Glyph завантажити на поверхню кнопки bmp файл створеної матриці. Як зазначалося вище, компоненти SpeedButton на відміну від BitBtn дозволяють створити групу кнопок, в якій тільки одна може бути в натисненому стані, при натисненні іншої попередня кнопка переходить у відтиснутий стан. У таких кнопок, які заносяться до певної групи, властивості GroupIndex присвоюється значення номера цієї групи.

В нашому проекті використаємо перераховані типи кнопок для моделювання руху бджоли на фоні лучного краєвиду, що можна зробити двома способами:

1) виведення зображення в нових координатах (x,y);
2) часткове перемальовування фону, зміна значення параметра f(0 або1) в процедурі pchelka(x,y,f:integer), який відповідає за перемальовування крильців бджоли різними кольорами, що імітує її переміщення.

В проекті Pro7 (рис.Б.7) за допомогою двох компонентів BitBtn реалізований перший підхід моделювання переміщення бджоли, але тільки в двох напрямках. При натисненні відповідної кнопки передбачена зміна зображення бджоли. Учням пропонується проаналізувати наведений програмний код і для варіантів інших напрямків руху створити власні кнопки з відповідними бітовими матрицями на них. Окрім цього, слід передбачити обробку ситуації, коли значення координат (x,y) виходять за межі розмірів компоненти Form1.PaintBox1.

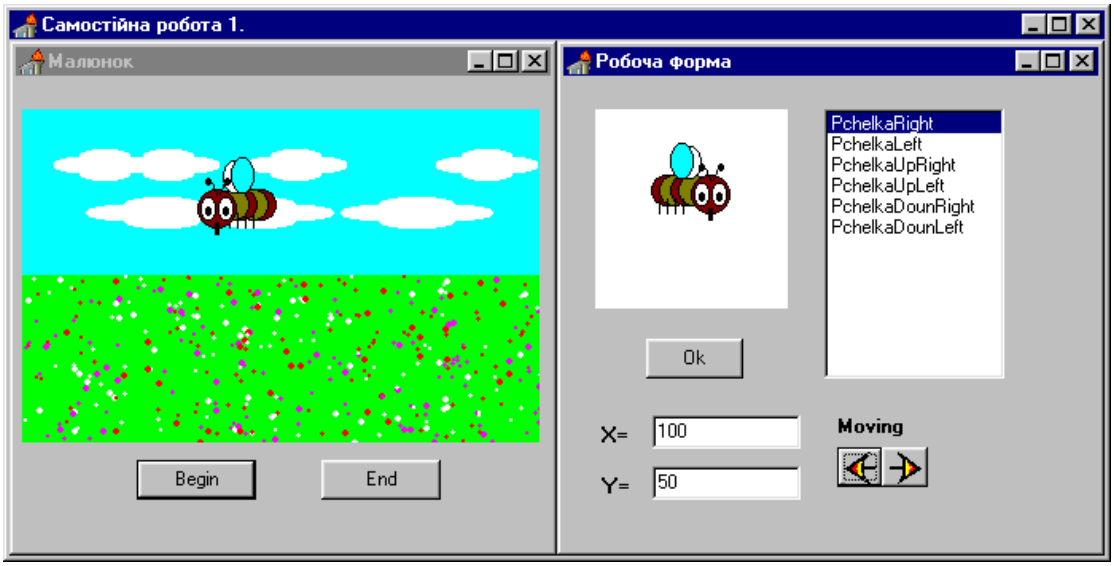


Рис.Б.7. Реалізація проекту Pro7.

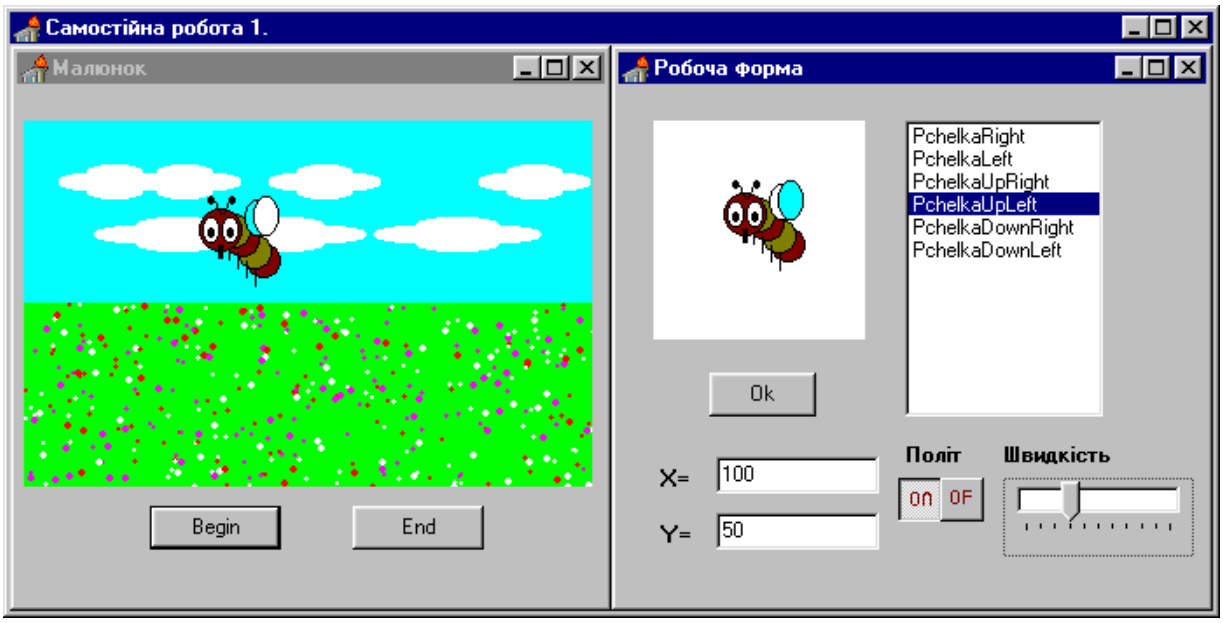
Фрагмент програмного коду робочого файлу WorkForm:

```
procedure TForm3.BitBtn1Click(Sender: TObject);
begin
fl1:=1;fl2:=0;Fon; PchelkaLeft(x,y,0); x:=x-5;
Form1.PaintBox1.Canvas.CopyRect(Rect(0,0,xx1,yy1),Bt1.Canvas,
                                Rect(0,0,xx1,yy1));
Bt1.Free;
end;
procedure TForm3.BitBtn2Click(Sender: TObject);
```

```
begin
fl1:=0;fl2:=1;Fon; PchelkaRight(x,y,0); x:=x+5;
Form1.PaintBox1.Canvas.CopyRect(Rect(0,0,xx1,yy1),Bt1.Canvas,
                                Rect(0,0,xx1,yy1));
Bt1.Free;
end;
```

Кнопки типу SpeedBtton (сторінка Additional) використаємо для моделювання переміщення бджоли другим способом SpeedBtton1 (‘ON’) – включає режим руху, SpeedBtton2 (‘OFF’) – виключає його. Швидкість перемальовування зображення відбувається в залежності від значення властивості Interval компонента Timer (сторінка System). Натиснення кнопки ‘ON’ передбачає включення таймера (Timer1.Enabled:=true), кнопка ‘OFF’ – його виключення (Timer1.Enabled:=false).

Зміна “швидкості руху” відбувається за рахунок зміни значення



властивості Timer1.Interval. Компонент TrackBar дозволяє реалізувати збільшення або зменшення значення властивості Interval компоненти Timer в залежності від значення своєї властивості Position, яке визначається позицією бігунка (Pro8, рис.Б.8).

Рис.Б.8. Реалізація проекту Pro8.

Фрагмент програмного коду проекту Pro8:

```
procedure TForm3.SpeedButton1Click(Sender: TObject);
begin
Timer1.Interval:=Timer1.Interval-TrackBar1.Position*50;
Timer2.Interval:=Timer2.Interval-TrackBar1.Position*50;
Timer1.Enabled:=true;
```

```
Timer2.Enabled:=true;
end;
procedure TForm3.SpeedButton2Click(Sender: TObject);
begin
Timer1.Enabled:=false;
Timer2.Enabled:=false;
end;
procedure TForm3.Timer1Timer(Sender: TObject);
begin
Fon;ChPchela(x,y,0,h);
Form1.PaintBox1.Canvas.CopyRect(Rect(0,0,xx1,yy1),Bt1.Canvas,
Rect(0,0,xx1,yy1));
Bt1.free;
end;
procedure TForm3.Timer2Timer(Sender: TObject);
begin
Fon;ChPchela(x,y,1,h);
Form1.PaintBox1.Canvas.CopyRect(Rect(0,0,xx1,yy1),Bt1.Canvas,
Rect(0,0,xx1,yy1));
Bt1.free;
end;
```

Завдання для учнів. Використайте компоненту UpDown (сторінка Win95) – кнопку–лічильник та компоненту Edit для реалізації маніпулювання швидкістю руху бджоли.

Додаток В

Реалізація проекту “Озеленення міста”

ПРЕДМЕТ МОДЕЛЮВАННЯ

В наш час екологічних катастроф та безвідповідального відношення до навколишнього середовища з кожним роком збільшується кількість шкідливих відходів. Сучасна наука здатна допомогти людині захистити себе та навколишнє середовище від екологічних негараздів надаючи природні, науково обгрунтовані засоби очищення повітря, захисту від шуму, електричного забруднення тощо.

Завдання. Створити програму моделювання процесу озеленення міста “Створення зелених зон міста”, в якій передбачити ознайомлення з деревами, що використовуються для озеленення, імітацію побудови міста, висадження дерев, ознайомлення зі списками дерев, які необхідно замовити для озеленення побудованого міста і окремих його районів.

Дослідницька робота. В процесі дослідницької діяльності, з певних літературних джерел учні повинні з’ясувати, що для озеленення об’єктів господарської діяльності людини з урахуванням біологічних та екологічних особливостей рослин, рекомендується:

1) біля доріг слід насаджувати шумопоглинаючі рослини та рослини, що зменшують концентрацію сірчистого газу в повітрі (береза бородавчаста, клен сріблястий, клен польовий, тополя чорна, тополя біла тощо);

2) біля заводів корисно садити антимікробні рослини та рослини, стійкі до отруєння сірчистим газом (гінкго, калина цілолиста, липа серцелиста, сосна Веймутова, тополя чорна, тополя біла тощо);

3) біля шкіл та дитячих садків рекомендується насаджувати антимікробні рослини та індикатори забруднення (ялина європейська, яловець звичайний, бульденеж, дуб звичайний, гінкго, клен сріблястий тощо) ;

4) біля ліній електропередач слід насаджувати рослини, що зменшують електричну забрудненість навколишнього середовища (береза бородавчаста, дуб звичайний, дуб північний, горобина тощо);

5) біля річки доцільно висаджувати рослини, які зміцнювали б її береги (осика, вільха клейка, вільха сіра, тополя чорна, тополя біла, верба ламка тощо);

6) в парках і скверах бажано насаджувати декоративні та антимікробні рослини (яловець звичайний, ялина європейська, гінкго дволопатеве, каштан, сосна звичайна, сосна Веймутова тощо);

7) біля лікарень обов'язково повинні рости антимікробні рослини (яловець звичайний, береза бородавчаста, ялина європейська, дуб звичайний, дуб північний, калина цілолиста, клен гостролистий, сосна Веймутова тощо).

ПРИНЦИПИ РЕАЛІЗАЦІЇ ПРОГРАМИ

Створена в середовищі візуального програмування Delphi програма моделює створення зелених зон міста. Реальні об'єкти господарської діяльності людства (лікарня, школа, дитячий садок, лінія електропередач, завод, парк, водоймище) замінені відповідними графічними зображеннями. Моделі об'єктів господарської діяльності людства знаходяться у взаємозв'язку з рослинами відповідно з наведеними вище принципами озеленення. Базу даних програми становлять наступні 24 рослини, які мають характерні біоекологічні особливості, які варто врахувати при озелененні міста:

1. Яловець звичайний
2. Береза бородавчаста
3. Ялина європейська
4. Бульденеж
5. Дуб звичайний
6. Дуб північний
7. Гінкго
8. Граб звичайний

9. Калина звичайна
10. Калина цілолиста
11. Каштан
12. Клен гостролистий
13. Клен сріблястий
14. Клен польовий
15. Липа серцелиста
16. Осика
17. Вільха клейка
18. Вільха сіра
19. Горобина
20. Сосна звичайна
21. Сосна Веймутова
22. Тополя чорна
23. Тополя біла
24. Верба ламка

Розглянемо функціонування робочого блоку програми.

Робочий блок (рис.В.1) базується на основній робочій формі, яка в своєму складі має:

Рис. В.1 Робоча форма

- поле для розміщення моделей об’єктів господарської діяльності людини (імітація побудови міста) і моделювання їх озеленення (в центрі форми),
- зображення передбачених для озеленення рослин (по лівому і верхньому краях форми),

- клавіші з відповідними малюнками для вибору об’єктів господарської діяльності людини (в нижній частині форми),
- ряд функціональних клавіш:

“Ok!” – виконати озеленення міста,

“Del” – вилучити черговий об’єкт діяльності людини,

“Exit” – вихід з програми,

довга вертикальна клавіша з правого боку – очистити робоче поле.

Для кожного малюнка рослини і клавіші передбачено підказування – виведення інформації про назву зображення або призначення клавіші, якщо потримати покажчик миші на відповідному об’єкті форми деякий час.

На початку роботи рекомендується спочатку ознайомитись з довідковою

інформацією про передбачені для озеленення рослини. Для цього можна на зображенні рослини клацнути двічі лівою кнопкою миші. Виведеться збільшене зображення рослини та інформація про неї (рис. В.2).

Рис.В.2 Виведення довідкової інформації про рослини.

Так, в наведеному прикладі (рис.В.2) була вибрана рослина - каштан. У зв’язку з цим висвітлюється його зображення і довідкова інформація: ”Каштан- прикраса вулиць і парків. Він не вибагливий до умов зростання.

Його задовольняють різні грунти і до тепла він не вимогливий. За своїми декоративними якостями каштан - рослина рідкісна. Деревина каштана - легка і м'яка - використовується у промисловості та виробництві фанери і т.д.”.

Ознайомившись з інформацією про рослини, можна перейти до моделювання створення зелених зон міста. Натисненням мишкою клавіші з зображенням об’єкта господарської діяльності людини вибирається відповідний об’єкт. Клацнувши лівою кнопкою миші на робочому полі, можна розмістити вибраний об’єкт на полі в координати розташування покажчика миші. Якщо ми не задоволені місцем розташування чергового об’єкта, то він вилучається натисненням клавіші “Del”.

Розмістивши на полі необхідні об’єкти господарської діяльності людства,

можна виконати озеленення створеного міста натисненням клавіші “Ok!”. Результат виконання зазначеної операції ми бачимо на рис. В.3.

Рис.В.3 Результат виконання моделювання створення зелених зон міста.

В наведеному прикладі умовне місто складається з дороги, лікарні, заводу, дитячого садка. Поряд з позначками об’єктів господарської діяльності людини розміщені зменшені зображення рослин, які слід насаджувати у вказаних місцях (вздовж дороги, біля лікарні, заводу, дитячого садка тощо).

Якщо людині недостатньо наочно поданої інформації, то клацнувши лівою клавішею миші на зображенні об’єкта на робочому полі, він отримує список окремих рослин, які доцільно садити біля вибраного об’єкта господарської діяльності людини. Ця дія показана на рис. В.4. Якщо клацнути мишею на вільному місті поля, то виведеться загальний список необхідних для озеленення умовного міста рослин, який можна зберегти у файлі Spicok.txt для подальшого друку на принтері. Для продовження роботи програми вікна зі списками необхідно попередньо закрити.

Рис.В.4 Виведення списку дерев для озеленення окремого об’єкта господарської діяльності людини.

Далі можна або продовжити роботу з створення моделей інших зелених зон міста, очистивши робоче поле довгою вертикальною клавішею з права, або вийти з програми за клавішею “Exit”.

ОРІЄНТОВНІ ТЕМИ ПРОЕКТІВ

1. Створення програми контролю за роботою об’єкта-імітації, наприклад, побутового приладу: соковижималки, пральної машини, холодильника тощо.
2. Створення програми шифрування і дешифрування повідомлення на основі криптографічного методу з відкритими ключами.
3. Створення програми підпису повідомлення за допомогою криптографічного методу “цифровий підпис” та аутентифікації джерела даних.
4. Створення програми встановлення паролю на повідомлення і його злому за допомогою пошуку в хешированому словнику (хешированый словник: увесь набір слів, який передбачається надалі використовувати, запам’ятовується на магнітному носії у вигляді бази даних, злом зводиться до пошуку необхідного слова). Передбачити доповнення словника новими паролями.
5. Створення програми перекладу тексту по словам, яка передбачала б виконання таких функцій: постановка документів в чергу згідно з їх пріоритетом; встановлення напрямку перекладу (наприклад, російсько-український); переклад файлів згідно з чергою; накопичення статистичної інформації і ведення журналу роботи; створення і редагування словника користувача.
6. Створення програми складського обліку, яка передбачала б резервування товару за складським розрізом даних з урахуванням строку придатності партії товару та підрахунок збитків, якщо товар не був вчасно реалізований.

7. Створення програми “Електронна комерція”, яка передбачала б: роботу з двома каталогами (урахування товарів і замовлень); визначення товарів, які користуються більшим попитом; підрахування платежів за замовленнями.
8. Створення програми “Електронний офіс”, яка передбачала б: роботу з каталогом товарів з указаною ціною і постачальником; організацію централізованої закупки на задану суму; вибір постачальників з найнижчими цінами.
9. Створення програми “Електронний аукціон”, яка б: дозволяла організувати одночасну взаємодію покупців і продавців; формувала чергу товарів, виставлених на аукціон, згідно з їх пріоритетом; імітувала продаж товару на аукціоні.
10. Написання програми реалізації спрощеної геоінформаційної системи для створення карт вибіркових об’єктів на основі повної карти (наприклад, карту земельних ділянок, карту нерухомості тощо).
11. Створення програми імітаційного моделювання, наприклад, будування міста, за результатами чого визначається потік людських ресурсів між районами, формується схема транспортних потоків, проводиться статистичний аналіз.
12. Створення програми прогнозування погоди як систему розмірковувань на основі аналогічних випадків CBR (case based reasoning). Ця система прогнозування на майбутнє для вибору правильного рішення знаходить в базі даних минулого близькі аналогії наявної ситуації і вибирає ту саму відповідь, яка була для них правильною.
13. Створення програми для обліку успішність учнів класу, їх відвідувань занять, кількості і характеру зауважень, яка б дозволяла робити відповідні вибірки, проводити аналіз і будувати графіки.

14. Створення програми “Підготовка до конференції”, яка б дозволяла працювати із списком учасників, з розкладом залізничного вокзалу, номерами в готелі, аудиторним фондом тощо.
15. Створення програми “Покупка комп’ютерного класу”, яка б дозволяла аналізувати пропозиції та підраховувала вартість покупки.
16. Створення програми “Бюро зайнятості”, яка б мала базу даних вакансій, безробітних, та допомагала б у складанні відповідних договорів.
17. Створення програми “Комп’ютерний підручник для молодших школярів”.
18. Створення програми “Анкета” для проведення соціологічного дослідження з деякої проблеми та відповідного статистичного аналізу.
19. Створення програми “Екологічний стан області”, що дозволяла б формувати базу даних з деяких впливів на екологію регіону, будувати графіки та виконувати певний статистичний аналіз.
20. Створення програми моделювання деякого процесу, наприклад, фотосинтезу у рослин, яка б дозволяла досліджувати цей процес при зміні освітлення, виконувати певний статистичний аналіз.
21. Створення програми “Полив теплиці”, за допомогою якої при введених значеннях вологості грунту, повітря, виду культури і т.д. визначався режим поливу теплиці.
22. Створення програми “Підприємство”, яка дозволяє сформувати план випуску продукту (яких компонентів і в якій кількості необхідно, у яких постачальників їх краще купити), провести аналіз доходів і розходів деякого підприємства, що виготовляє кілька типів продукції, співпрацює з кількома постачальниками і реалізаторами продукції.
23. Моделювання фізичного дослідження, наприклад, визначення сили тяжіння в різних частинах Землі та інших планетах і їх супутниках.

Додаток Д

СЛОВНИК ТЕРМІНІВ

Абстракція, abstraction – суттєві характеристики об’єкта, які відрізняють його від усіх інших об’єктів і чітко визначають концептуальні межі для спостереження.

Абстрагування – процес виявлення абстракцій.

Агент, agent – об’єкт, на який діють інші об’єкти, і який сам діє на інші об’єкти.

Актор, actor – об’єкт, що діє на інші об’єкти, але не зазнає дії з їх боку.

Атрибут, attribute – складова структури об’єкта, іменована властивість, що описує діапазон значень, які можуть приймати екземпляри цієї властивості.

Взаємодія – поведінка, що описується набором повідомлень, якими об’єкти обмінюються між собою для досягнення деякої мети.

Видимість, visibility – здатність одного об’єкта бачити інший і, таким чином, посилатися на його ресурси зовні.

Деструктор, destructor – операція класу, яка звільняє стан об’єкта і (або) знищує сам об’єкт.

Домен – іменована множина скалярних значень одного типу.

Закрита, private – частина інтерфейсу класу, об’єкта чи модуля, закрита (невидима) для інших класів, об’єктів і модулів.

Захищена, protected - частина інтерфейсу класу, об’єкта чи модуля, невидима для інших класів, об’єктів і модулів за виключенням підкласів.

Зворотний інжирінг (реінжирінг), revers-engineering – відновлення логічної або фізичної моделі системи за програмним кодом.

Ієрархія, hierarchy – підпорядкування або упорядкування об’єктів (класів). Визначаються два типи ієрархії: структура класів “загальне - окреме” і структура об’єктів “частина - ціле”.

Інкапсуляція, encapsulation – об’єднання та ізолювання елементів об’єкта в окремій структурній одиниці; розділення за різними частинами (відкрита, закрита, захищена) елементів об’єкта.

Інтерфейс, interface – зовнішній вигляд класу, об’єкта або модуля, який виокремлює його суттєві риси і не показує внутрішнього устрою і секретів поведінки.

Клас, class – сукупність об’єктів із загальною структурою і поведінкою.

Класифікатор (classifier) – механізм описання властивостей поведінки або структури. Класи, типи даних належать до класифікаторів.

Класифікація об’єктів (classification) – встановлення відповідності між деяким об’єктом і класифікатором.

Ключова абстракція, key abstraction – клас або об’єкт, що входить до словника предметної галузі.

Конструктор, constructor – операція, що створює об’єкт (або) ініціалізує його стан.

Метаклас, metaclass – клас класу; клас, екземпляри якого самі є класами.

Метод, method – операція над об’єктом, яка задана як частина описання класу.

Модуль, module – одиниця коду, що є будівельним блоком фізичної структури системи; програмний блок, який містить оголошення та фізичну реалізацію частини або всіх класів і об’єктів логічного проекту системи.

Модульність, modularity – властивість системи бути розділеною на модулі.

Об’єкт, object – конкретна матеріалізація абстракції; сутність з визначеними межами, в якій інкапсульовані її стан і поведінка; екземпляр класу.

Об’єктна модель, object model – сукупність основних принципів (абстрагування, інкапсуляція, модульність, ієрархічність тощо) об’єктно-орієнтованої парадигми програмування.

Підклас, subclass – клас, що є нащадком одного чи кількох класів.

Поведінка, behavior – дії і реакції об’єкта, виражені в термінах передачі повідомлень і зміни стану; видима зовні і відтворювана активність об’єкта.

Поліморфізм, polimorphism – положення теорії типів, згідно з яким однакові імена (наприклад, змінних, методів) можуть відноситись до об’єктів різних класів але таких, що мають загальний батьківський клас; перевизначення методу батьківського класу в класі нащадка.

Успадкування, inheritance – відношення між класами, при якому клас використовує структуру і поведінку іншого класу або класів.

Відношення між класами

Залежність (dependency) – відношення між двома класами (відношення використання “клієнт-сервер”), при якому зміна в екземплярі одного класу (незалежному) призводить до зміни в об’єкті іншого (залежного) класу. Графічно на діаграмі класів залежність зображується пунктирної лінію із стрілкою, спрямованою від даного елемента, до того, від якого він залежить.

Приклад:

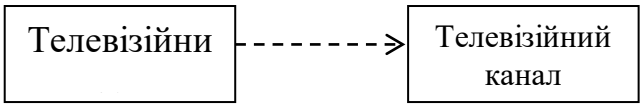


Рис.Д.1.

Асоціація (association) – відношення між класами, між якими існує зв’язок. Бінарна асоціація пов’язує два класи, N-арна асоціація (n-ary association) – асоціація між трьома або більшою кількістю класів. Кожний екземпляр такої асоціації – це упорядкований набір (кортеж), що місить n екземплярів із відповідних класів. Графічно зображується у вигляді лінії, що з’єднує класи

Приклад:
(бінарна асоціація)

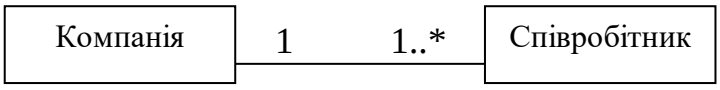


Рис.Д.2.

Цифри на лінії означають кратність асоціації. Кратність “1” для класу Компанія означає, що кожний співробітник може працювати тільки в одній

компанії. Кратність “1..*” для класу Співробітник означає, що в компанії може працювати кілька співробітників, кількість яких невідома. Конкретне число замість зірочки буде означати точне число співробітників в компанії.

Узагальнення (generalization) – відношення між більш загальним класом (пращуром) і окремим або спеціальним класом (нащадком). Описує відношення “загальне-окреме”. Графічно відношення узагальнення зображується лінією з великою незамальованою стрілкою, спрямованою до батьківського класу.

Приклад:

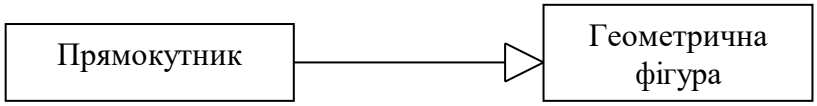
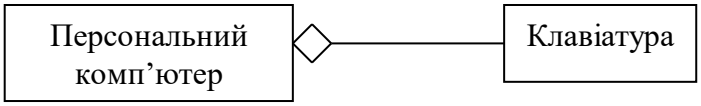


Рис.Д.3.

Агрегація (aggregation) – спеціальна форма асоціації, яка використовується для подання відношення “частина-ціле” між агрегатом (ціле) і його складовою.

Графічно зображується лінією з незамальованим ромбом на кінці, що спрямований на той клас, який є “цілим”.



Приклад:

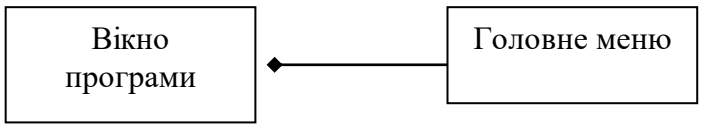
Рис.Д.4.

Композиція (composition) – форма асоціації агрегації, при якій складові цілого мають той самий час життя, що і ціле. Ці складові знищуються разом із знищенням цілого. Графічне зображення – лінія з замальованим ромбом біля класу, який є “цілим”.

Приклад:

Рис.Д.5.

Використано визначення термінів із джерел [34, с.469-478; 35, с.405-416].



Додаток Е

ТИПИ ДІАГРАМ

1. **Діаграма класів** (описує вигляд з точки зору проектування) – діаграма, на якій показано множину класів, інтерфейсів і відношень між ними;

Діаграма класів містить в собі наступні сутності:

- класи (описання сукупності об’єктів з загальними атрибутами, операціями, відношеннями і семантикою), класи з їх інтерфейсом (сукупність послуг, що надаються класом, описує видиму зовні поведінку об’єкта);
- відношення між класами: асоціація, агрегація, узагальнення (успадкування), залежність (використання).

Окремий клас позначається індивідуальним ім’ям, при необхідності перераховуються кілька його атрибутів та операцій.

Рис.Е.1. Позначення класу на діаграмі класів.

Приклад. Діаграма класів для проекту “Бібліотека” (рис.Е.2).

Зображена на рис.Е.2 діаграма має таку інтерпретацію. Бібліотека складається з двох великих відділів: каталогу і абонемента. Тому клас Бібліотека знаходиться у відношенні композиції до класів Каталог і Абонемент. Каталог бібліотеки містить великий список книжок, кількість яких точно не відома. Аналогічно, абонемент містить заздалегідь невідому кількість читачів. Окрема книжка може бути взятою одним читачем, але на руки читачу видається не більше п’яти книжок. Класи Каталог - Книжка, Абонемент – Читач, Читач – Книжка знаходяться у відношення асоціації. Кожний читач має номер читацького квитка, прізвище, адресу, список взятих книжок і дату останнього відвідування бібліотеки. З окремою книжкою пов’язується її код, прізвище автора, назва, ознака наявності в бібліотеці,

Рис.Е.2. Діаграма класів для проекту “Бібліотека”.

орієнтовна дата повернення та номер читача, який взяв цю книжку. Клас Дата як атрибут входить до класів Книжка, Читач, тобто клас Дата знаходиться у відношенні агрегації з цими класами. Символ “+” означає, що атрибут або метод належить до відкритої частини класу, а “-” – до закритої.

Використання діаграм класів допомагає розробникові програмного проекту прийняти рішення, які абстракції є частиною системи (входять до словника предметної галузі), які ні, для реалізації яких компонентів при програмуванні необхідно буде створювати класи, а які (інтерфейси) можна буде реалізувати за допомогою стандартних засобів середовища програмування**.**

2. **Діаграма об’єктів** (описує вигляд з точки зору проектування, процесів) – діаграма, на якій представлені об’єкти і відношення між ними. Вони є статичними “фотографіями” екземплярів сутностей (фіксують стан системи в конкретний момент часу), зображених на діаграмах класів.

На діаграмах об’єктів зображуються об’єкти – екземпляри класів (позначаються так як і класи на діаграмі класів) та зв’язки (позначаються прямою лінією). За допомогою діаграм об’єктів, як і за допомогою діаграми класів, моделюють статичний вигляд системи з точки зору проектування або процесів, але на основі реальних екземплярів. Приклад діаграми класів для проекту “Бібліотека” наведений на рис.Е.3.

3. **Діаграма прецедентів** або випадків використання (описує вигляд з точки зору прецедентів) – діаграма, на якій зображені прецеденти і суб’єкти (актори), які користуються системою , а також відношення між ними.

За допомогою діаграми прецедентів описується поведінка системи, що розробляється, без визначення її реалізації. Прецедент описує множину послідовностей, кожна з яких є взаємодією сутностей, що знаходяться зовні

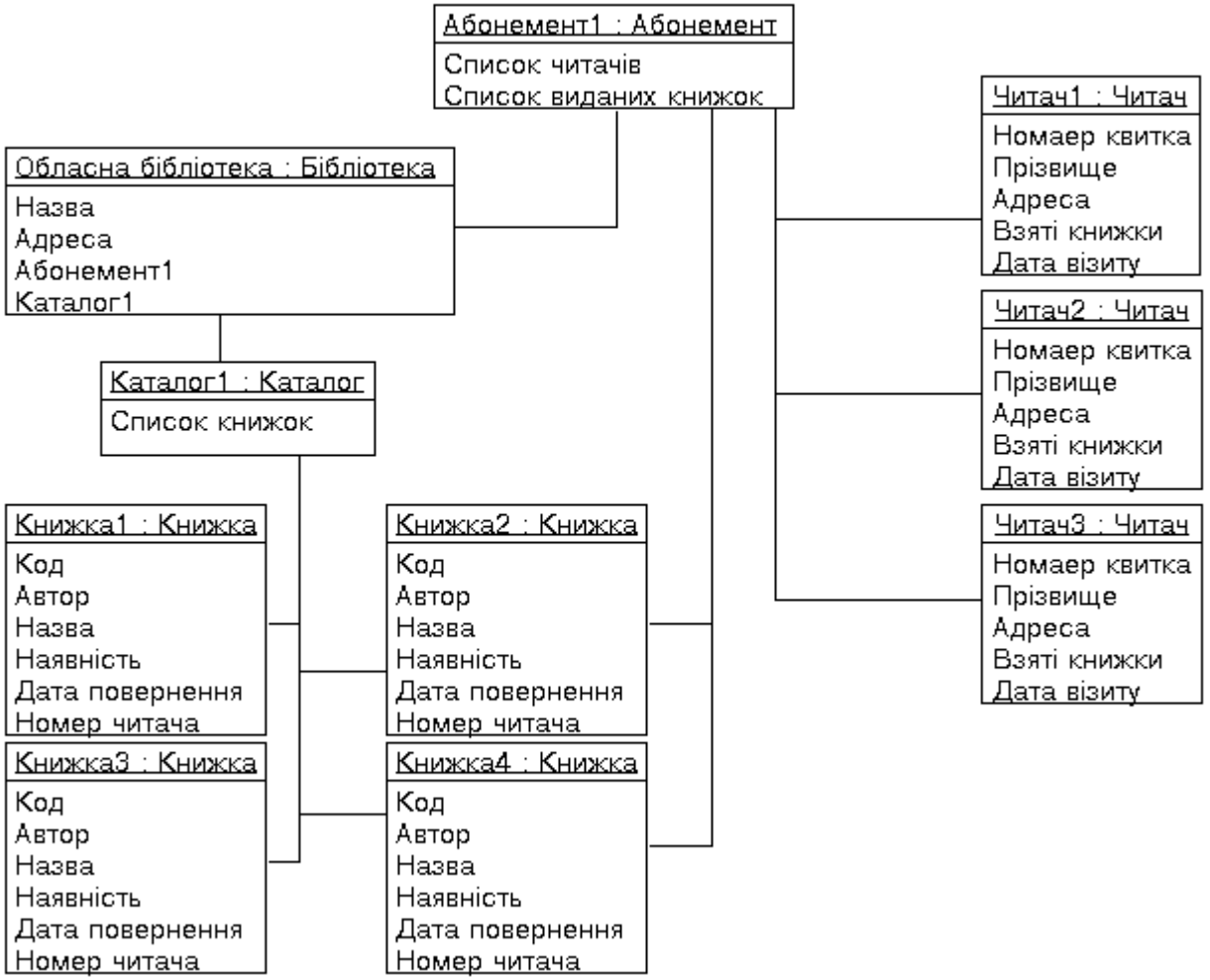


Рис.Е.3. Приклад діаграми об’єктів для проекту “Бібліотека”.

системи (її акторів), з системою в цілому та її ключовими абстракціями. Актор – це логічно зв’язана множина ролей, які грають користувачі під час взаємодії з системою. Акторами можуть бути як люди, так і автоматизовані системи. З точки зору окремого актору, прецедент повинен виконувати щось важливе для нього. Прецеденти можуть бути застосовані як до всієї системи, так і до її складових (підсистем, окремих класів, підсистем) і використовуватися як основа для їх тестування на різних етапах розробки. Прецедент описує, що робить система чи її складова, але не описує яким чином вона це робить, даючи зовнішнє уявлення про систему. Діаграми класів і об’єктів, навпаки, дають внутрішній вигляд системи.

Специфікувати прецедент можна за допомогою графічної діаграми прецедентів або шляхом описання потоку подій в текстовій формі. При застосуванні текстового описання прецеденту, необхідно вказати, як і коли прецедент починається і закінчується, коли відбувається взаємодія з акторами і відповідними об’єктами системи.

Один прецедент може включати в себе кілька, що відображають різні потоки подій. Прецеденти як і класи можуть бути організовані за допомогою відповідних відношень: узагальнення, включення та розширення. Відношення узагальнення означає, що прецедент - нащадок успадковує поведінку і семантику свого пращура, може заміщувати його або доповнювати його поведінку. Наприклад, якщо система має прецедент обмеженого доступу до списку виданих книжок “перевірка користувача”. Цей прецедент може мати двох нащадків “перевірка імені користувача”, “перевірка паролю”. Обидва прецеденти ведуть себе так, як і прецедент “перевірити користувача”, можуть використовуватися там, де використовується їх пращур, але при цьому кожний з них добавляють свою власну поведінку.

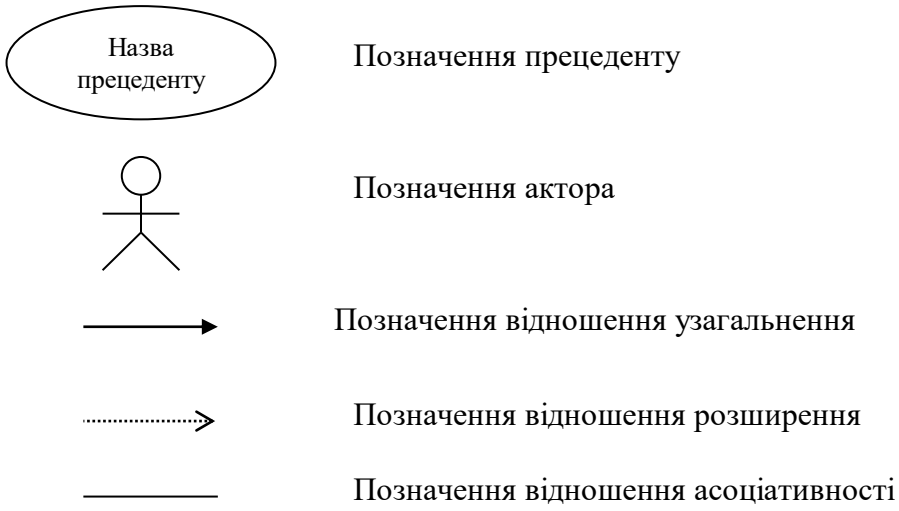
Відношення включення між прецедентами означає, що в деякій точці базового прецеденту відбувається включення поведінки іншого прецеденту. Завдяки наявності цього відношення надається можливість запобігти багаторазового описання однакового потоку подій, оскільки загальну

поведінку можна описати у вигляді самостійного прецеденту, що включається в базові. Наприклад, при доповненні каталогу даних новими книжками після введення чергового рецепту з'являється вікно з запитом "Будете вводити наступну книжку?" і відповідними кнопками. Те ж саме вікно, але з запитом "Будете записувати нового читача?" повинно виводитися у процесі роботи "адміністратора". Ці два описаних прецеденти мають однакову лінію поведінки, тому можуть бути описані як один і включатися у відповідні точки базових прецедентів роботи "бібліографа" та "адміністратора".

Відношення розширення використовують для моделювання таких частин прецеденту, які користувач сприймає як необов'язкову поведінку системи. Наприклад, в прецедент видачі книжки читачу можна було б включити прецедент розширення "перевірка права читача користуватися книжкою" (деякі книжки можуть видаватися читачам окремої професії). При звичайних обставинах базовий прецедент "видача книжки" виконується без врахування обмежень на користування книжками. Якщо зустрічається, наприклад, рідкісна книжка, то потік подій буде виконуватися, як звичайно, до точки розширення "обмеження користування книжкою", а в ній буде виконаний прецедент " перевірка права читача користуватися книжкою ".

Рис.Е.4. Умовні позначення для діаграми прецедентів.

Між акторами і прецедентами, між самими прецедентами можуть



встановлюватися відношення асоціативності (відношення зв’язка).

Як правило, на початку роботи потоки подій прецеденту описують в текстовій формі, а потім переходять до графічного зображення. На діаграмах прецедентів зображуються: прецеденти, актори, відношення між прецедентами. Для побудови діаграм прецедентів використовуються відповідні умовні позначення (рис.Е.4.).

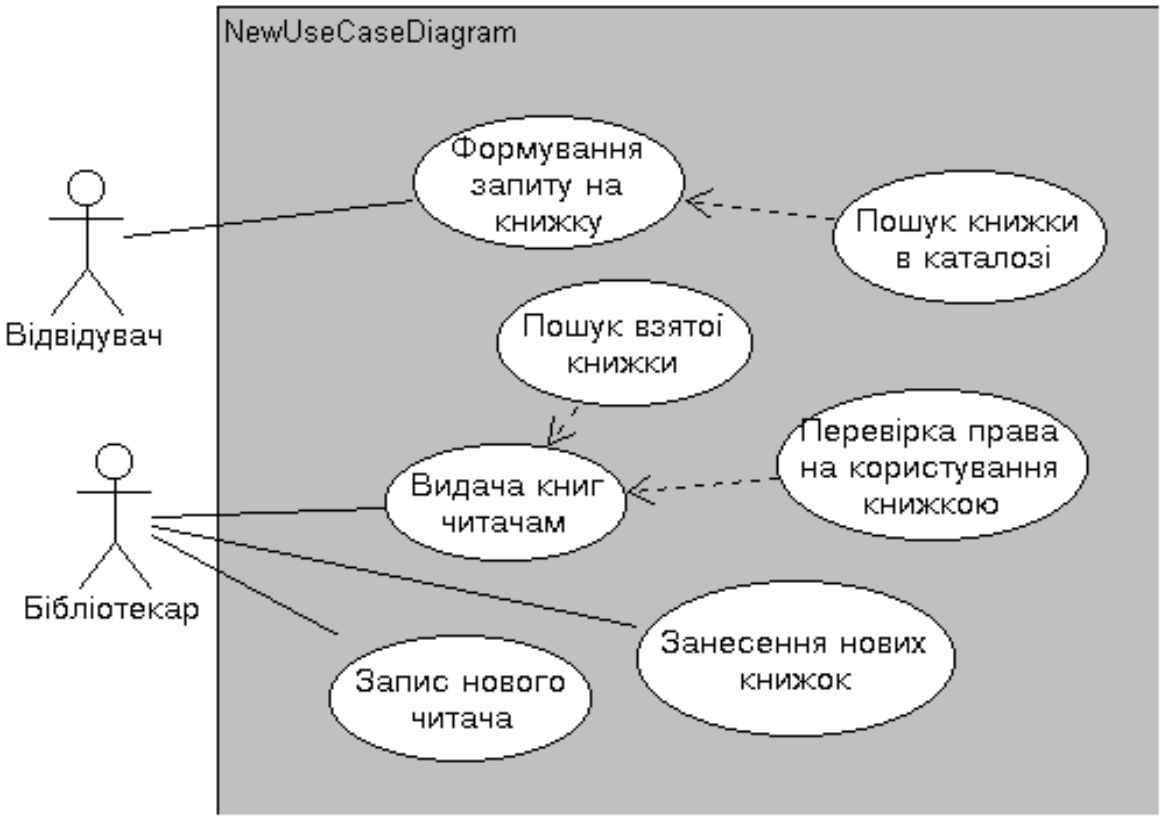


Рис.Е.5. Приклад діаграми прецедентів.

Діаграма прецедентів використовується для виявлення акторів та їх ролей, а також для моделювання вимог до системи, що передбачає визначення того, що система повинна робити (з точки зору зовнішнього спостерігача), незалежно від того, як вона це повинна робити. На основі діаграми прецедентів можна намітити, які додаткові класи і інтерфейси необхідно ввести до системи для реалізації виділених прецедентів. Приклад діаграми прецедентів для проекту “Бібліотека” наведений на рис.Е.5.

4. **Діаграма функцій** (описує вигляд з точки зору прецедентів) – діаграма, на якій зображена схема виконуваних системою функцій для певного прецеденту її використання. Тобто прецедент аналізується і розкладається на окремі функції, важливі з точки зору реалізації системи. Зовнішній вигляд діаграми функцій аналогічний діаграмі прецедентів.

5. **Діаграма послідовностей** (описує вигляд з точки зору процесів) відображає упорядкованість повідомлень в часі.

Діаграма послідовностей дозволяє візуалізувати спостереження в думці за роботою компонентів системи. Протягом часу об’єкти можуть створюватися, змінювати значення своїх атрибутів, бути ліквідованими. Для створення такої діаграми компоненти, що беруть участь у взаємодії, розташовуються у верхній частині вздовж осі X. Кожна компонента представляється вертикальною лінією - лінією життя. Якщо об’єкт існує протягом всієї взаємодії, то його лінія життя промальована зверху донизу. Якщо об’єкт створюється під час взаємодії, то лінія життя починається після отримання повідомлення “Створити”. Якщо об’єкт ліквідується під час взаємодії, то лінія життя закінчується після прийняття повідомлення “Зруйнувати”. Повідомлення, які компоненти посилають і приймають, розміщуються вздовж осі Y. Повідомлення від однієї компоненти до іншої зображуються горизонтальною стрілкою між вертикальними лініями. Повернення управління до компоненти здійснюється аналогічною стрілкою. Коментарі справа від малюнка пояснюють взаємодію. Зображення більш пізнього повідомлення розміщується на діаграмі нижче. Час рухається зверху вниз. Фрагмент діаграми послідовностей зображена на рис.Е.6.

Рис.Е.6. Фрагмент діаграми послідовностей.

6. **Діаграма кооперацій** (описує вигляд з точки зору прецедентів, проектування, процесів) відображає структурну організацію об’єктів, що обмінюються повідомленнями.

Діаграма кооперації використовується для зображення передачі повідомлень між компонентами з урахуванням їх структури та зміни значень відповідних атрибутів. Для створення діаграми кооперації компоненти, що беруть участь у взаємодії, розміщуються у вигляді вершин графу. Зв’язки, що поєднують ц ...афу. Ці зв’язки доповнюють ... і посилають. Повідомленн ...ості із часовою послідовніст ...ою десяткової нотації Дьюі ...ня, вкладене у повідомленн ...ення 1 і т.д.). На діаграмі ...кові значення задіяних у к ... стан або роль компонентів змінюється під час взаємодії, то на діаграмі розміщуються їх дублікати з новими значеннями. Нові компоненти поєднуються зі стереотипами за допомогою відповідних повідомлень (стати або скопіювати). Одна діаграма кооперації показує тільки один потік управління, тому для складних проектів використовується кілька діаграм кооперації. На рис.Е.7 наведений приклад діаграми кооперації, на якій зображені прецеденти: “формування запиту на книжку”, “видача книжки”.

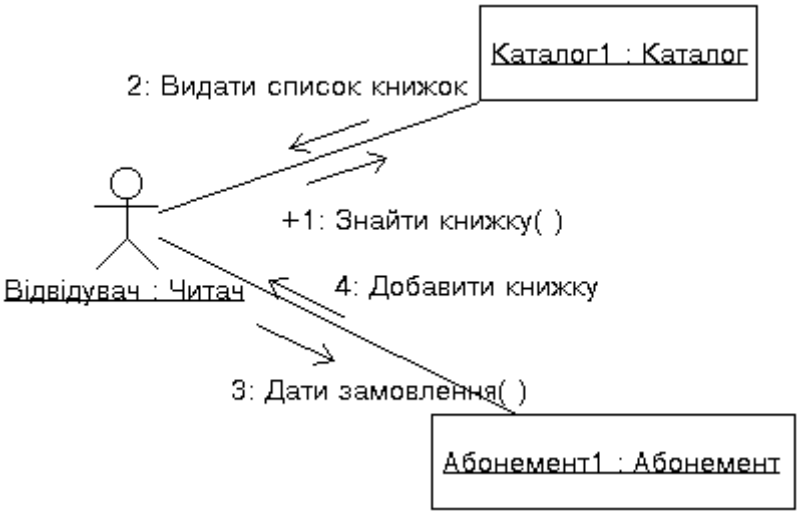


Рис.Е.7. Приклад діаграми кооперації.

7. **Діаграма діяльності** (описує вигляд з точки зору прецедентів, проектування, процесів) містить зображення потоку управління між об’єктами, переходів потоку управління від однієї діяльності до іншої.

Діаграми діяльності використовуються для моделювання динамічних аспектів системи. Але, якщо діаграма послідовностей акцентує увагу на упорядкованості повідомлень в часі, а діаграма кооперації – на структурних відношеннях між об’єктами, що взаємодіють, то в діаграмі діяльності увага зосереджується на змісті діяльності. Діаграма діяльності – це своєрідна блок-схема, яка описує послідовність виконання операцій. Окрема операція складається з атомарних дій, що призводять до змін у стані системи. Діаграма діяльності може включати: стани діяльності і стани дії, переходи та об’єкти. У графічному вигляді діаграма діяльності представляється за допомогою умовних позначень, наведених на рис. Е.8.

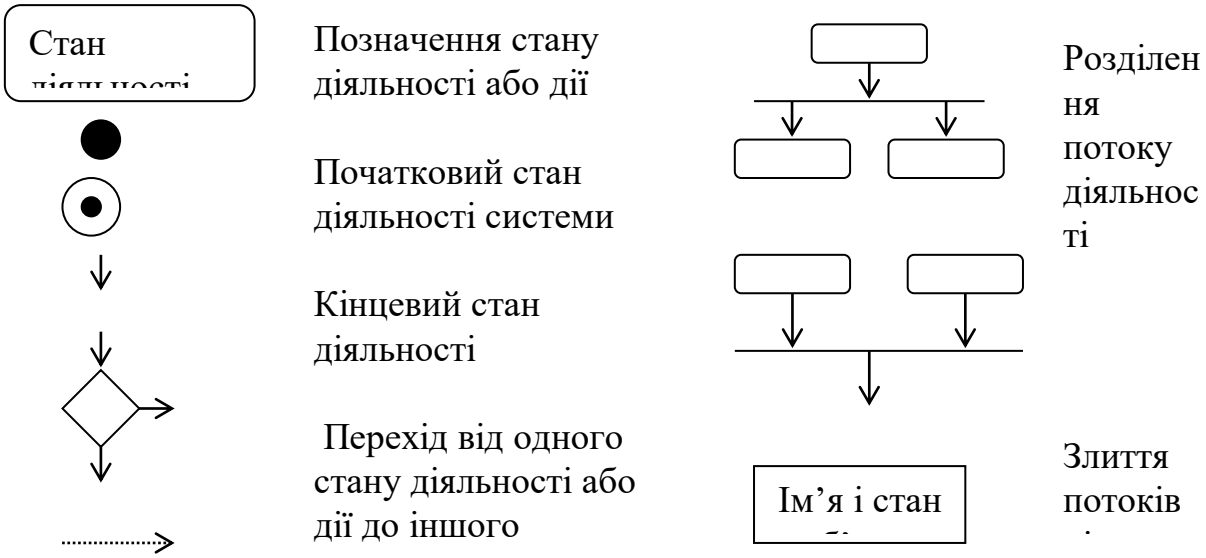


Рис. Е.8. Умовні позначення для діаграми діяльності.

Під станом дії розуміють окреме атомарне обчислення, наприклад: обчислення виразу, в результаті чого зміниться значення деякого атрибуту

об’єкта, виконання операції над об’єктом, відправлення йому повідомлення, створення або ліквідація об’єкта. Стани дії вважаються атомарними. В їх середині можуть відбуватися різні події, але виконувана в стані дії робота не може бути перервана. Передбачається, що тривалість одного стану дії займає невідчутно малий час. На відміну від стану дії, стан діяльності не є атомарним, його потік управління включає інші стани діяльності і дії. Стан діяльност м діяльності, може бу ображуються однаково, ві дії (дії, що виконуют позначається перехід г В діаграму діяльност різні шляхи виконанн виразу. В точку роз дити – два і більше. Д вираз, який обчислює уження. Але значення Об’єкти, що відносять о діаграми, прикріпля переходу, де вони ство

Рис.Е.9. Приклад діаграми діяльності.

Окрім послідовних і розгалужених в системах можуть існувати паралельні потоки управляння діяльністю. Один потік може розділятися на кілька незалежних потоків, що виконуються паралельно в один і той же час, а кілька потоків можуть об’єднуватися в один. При об’єднанні в точці злиття паралельні потоки синхронізуються, тобто кожний з них чекає, поки всі інші досягнуть цієї точки, після чого виконання діяльності продовжується в межах одного потоку.

Приклад системи “Бібліотека” не дозволяє продемонструвати паралельність і злиття потоків діяльності, це не передбачається її функціонуванням. Тому розглянемо один з прецедентів функціонування залізної дороги: “Станції С і О, О і D зв’язує залізна дорога в одну колію. На самій станції може бути кілька перонів. Поїзд А в заданий за розкладом час відправляється зі станції С, а поїзд В відправляється зі станції В. Досягнувши станції О, якщо шлях OD зайнятий, то поїзд А чекає прибуття на станцію О поїзда В, а потім відправляється”.

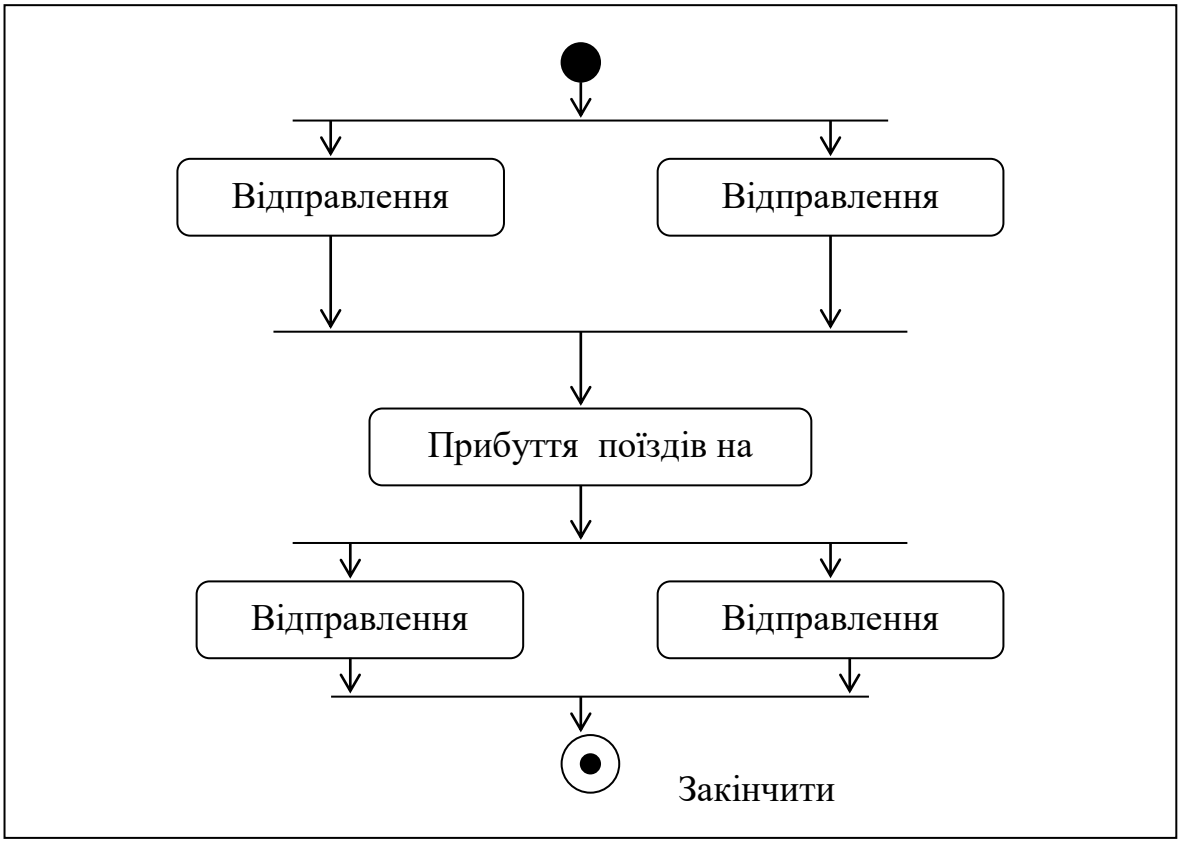


Рис.Е.10. Приклад діаграми діяльності для прецеденту управління залізною дорогою.

Моделюючи динамічні аспекти системи діаграми діяльності важливі для моделювання робочого процесу та моделювання окремих операцій. При моделюванні робочого процесу увага концентрується на діяльності з точки зору акторів, які співпрацюють з системою, в результаті чого виділяються процеси, що становлять її сутність. При моделюванні операцій на діаграмі за допомогою елементів блок-схеми (особливо, розгалуження, розділення і злиття) зображуються деталі обчислень, які можуть включати деталі обчислень та окремі об’єкти.

8. Діаграма станів (описує вигляд з точки зору прецедентів, проектування, процесів) акцентує увагу на поведінці об’єкта, що залежить від послідовності подій. На ній зображується автомат, що включає в собі стани, переходи, події та види його дій.

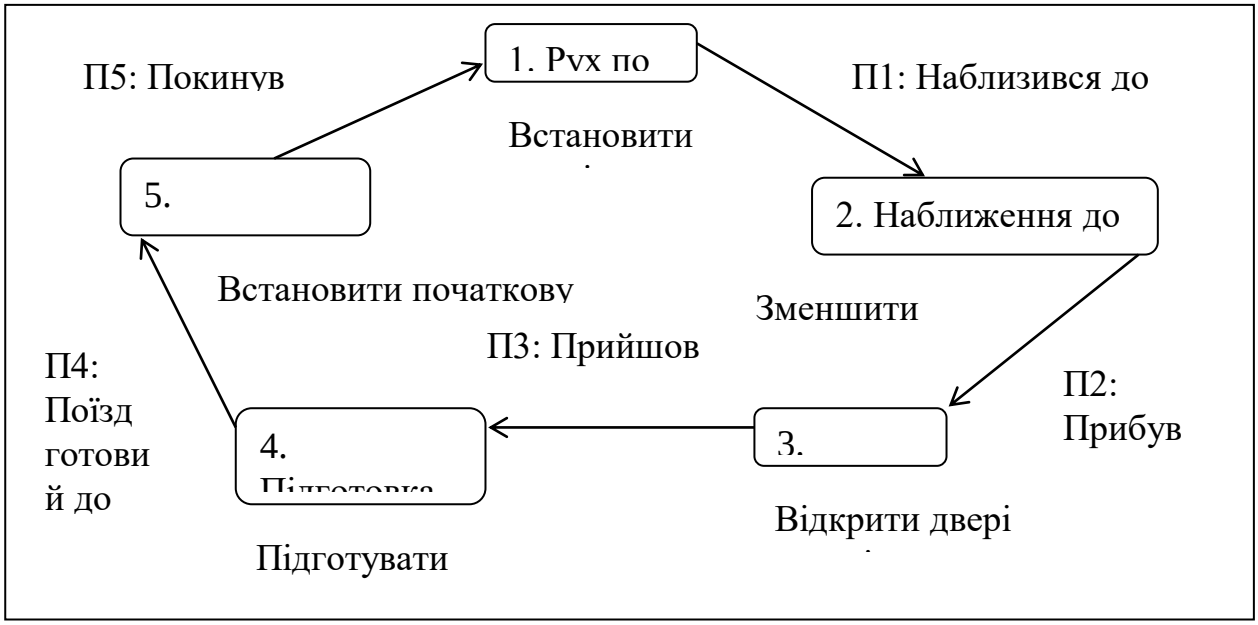
Використання для моделювання динамічних аспектів системи діаграм станів обумовлюється необхідністю моделювання поведінки реактивних об’єктів. Реактивний (що управляється подіями) об’єкт – це об’єкт, поведінка якого характеризується його реакцією на зовнішні події. Як правило, реактивний об’єкт знаходиться у стані очікування, поки не трапиться якась подія, а коли подія трапляється, його реакція залежить від попередніх подій. Після того, як об’єкт відреагує на подію, він знов переходить до стану очікування наступної події. Для таких об’єктів доцільно виділяти тривалі стани, події, що викликають перехід з одного стану до іншого, та дії, що виконуються при зміні стану. Для моделювання поведінки саме таких об’єктів з точки зору виникнення подій застосовуються діаграми станів.

Під автоматом розуміється описання послідовності станів, через які проходить об’єкт протягом всього життєвого циклу, реагуючи на події, та описання реакцій на ці події. Стан – це ситуація в житті об’єкта, протягом якої

він задовольняє деякій умові, здійснює деяку діяльність або очікує якоїсь подію. Стан визначається наступними елементами: ім’ям (текстовий рядок, який відрізняє один стан від іншого), дією при вході і виході, внутрішніми переходами (що відпрацьовуються без виходу зі стану), вкладеними станами (послідовними або паралельними), відкладеними для обробки в наступних станах подіями. Подія – це описання суттєвого факту, який відбувається в часі і просторі. Реактивні об’єкти можуть реагувати на наступні події: сигнали, виклики, закінчення часового проміжку, зміна стану. Під сигналом розуміють іменований об’єкт, який збуджується одним об’єктом і приймається, перехоплюється іншим (наприклад, механізм обробки виключних ситуацій). Подія виклик відбувається, коли один об’єкт починає виконання операції на іншому об’єкті, у якого є свій автомат, управління передається від відправника одержувачу, спрацьовує відповідний перехід, потім операція закінчується, одержувач переходить до нового стану і повертає управління відправникові. В контексті автоматів під подією розуміють стимул, здатний викликати перехід. Перехід – це відношення між двома станами, яке показує, що об’єкт, який знаходиться в першому стані, повинен виконати деякі дії і перейти до другого стану, як тільки відбудеться певна подія і будуть виконані відповідні умови. Стани, переходи разом з подіями, що їх викликають, та

діями є складовими діаграми станів. Для побудові графічного зображення діаграми станів використовуються відповідні графічні позначення (рис.Е.11).

Рис.Е.11. Умовні позначення для діаграми станів.



Прикладом реактивного об’єкта можна вважати поїзд, що рухається по залізничній колії. Поїзд починає свій рух згідно розкладу, коли закінчується певний проміжок часу, чекає прибуття зустрічного поїзду тощо. Його діаграма станів наведена на рис.Е.12.

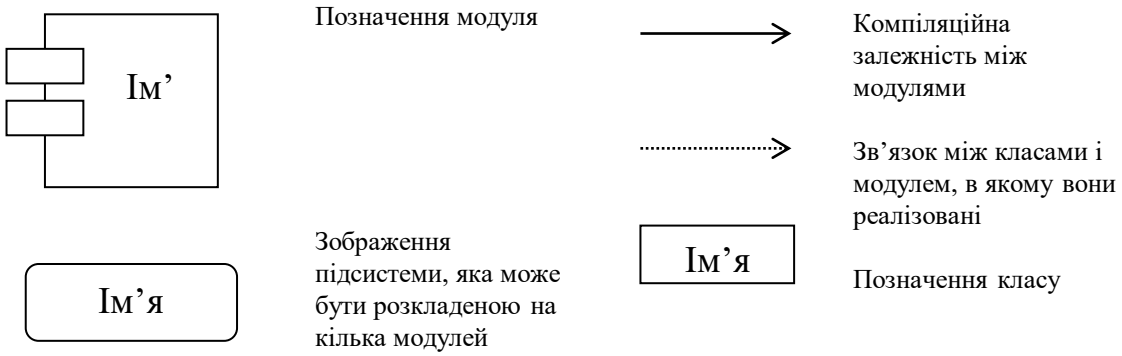
Рис. Е.12. Приклад діаграми станів для об’єкта “Поїзд”.

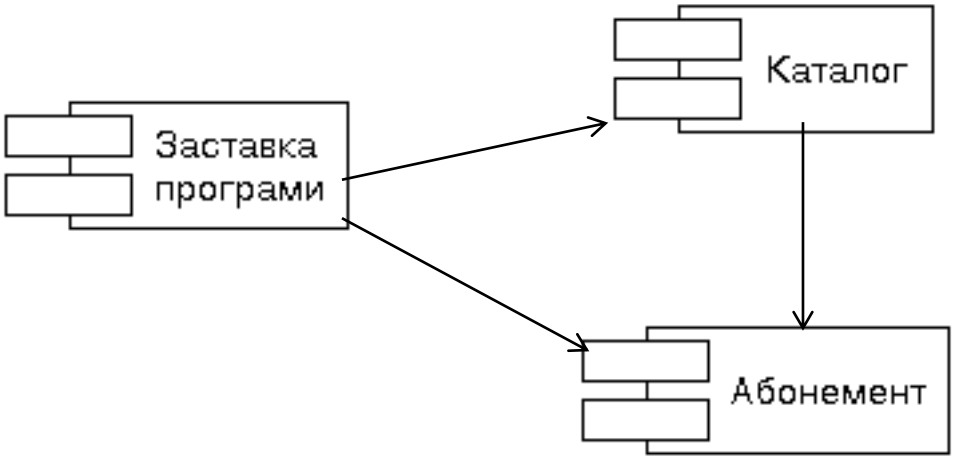
Якщо розглянуті вище моделі системи вважаються логічними (описують словник предметної галузі та взаємодію сутностей системи з точки зору їх структури та поведінки), то діаграми модулів та розгортування відображають фізичний вигляд системи.

9. **Діаграма модулів** (описує вигляд з точки зору реалізації) показує розподілення класів і об’єктів по модулям при фізичному проектуванні системи.

Діаграма модулів показує структуру модулів системи, в яких містяться визначенні в процесі логічного моделювання класи, об’єкти, реалізації їх інтерфейсів. Серед можливих модулів застосовуються: виконувані файли, бібліотеки (статична або динамічна об’єктна бібліотеки), таблиці (таблиці баз даних), файли даних тощо. Основними елементами діаграми модулів є модулі та їх залежності. Кожний модуль повинен мати своє ім’я, яке доцільно вибирати таким як і ім’я відповідного фізичного файлу в каталозі проекту. Єдиний зв’язок, який може існувати між двома модулями, - це компіляційна

залежність. Вона зображується стрілкою, що виходить із залежного модуля. Але в зображеній на діаграмі множині компіляційних залежностей не можуть зустрічатися цикли. У зв'язку з тим, що модуль може містити різні класи,





реалізацію інтерфейсів, то при необхідності вони відображаються на зображені діаграми поряд з модулями. Для побудови діаграм моделей використовуються відповідні умовні позначення (рис.Е.13).

Рис. Е.13. Умовні позначення для діаграми модулів.
Рис. Е.14. Приклад діаграми модулів.

Розбиття реалізації проекту на модулі дозволяє розподілити обов'язки між учнями групи, що виконуватимуть завдання із розробки окремих модулів програми, наприклад, модуля роботи каталогу в бібліотеці, модуля роботи абонемента, модуля вибору, переключення режиму роботи програми.

10. **Діаграма розгортання** (описує вигляд з точки зору розгортання) – діаграма, на якій представлена конфігурація обробляючих вузлів системи та розміщених в них компонентів. Застосовується при розробці професійного

програмного забезпечення, яке повинно обслуговувати сервери, окремі робочі станції, вузли тощо. Звісно, що такі операції занадто складні для учнівських проектів. Тому цей тип діаграм в більшості випадків не використовується.

Додаток Ж

Правила роботи над проектом в середовищі Visual UML

В CASE засобі Visual UML реалізовані загальноприйняті стандарти на робочий інтерфейс програми, подібно середовищам візуального програмування. Запуск цієї програми в середовищі MS Windows 95/98 призводить до появи на екрані робочого інтерфейсу (рис.Ж.1).

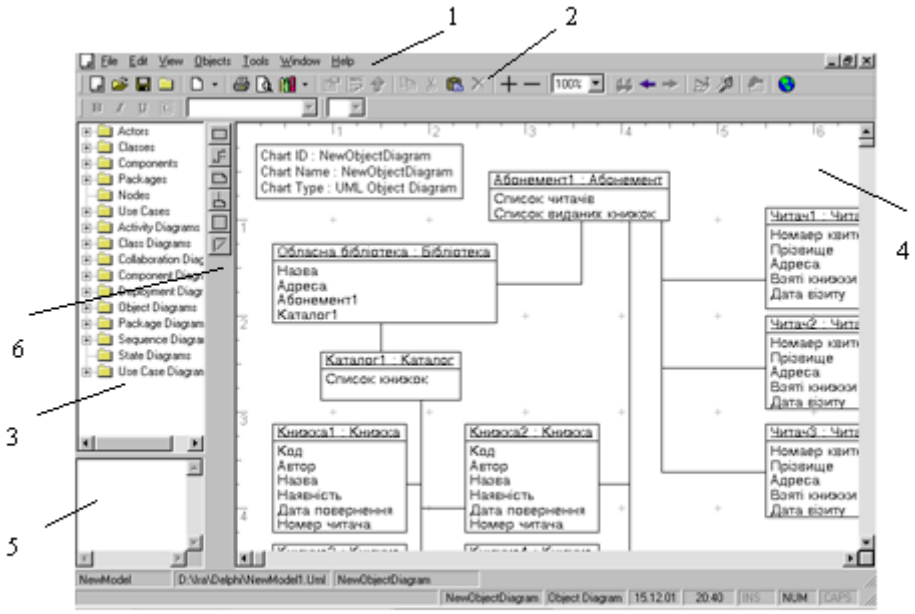


Рис.Ж.1. Загальний вид робочого інтерфейсу програми Visual UML.

Робочий інтерфейс Visual UML складається з різноманітних елементів, основними з який є: 1- головне меню програми, 2- стандартна панель інструментів, 3- вікно браузера, 4- вікно діаграми, 5- вікно документації, 6- спеціальна панель інструментів.

Окремі пункти головного меню, призначення яких зрозуміло з їхніх назв, об'єднують операції, що застосовуються до всього проекту в цілому (відкриття

проекту, роздруковування діаграм, копіювання в буфер і вставка з буфера різноманітних елементів діаграм тощо), та інші - специфічні операції (опції генерації програмного коду, перевірка узгодженості моделей, підключення додаткових модулів).

Стандартна панель інструментів забезпечує швидкий доступ до тих команд меню, що виконуються розроблювачами найчастіш. Користувач може настроїти зовнішній вигляд цієї панелі на свій розсуд. Для цього необхідно вибрати пункт меню **Tools -> Options** (Інструменти -> Параметри) і відчинити вкладку **Toolbars** (Панелі інструментів). Цим засобом можна показати або сховати різноманітні кнопки інструментів, а також змінити їх розмір.

Вікно брузера за замовчуванням розташовується в лівій частині робочого інтерфейсу під стандартною панеллю інструментів. Браузер організує зображення моделі у виді ієрархічної структури, що спрощує навігацію і дозволяє відшукати будь-який елемент моделі в проекті. При цьому будь-який елемент, який розробник додає до моделі, відразу відображається у вікні браузера. Відповідно, обравши елемент у вікні браузера, можна його візуалізувати у вікні діаграми або змінити його специфікацію. При бажанні вікно браузера можна розташувати в іншому місці робочого інтерфейсу або сховати зовсім, використовуючи для цього пункт меню **View** (Вид). Можна також змінити розміри браузера, перемістивши мишею межу його зовнішньої рамки.

Вікно діаграми є основною робочою частиною інтерфейсу програми. В ній візуалізуються різноманітні моделі проекту. За замовчуванням вікно діаграми розміщується в правій частині робочого інтерфейсу, проте його розташування і розміри також можна змінити. При розробці нового проекту, вікно діаграми порожнє, не містить ніяких елементів моделі. Назва діаграми, що зображена у вікні діаграми, вказується у рядку заголовка програми або, якщо вікно не розгорнуте на весь екран, у рядку заголовка вікна діаграми.

Одночасно у вікні діаграм можуть міститися кілька діаграм, проте активною може бути тільки одна з них. Перехід між діаграмами можна здійснювати вибором потрібного зображення на стандартній панелі інструментів або через пункт меню **Window** (Вікно). При активізації окремого виду діаграми змінюється зовнішній вигляд спеціальної панелі інструментів, яка налаштовується під конкретний тип діаграми. Можна настроювати склад спеціальної панелі, додаючи або вилучаючи окремі кнопки, що відповідають тим або іншим інструментам, за допомогою яких будуються діаграми. Про призначення кнопок можна дізнатися зі спливаючих підказок, що з'являються після затримки вказівника миші над відповідною кнопкою. Формування окремого елемента діаграми (задання атрибутів, перейменування тощо) можна здійснювати за допомогою як головного, так і контекстного меню, яке викликається натисненням правої клавіші миші .

Вікно документації за замовчуванням може не бути присутнім на екрані. В цьому випадку воно може бути активізоване через пункт меню **View - Documentation** (Вид-документація), після чого з'явиться нижче браузера. Вікно документації, як випливає з його назви, призначено для документування елементів зображення моделі. До вікна документації можна записувати різноманітну інформацію, і що важливо - українською мовою. Ця інформація надалі перетвориться у коментар і ніяк не впливає на логіку виконання програмного коду. У вікні документації активізується та інформація, що належить окремому виділеному елементу діаграми. При цьому виділити елемент можна або у вікні браузера, або у вікні діаграми.

Загальна послідовність робіт над проектом аналогічна послідовності робіт з створення розглянутих діаграм. Початковим кроком розробки нового проекту є створення окремих моделей в контексті побудови діаграм. Для нового проекту можна скористатися меню **File-New Model** (Файл-Створити модель). З'являється робочий інтерфейс програми з порожнім вікном

діаграми. Якщо є готовий проект (файл із розширенням uml - модель), то її можна відкрити для наступної модифікації через меню **File-Open** (Файл Відкрити). У цьому випадку програма завантажить існуючий проект з усіма наявними в ньому діаграмами і документацією. Після закінчення сеансу роботи над проектом виконану роботу доцільно зберегти у файлі проекту з розширенням uml. Це можна зробити через меню **File-Save** (Файл-Зберегти) або **File-Save As** (Файл-Зберегти як). При цьому вся інформація про проект буде збережена в одному файлі. Як і інші програми, Visual UML дозволяє настроювати глобальні параметри середовища, такі як вибір шрифтів і кольорів для зображення елементів моделі. Настроювання шрифтів провадиться через меню **Tools-Options** (Інструменти-параметри). Характерною рисою середовища є можливість роботи із символами кирилиці. Проте слід зауважити, що при специфікації елементів моделі з метою наступної генерації тексту програмного коду потрібно відразу записувати імена і властивості елементів символами тієї мови, що підтримується відповідною мовою програмування, яка вибирається через меню **Tools – Target Language** (Інструменти – цільова мова).

Загальний процес роботи над проектом полягає у додаванні на поверхні діаграми відповідних графічних елементів, у встановленні відношень між цими елементами. Створення нової діаграми починається з натиснення правої клавіші миші на назві діаграми в браузері, в контексному меню вибирається пункт **New**, що призводить до появи порожнього вікна для діаграми і занесення нової діаграми до браузера. Процес додавання графічних елементів на діаграми аналогічний реалізованому в популярних середовищах візуального програмування. При цьому варто застерегти від необережного додавання елементів на діаграми, оскільки кожний елемент, що додається, заноситься до браузера. Наступне видалення елемента з діаграми автоматично не видаляє його з браузера, і необхідно проводити додаткові заходи (натиснення клавіші **delete** на виділеному елементі) для видалення непотрібного елемента з моделі проекту.

Описані правила роботи над проектом є типовими не тільки для Visual UML, а й для інших аналогічних CASE засобів.

Правила роботи з Rose Delphi Link в середовищі Rational Rose

Утілита Rose Delphi Link (RDL), розроблена фірмою Ensemble Systems, забезпечує генерацію програмного коду на Delphi та зворотний реінжиніринг в CASE –системі Rational Rose. Програмний код генерується за діаграмою класів.

Після інсталяції RDL в Rational Rose в менюTools з'явиться новий пункт Ensemble Tools → Rose Delphi Link, вибір якого викликає вікно RDL. Вікно RDL (6, рис.Ж.2) поділяється на дві частини: в лівій частині створюється дерево елементів моделі розроблюваного програмного проекту в CASE –системі, а в правій частині розміщується дерево відповідних елементів (модулі, об'єкти, атрибути тощо) в Delphi. Кнопка "UpDateAll→" означає генерування програмного коду Delphi за побудованою діаграмою класів Rational Rose, а кнопка "← UpDateAll " виконує обернену операцію – за програмним кодом будує діаграму класів. Кнопка "Refresh" використовується

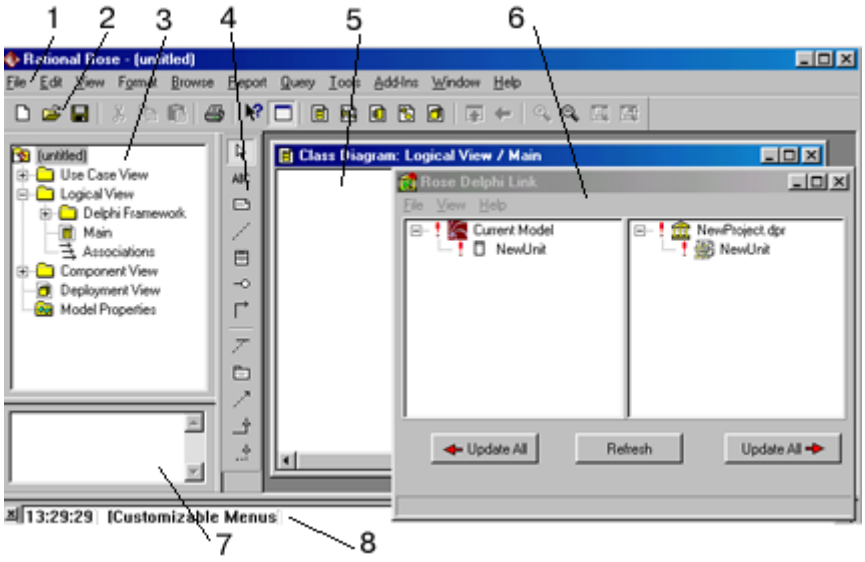


Рис.Ж.2. Середовище Rational Rose: 1– головне меню програми, 2- панель інструментів, 3- вікно браузера, 4- панель інструментів для побудови поточної діаграми, 5- вікно діаграми, 5- вікно RDL, 7- вікно документації, 8- вікно журналу.

для оновлення змісту вікна RDL після змін, які були зроблені в Rational Rose, наприклад, при доповненні діаграми класів.

Для роботи з програмним проектом в Delphi за допомогою CASE – засобу, що розглядається, необхідно спочатку викликати і перейти до вікна RDL. Через пункт меню File→New відкрити новий проект, у випадку відкриття існуючого проекту необхідно скористатися командою File→Open. При цьому в лівій частині екрана буде напис “Current Model” (поточна модель), а в правій – ім’я файла .dpr (проекту Delphi). Клацнувши правою кнопкою миші на написі “Current Model”, з контекстного меню, що з’явиться, можна вибрати New→Unit, тим самим додати до проекту окремий модуль. При цьому як у вікні RDL, так і у вікні браузера Rational Rose (3, рис.Ж.2) будуть відбуватися певні зміни. Знаки оклику у вікні RDL поряд окремими елементами означають, що ці елементи ще не занесені до програмного коду проекту. Для їх занесення необхідно натиснути кнопку “UpDateAll→”. Після цього в комп’ютері з’явиться програмний код поки ще порожнього проекту і його модуля.

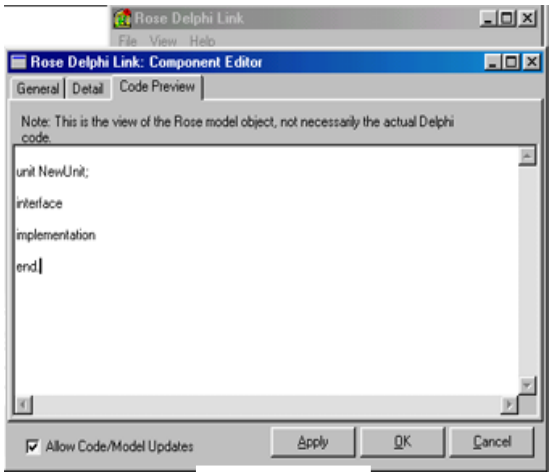


Рис.Ж.3.

Наведені програмні коди можна переглянути, або відкривши ці файли у “Блокноті”, або через команду Open Specification, яка знаходиться в контекстному меню, що викликається при натисненні правої кнопки миші, наприклад, на написі NewUnit. В другому випадку з’явиться вікно Component Editor (рис.Ж.3) з кількома закладками. На закладці General можна задати нове ім’я файлу модуля (Name), написати коментарі

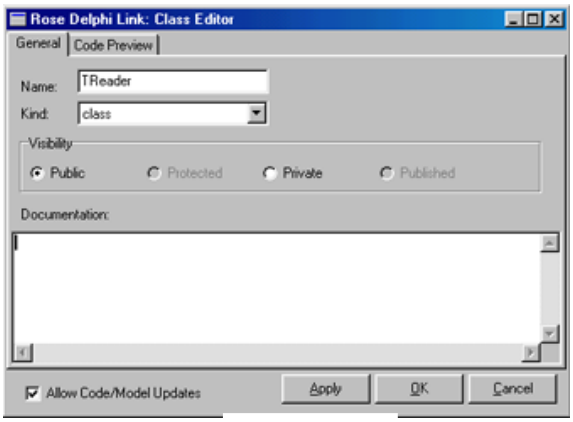


Рис.Ж.4.

(Documentation). На закладці Datail можна вказати шлях до файлу (Source), а на сторінці Code Preview можна переглянути і змінити програмний код модуля.

Аналогічно, клацнувши правою кнопкою миші у вікні RDL на написі NewUnit та вибравши команду контекстного меню New→Class, за допомогою вікна Class Editor (рис.Ж.4) можна добавити необхідні класи. При необхідності так само до створених класів через контекстне меню можна добавити оголошення їх атрибутів та методів, а до методів – їх параметри.

Створені класи автоматично будуть додані як у двох частинах вікна RDL, так і у вікні браузера Rational Rose (3, рис.Ж.2). Задавши класи, далі можна перейти до описання їх відношень між собою. Це зручніше виконувати у візуальному, графічному вигляді, з'єднуючи об'єкти між собою зображеннями певних відношень, вибраними на спеціальній панелі (4, рис.Ж.2). Для висвітлення діаграми класів у вікні браузера Rational Rose необхідно клацнути на написі "Overview". Наприклал, Читач бібліотеки (TReader) є Людиною (TPeopl), тобто клас TPeopl повинен бути загальним (батьківським) для класу TReader. Тому необхідно вибрати інструмент Generalization і провести лінію, з'єднуючи, від Читача до Людини. В результаті отримаємо відповідну діаграму класів (рис.Ж.5.).

Далі на діаграмі класів можна здійснювати певні зміни: додавати нові атрибути і методи існуючим об'єктам через контекстне меню (викликати його можна, клацнувши правою кнопкою миші прямо на графічному зображенні об'єкта), додати новий клас (навіть можна вибрати деякий стандартний клас Delphi), поміняти зв'язки між класами тощо.

Для того, щоб відобразити виконані зміни в програмному коді проекту, необхідно перейти до вікна RDL, а в ньому натиснути кнопку “Refresh ”. Тільки після цього і натиснення кнопки “UpDateAll→” буде згенерований відповідний програмний код для модуля NewUnit на Delphi.

Додаток З

Тест N1

Складні аналогії

Вам пропонується 20 пар слів, відношення між якими побудовані на абстрактних зв’язках. На цьому ж бланку в квадраті “Шифр” розташовані 6 пар слів з відповідними цифрами від 1 до 6. Вам необхідно визначити відношення між словами в парі, йому необхідно знайти аналогічну пару слів в квадраті “Шифр” та обвести кружком відповідну цифру.

Шифр
1. Вівця – стадо
2. Малина – ягода
3. Море – океан
4. Світло – темінь
5. Отруєння – смерть
6. Ворог - супротивник

1. Переляк – утеча 1 2 3 4 5 6
2. Фізика – наука 1 2 3 4 5 6
3. Правильно – вірно 1 2 3 4 5 6
4. Грядка – огород 1 2 3 4 5 6
5. Пара – два 1 2 3 4 5 6
6. Слово – фраза 1 2 3 4 5 6
7. Бадьорий – млявий 1 2 3 4 5 6
8. Свобода – воля 1 2 3 4 5 6
9. Країна – місто 1 2 3 4 5 6
10. Похвала – лайка 1 2 3 4 5 6
11. Помста – підпал 1 2 3 4 5 6

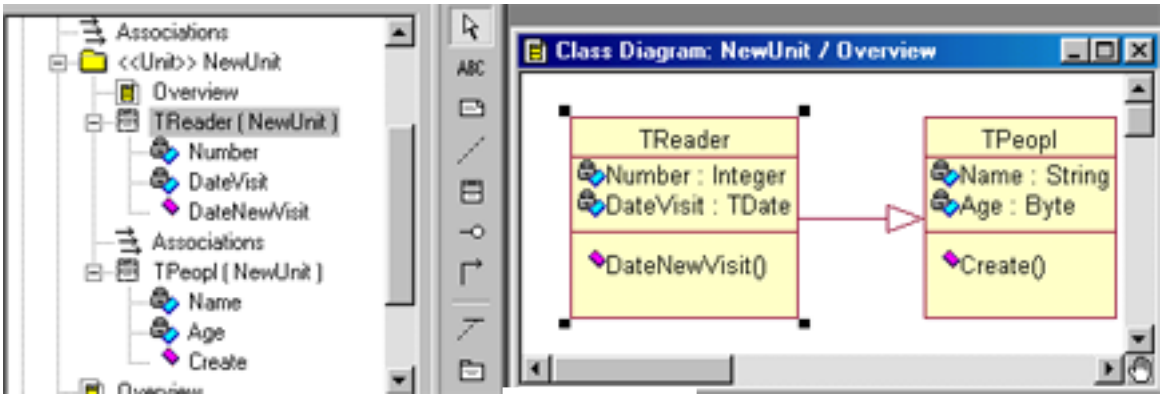


Рис.Ж.5.

12. Десять – число 1 2 3 4 5 6
13. Плакати – ревіти 1 2 3 4 5 6
14. Глава – роман 1 2 3 4 5 6
15. Спокій – дихання 1 2 3 4 5 6
16. Сміливість – геройство 1 2 3 4 5 6
17. Прохолода – мороз 1 2 3 4 5 6
18. Обман – недовіра 1 2 3 4 5 6
19. Спів – мистецтво 1 2 3 4 5 6
20. Тумбочка – шафа 1 2 3 4 5 6

Тест N2

Прості аналогії

Зліва розташовані пари слів (зразки), а з правого краю - набір слів, із яких буде підбиратися нова пара, аналогічна за ознакою, поданою в зразку. Потрібно підібрати аналогічну пару справа і номер слова з цієї пари, поданого в скобках, закреслити на бланку відповідей.

1. Школа – Директор: Гурток – (1.Голова, 2.Член, 3.Керівник, 4.Завідуючий, 5.Відвідувач).
2. Годинник – Час: Термометр – (1.Прибор, 2.Вимірювання, 3.Ртуть, 4.Тепло, 5.Температура).
3. Шукати – Знаходити: Розмірковувати – (1.Запам'ятовувати, 2.Приходити до висновку, 3.Розслідувати, 4.Пригадувати).
4. Круг – Куля: Квадрат – (1.Призма, 2.Прямокутник, 3.Тепло, 4.Геометрія, 5.Куб).
5. Робота – Результат: Знання – (1.Сила, 2.Праця, 3.Прогрес, 4.Досягнення, 5.Досконалість).
6. Ссавець – Коза: Харчі – (1.Продукт, 2.Їжа, 3.Обід, 4.Хліб, 5.Кухня).
7. Голод – Худоба: Праця – (1.Зусилля, 2.Втомленість, 3.Ентузіазм, 4.Плата, 5.Відпочинок).
8. Луна – Земля: Земля – (1.Марс, 2.Зірка, 3.Сонце, 4.Повітря, 5.Планета).
9. Ножиці – Різати: Орнамент – (1.Вишивати, 2.Прикрашати, 3.Створювати, 4.Малювати, 5.Випилювати).
10. Автомобіль – Мотор: Човен - (1.Кіль, 2.Весло, 3.Борт, 4.Вітрило, 5.Щогла).
11. Роман – Пролог: Опера – (1.Афіша, 2.Програма, 3.Лібрето, 4.Увертюра, 5.Арія).
12. Ялина – Дуб: Стіл – (1.Мбелі, 2.Шафа, 3.Скатертина, 4.Гардероб, 5.Гарнітур).
13. Язик – Гіркий: Око – (1.Зір, 2.Червоний, 3.Окуляри, 4.Світло, 5.Зіркий).
14. Харчі – Сіль: Лекції – (1.Нудьга, 2.Конспект, 3.Гумор, 4.Мова, 5.Логіка).
15. Рік – Весна: Життя – (1.Радість, 2.Старість, 3.Народження, 4.Юність, 5.Навчання).
16. Поранення – Біль: - Перевищення швидкості – (1.Відстань, 2.Протокіл, 3.Арешт, 4.Аварія, 5.Опір повітря).
17. Наука – Математика: Видання – (1.Типографія, 2.Оповідання, 3.Журнал, 4.Друк, 5.Редакція).
18. Гори – Перевал: Ріка – (1.Човен, 2.Міст, 3.Брід, 4.Паром, 5.Берег).
19. Шкіра – Дотик: Око – (1.Освітлення, 2.Зір, 3.Спостереження, 4.Погляд, 5.Зніяковілість).
20. Смуток – Настрій: Гнів – (1.Сум, 2.Лють, 3.Страх, 4.Почуття, 5.Прощення).

Тест N3

Виділення суттєвих ознак

У кожному рядку Ви знайдете одне слово, що стоїть перед дужками, і далі - 5-6 слів у скобках. Всі слова, що знаходяться в дужках, мають деяке відношення до того, що стоїть перед скобками. Виберіть тільки два і обведіть у кружок їхні номера на бланку відповідей.

1. Сад (1.рослини, 2.садівник, 3.собака, 4.паркан, 5.земля).
2. Ріка (1.берег, 2.риба, 3.рибак, 4.твань, 5.вода).
3. Міста (1.автомобіль, 2.будівля, 3.натовп, 4.вулиця, 5.велосипед).
4. Сарай (1.сінник, 2.коні, 3.дах, 4.худоба, 5.стіни).
5. Куб (1.кути, 2.креслення, 3.бік, 4.камінь, 5.дерево).
6. Ділення (1.клас, 2.ділене, 3.олівець, 4.дільник, 5.бумага).
7. Кільце (1.діаметр, 2.алмаз, 3.прохання, 4.опуклість, 5.друк).
8. Читання (1.очі, 2.книга, 3.картинка, 4.друк, 5.слово).
9. Газета (1.правда, 2.додатки, 3.телеграми, 4.бумага, 5.редактор).
10. Гра (1.карти, 2.гроки, 3.штрафи, 4.покарання, 5.правила).
11. Війна (1.аэроплан, 2.пушки, 3.бої, 4.рушниці, 5.солдати).
12. Книга (1.малюнки, 2.оповідання, 3.бумага, 4.заголовок, 5.текст).

13. Спів (1.дзвін, 2.мистецтво, 3.голос, 4.оплески, 5.мелодія).
14. Землетрус (1.пожежа, 2.смерть, 3.коливання грунту, 4.шум, 5.повінь).
15. Бібліотека (1.столи, 2.книги, 3.читальний зал, 4.гардероб, 5.читачі).
16. Ліс (1.грунт, 2.гриби, 3.мисливець, 4.дерево, 5.вовк).
17. Спорт (1.медаль, 2.оркестр, 3.змагання, 4.перемога, 5.стадіон).
18. Лікарня (1.приміщення, 2.уколи, 3.лікар, 4.градусник, 5.хворі).
19. Кохання (1.троянди, 2.почуття, 3.людина, 4.побачення, 5.весілля).
20. Патріотизм (1.місто, 2.друзі, 3.батьківщина, 4.родина, 5.людина).

Тест N4
Узагальнення та класифікація понять

Вам дані п'ять слів. Чотири з них об'єднані загальною ознакою. П'яте до них не підходить. Його треба знайти. Після цього на бланку відповідей закресліть цифру, що позначає обране Вами слово. Приклад: *1.Тарілка, 2.Чашка, 3.Стіл, 4.Кастрюля, 5.Чайник.* Перше, друге, четверте і п’яте означають посуд, а третє – меблі, тому його номер слід закреслити.

1. 1.Стіл, 2.Стілець, 3.Ліжко, 4.Підлога, 5.Шафа.
2. 1.Молоко, 2.Вершки, 3.Сало, 4.Сметана, 5.Сир.
3. 1.Черевики, 2.Чоботи, 3.Шнурки, 4.Валянки, 5.Тапочки.
4. 1.Молоток, 2.Кліщі, 3.Пила, 4.Цвях, 5.Сокира.
5. 1.Солодкий, 2.Гарячий, 3.Гіркий, 4.Кислий, 5.Солоний.
6. 1.Береза, 2.Сосна, 3.Дерево, 4.Дуб, 5.Ялина.
7. 1.Літак, 2.Віз, 3.Людина, 4.Корабель, 5.Велосипед.
8. 1.Василій, 2.Федір, 3.Семен, 4.Іванов, 5.Петро.
9. 1.Сантиметр, 2.Метр, 3.Кілограм, 4.Кілометр, 5.Міліметр.
10. 1.Токар, 2.Вчитель, 3.Лікар, 4.Книга, 5.Космонавт.
11. 1.Глибокий, 2.Високий, 3.Світлий, 4.Низький, 5.Дрібний.
12. 1.Дім, 2.Мрія, 3.Машина, 4.Корова, 5.Дерево.
13. 1.Доба, 2.Вечір, 3.Хвилина, 4.Секунда, 5.Рік.
14. 1.Блакитний, 2.Світлий, 3.Темний, 4.Яскравий, 5.Тьмяний.
15. 1.Успіх, 2.Перемога, 3.Удача, 4.Спокій, 5.Виграш.
16. 1.Гніздо, 2.Нора, 3.Мурашник, 4.Курятник, 5.Барліг.
17. 1.Француз, 2.Італієць, 3.Вегетаріанець, 4.Грузин, 5.Іспанець.
18. 1.Грабіж, 2.Крадіжка, 3.Землетрус, 4.Підпал, 5.Напад.
19. 1.Америка, 2.Європа, 3.Австрія, 4.Африка, 5.Азія.
20. 1.Ненавидіти, 2.Радіти, 3.Обурюватися, 4.Любити, 5.Називати.

Тест N5
Узагальнення та класифікація понять

Вам дані п'ять слів. Чотири з них об'єднані загальною ознакою. П'яте слово до них не підходить. Його треба знайти. Після цього закресліть на бланку відповідей цифру, що позначає обране Вами слово. Приклад: *1.тарілка, 2.чашка, 3.стіл, 4.кастрюля, 5. чайник.* Перше, друге, четверте і п’яте означають посуд, а третє – меблі, тому його номер слід закреслити.

1) 1.пряма, 2.ромб, 3.прямокутник, 4.квадрат, 5.трикутник.
2) 1.префікс, 2.прийменник, 3.суффікс, 4.закінчення, 5.корінь.
3) 1.прислів'я, 2.вірш, 3.поема, 4.оповідання, 5.повість.
4) 1.пейзаж, 2.мозаїка, 3.ікона, 4.фреска, 5.пензель.
5) 1.трикутник, 2.відрізок, 3.довжина, 4.квадрат, 5.круг.
6) 1.дощ, 2.сніг, 3.опади, 4.іній, 5.град.
7) 1.нарис, 2.роман, 3.оповідання, 4.сюжет, 5.повість.
8) 1.рабовласник, 2.раб, 3.селянин, 4.робітник, 5.ремісник.
9) 1.цитоплазма, 2.харчування, 3.ріст, 4.подразливість, 5.розмноження.

10) 1.барометр, 2.флюгер, 3.термометр, 4.компас, 5.азимут.
11) 1.кома, 2.точка, 3.двокрапка, 4.тире, 5.союз.
12) 1.морфологія, 2.синтаксис, 3.пунктуація, 4.орфографія, 5.термінологія.
13) 1.рух, 2.інерція, 3.вага, 4.коливання, 5.деформація.
14) 1.легенда, 2.драма, 3.комедія, 4.транедія, 5.преса.
15) 1.круг, 2.трикутник, 3.трапеція, 4.квадрат, 5.прямокутник.
16) 1.картина, 2.мозаїка, 3.ікона, 4.скульптура, 5.фреска.
17) 1.аорта, 2.стравохід, 3.вена, 4.серце, 5.артерія.
18) 1.глобус, 2.медіан, 3.полюс, 4.паралель, 5.екватор.
19) 1.робітник, 2.селянин, 3.раб, 4.феодал, 5.ремісник.
20) 1.Канада, 2.Бразілія, 3.Вьєтнам, 4.Іспанія, 5.Норвегія.

Додаток К

Створення діагностичних експертних систем за допомогою програмної оболонки DESS (Diagnostics Expert Systems Shell)

Програмна оболонка DESS застосовується для створення діагностичних експертних систем на основі семантичних мереж, тобто на основі орієнтованих графів, вершини якого – етапи процесу діагностики системи (об’єкти), а дуги – переходи до інших етапів діагностики (відношення між об’єктами). Проаналізувавши предметну галузь, учень в ролі розробника має можливість в DESS побудувати графову модель процесу здійснення діагностики (консультації), яка ілюструє можливі етапи процесу здійснення діагностики системою і логічні переходи між ними. При переході в режим користувача, надається можливість для роботи з створеною експертною системою.

Приклад К.1. Для розробки експертної системи скористаємось базою знань, яка надається в ботанічному визначнику:

Якщо клас голонасінні і форма листа лускоподібні, то родина – кипарисові**.**

Якщо клас голонасінні і форма листа голкоподібна і конфігурація хаотична, то родина – соснові.

Якщо клас голонасінні і форма листа голкоподібна і конфігурація – 2 рівних ряди і сріблясta смуга, то родина соснові.

Якщо клас голонасінні і форма листа голкоподібна і конфігурація – 2 рівних ряди і сріблястої смуги немає, то родина – болотний кипарис.

Якщо тип – дерева і форма листа широка і плоска – то покритонасінні.

Якщо тип – дерева і неправильно, що форма листа широка і плоска, то клас – голонасінні.

Якщо стебло зелене, то тип – трав'яні.

Якщо стебло дерев'янисте і стелиться, то тип – ліани.

Якщо стебло дерев'янисте і прямостояче і одне основне стебло, то тип – дерева.

Якщо стебло дерев'янисте і прямостояче і неправильно, що одне основне стебло, то тип – чагарники.

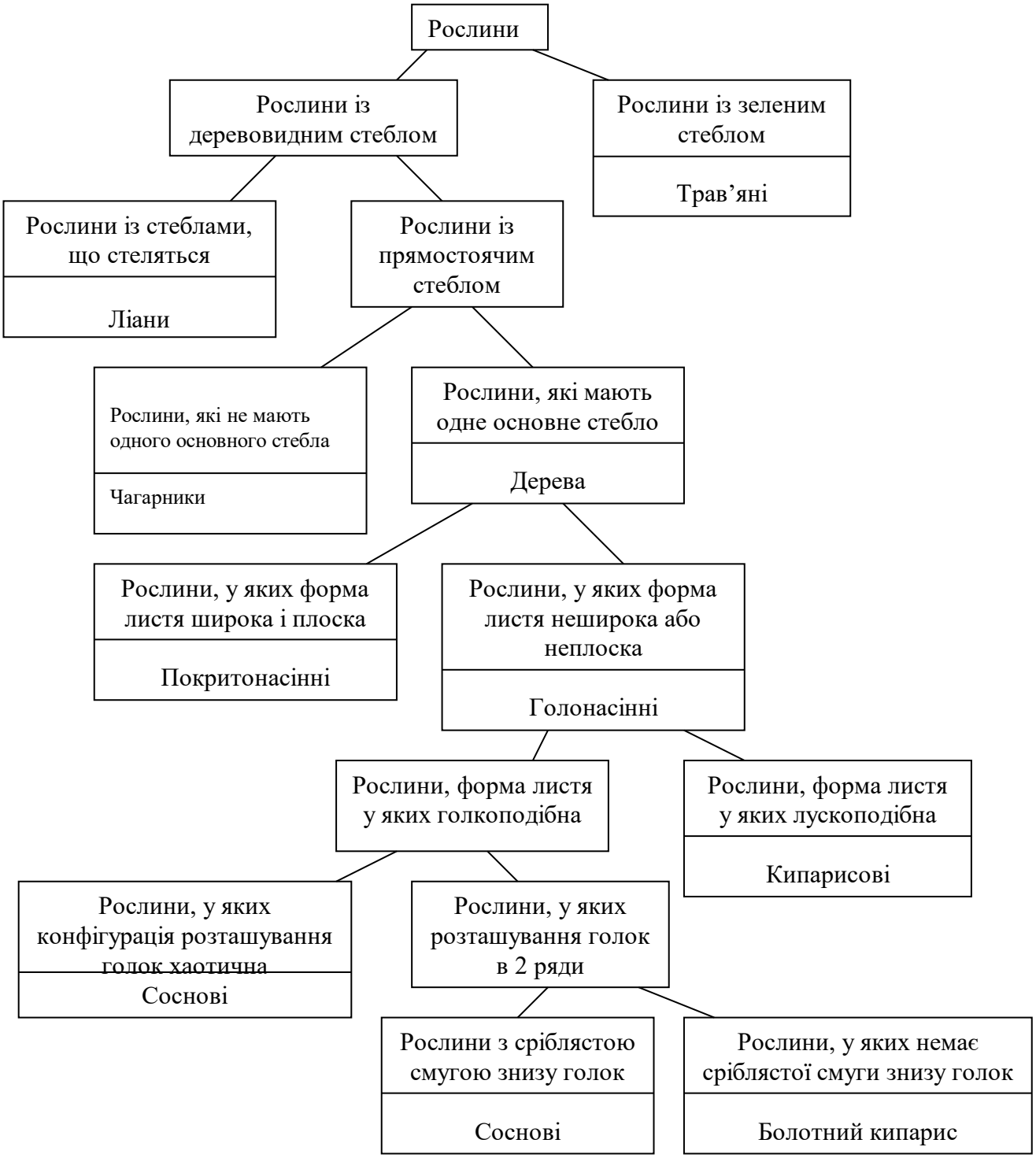
Орієнтовний розв'язок.

Знання про предметну галузь подані продуційними правилами. Зазначимо, що правило в загальному випадку записується так:

ЯКЩО $P_1, P_2, \ldots, P_n$ ТО B.

Такий запис означає, що "якщо всі умови від P_1 до P_n істинні, то B також істинне" або "якщо всі умови від P_1 до P_n виконуються, то слід виконати дію B".

Рис.К.1. Деревовидна структура (деревовидна інформаційна модель) бази знань експертної системи для ідентифікації рослин.



Проаналізувавши подані ботанічні описи, бачимо, що класифікація в базі знань може грунтуватися на деревовидній структурі (рис.К.1).

Згідно наведеної деревовидної класифікації рослини поділені на ті, у яких стебло зелене, і з древесним стеблом. Рослини із зеленим стеблом ідентифікуються відразу (тип – трав'яні), інші попадають в певні підмножини в залежності від своїх характеристик, властивостей (атрибутів). Тільки в одному випадку взята характеристика виявилася визначальною, в останніх для ідентифікації використовується цілий набір атрибутів.

В експертній системі наведеного прикладу використовуються такі характеристики: 1) стебло дерев'янисте; 2) стебло зелене; 3) стебло прямостояче; 4) стебло стелиться; 5) одне основне стебло; 6) не одне основне стебло; 7) форма листа широка і плоска; 8) форма листа нешироka або неплоска; 9) форма листа лускоподібна; 10) форма листа голкоподібна; 11) конфігурація розташування голок хаотична; 12) конфігурація розташування голок в 2 ряди; 13) сріблясta смуга знизу голки; 14) немає сріблястої смуги знизу голок.

Всі 14 перелічених характеристик є необхідними для ідентифікації рослин. Це тому, що ні одна з характеристик не характерна для всіх досліджуваних об'єктів одночасно. Зазначимо, що якщо деякий атрибут характеризує всі об'єкти одночасно, то він, як правило, не є необхідним. Кожна характеристика для конкретної категорії (типу, роду, родини) або правильна, або неправильна. Характеристики, справедливі для об'єктів, що визначаються в прикладі, наведені в таблиці К.1.

Таблиця К.1.

Тип, рід, родина	Характеристики
Трав'яні	2
Ліани	1, 4
Чагарники	1, 3, 6
Покритонасінні	1, 3, 5, 7
Кипарисові	1, 3, 5, 8, 9
Соснові	1, 3, 5, 8, 10, 11
або	1, 3, 5, 8, 10, 12, 13
Болотний кипарис	1, 3, 5, 8, 10, 12, 14
Дерева	1, 3, 5
Голонасінні	1, 3, 5, 8

При проектуванні бази знань експертної системи деревовидна структура, множина характеристик для ідентифікації об’єктів, набори номерів для кожного об’єкта складають робочу модель бази знань для вибору родини (або роду, або типу). За робочою моделлю бази знань легко побудувати відповідну графову модель (семантичну мережу) процесу ідентифікації об’єкта експертною системою (рис.К.2). На графовій моделі вузли для спрощення подані схематично.

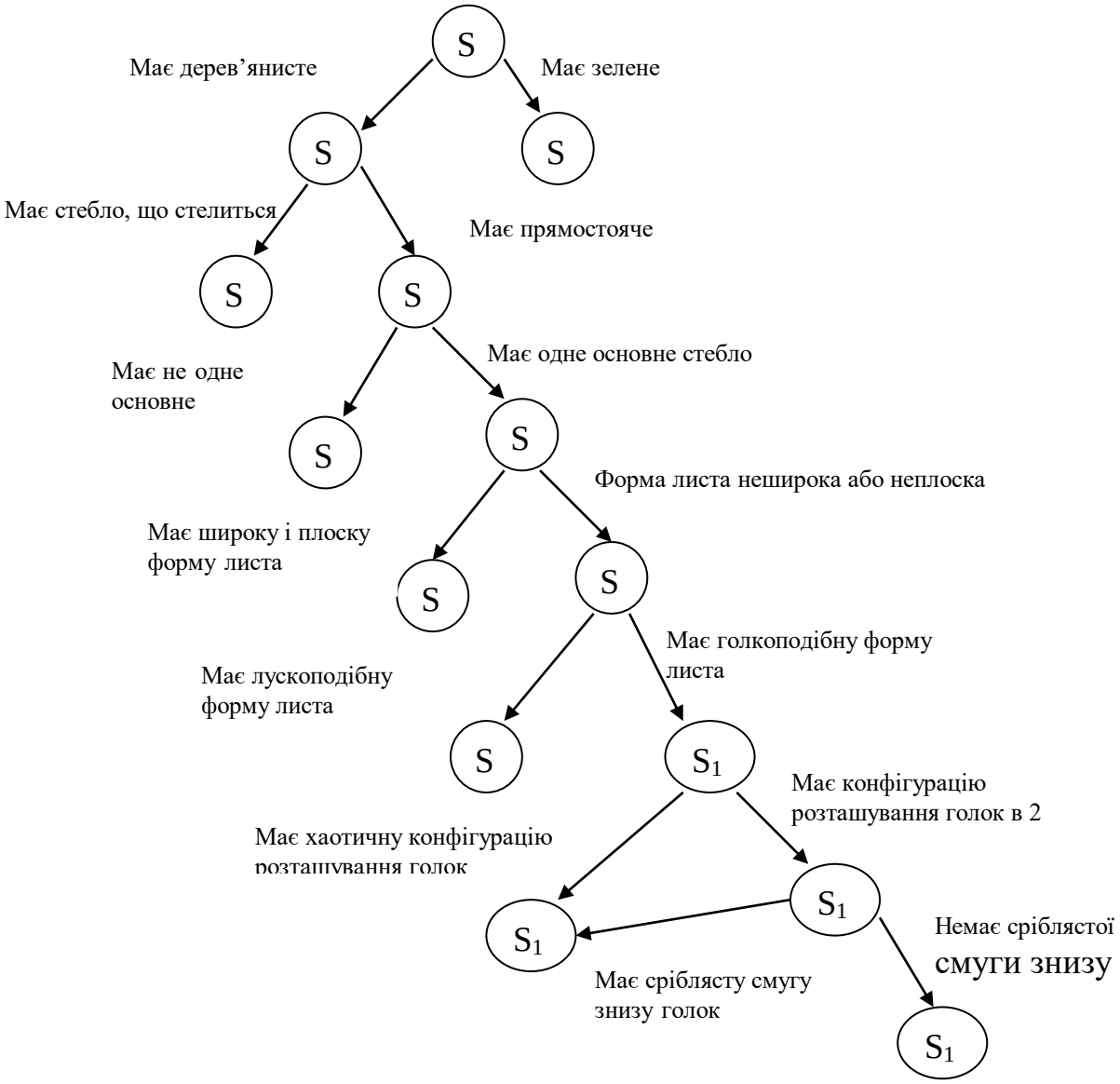


Рис. К.2. Графова модель ідентифікації рослин.

Кожний вузол на схемі (рис.К.2) відповідає певному етапу процесу ідентифікації об’єкта. Зазначимо, що оболонка DESS дозволяє створювати графові моделі не лише деревовидного типу, а й складнішої структури.

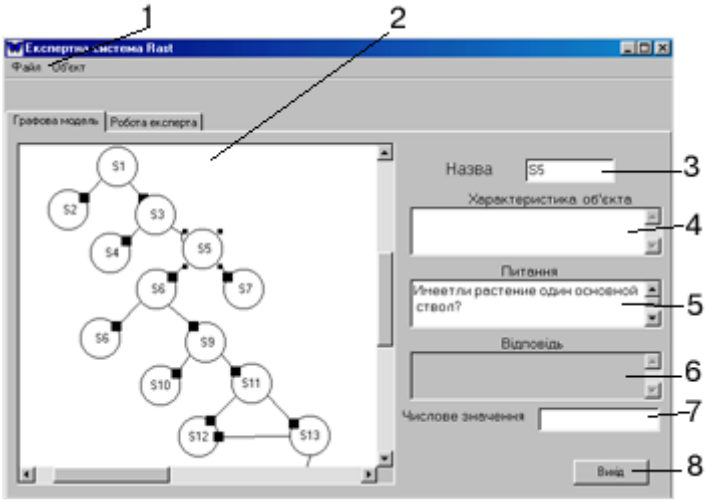


Рис.К.3. Режим розробки експертної системи: 1 – меню, 2 – область для побудови графової моделі системи, 3 – введення назви вершини графа, 4 – введення характеристики етапу діагностики, певних рекомендацій, 5 – введення чергового питання, 6 – введення можливої відповіді користувача, що відповідає певному ребру, 7 – введення числового значення вершини (якщо потрібно), 8 – кнопка виходу з програми

Оболонка DESS проста у використанні. На рис.К.3, К.4 наведені елементи її середовища. В оболонці для роботи використовуються тільки 2 пункти меню зі своїми підпунктами (1, рис.К.3), що дозволяє учням легко орієнтуватись в системі команд середовища. Пункт меню “Файл” має такі підпункти: “Створити” - створення нової експертної системи, “Відкрити” – відкриття існуючої експертної системи, “Записати” – збереження на диску створеної або зміненої експертної системи, “Закрити” – закриття відкритої експертної системи. Пункт меню “Об’єкт” має такі підпункти: “Створити” – створення нової вершини графа або нового етапу діагностики, “Вилучити” – вилучення з системи вибраної вершини графа, що призводить до вилучення певного етапу діагностики, “Встановити зв’язок” – встановлення зв’язку між двома вершинами графа (ребро), що символізує відповідь користувача, за якою відбувається перехід системи з одного етапу до іншого, “Вилучити зв’язок” – вилучення ребра графа, “Переглянути зв’язок” – перегляд відповіді користувача, що відповідає вибраному ребру, “Очистити” – очищення вмісту відкритої експертної системи.

Для розробки експертної системи необхідно виконати таку послідовність дій:

1. Перейти на закладку “Графова модель”, якщо не знаходитесь на ній, тим самим здійснюється перехід до режиму розробки та редагування експертної системи.

2. Через меню виконати команду Файл⇒Створити. В результаті цього на диску буде створений певний каталог з необхідними для системи файлами.

3. Далі можна приступити до створення графової моделі процесу діагностики, що здійснюється системою. Додавати чергову вершину необхідно через пункт меню Об’єкт⇒Створити. При цьому можна заповнити області: для введення назви (3, рис.К.3), для введення характеристики етапу діагностики та рекомендацій (4, рис.К.3), для введення питання (5, рис.К.3), для введення числового значення (7, рис.К.3). Клацнувши мишею по області (2, рис.К.3), можна включити вершину до зображення графової моделі процесу діагностики.

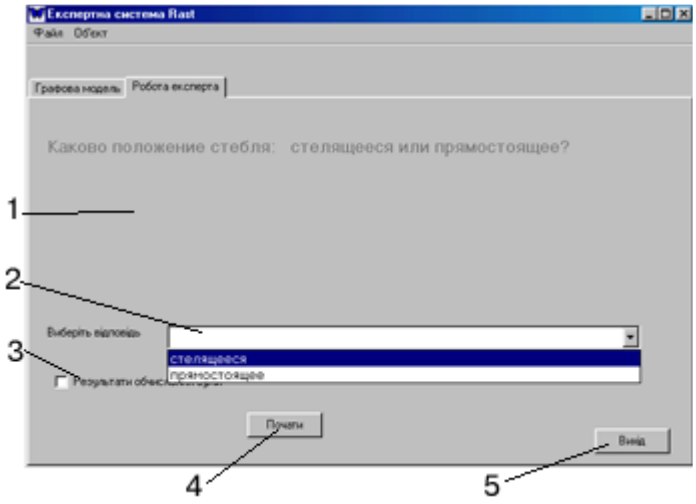


Рис.К.4. Режим роботи користувача з експертною системою: 1 – область для виведення питання, характеристики етапу діагностики, певних рекомендацій, 2 – вибір передбачених відповідей на питання, 3 – виведення суми числових значень вершин (якщо

4. Розмістивши на зображенні графової моделі кілька вершин, можна перейти до встановлення зв’язків між ними. Для цього необхідно через меню виконати команду Об’єкт⇒Встановити зв’язок. При цьому треба заповнити область (6, рис.К.3) можливою відповіддю користувача на питання, задане у вершині, звідки буде виходити ребро. Потім при натисненій лівій клавіші провести мишею від вершини поточного етапу діагностики до вершини – етапу, до

якого перейде система після відповідної відповіді користувача. На графі з’явиться зображення ребра.

5. Зберегти сформовану експертну систему, виконавши команду Файл⇒Записати. Далі можна перейти до користувацького режиму роботи з створеною експертною системою.

В ході розробки експертної системи її можна редагувати. Для зміни змісту питання, числового значення, характеристики об’єкта необхідно дворазово клацнути мишею на відповідній вершині графа. Зробити відповідні зміни у відповідних областях вікна (4, 5, 7, рис.К.3) і клацнути один раз мишею на зображенні графа. Нові дані про стан будуть занесені до системи. Для вилучення вершини графа – клацнути дворазово на вершині і через меню Об’єкт⇒Вилучити. Вибрана вершина зникне із зображення графової моделі разом зі своїми ребрами (зв’язками). Для ліквідації вже встановленого зв’язку (ребра) необхідно виконати команду Об’єкт⇒Вилучити зв’язок, провести при натисненій лівій клавіші мишею від однієї вершини до іншою. Вибране ребро зникне з графа, а зв’язок з системи. Можна переглянути зміст відповіді користувача, за якою передбачається перехід системи з одного стану до іншого. Для цього треба виконати команду Об’єкт⇒Переглянути зв’язок і провести при натисненій лівій клавіші мишею від однієї вершини до іншої. В області (6, рис.К.1) з’явиться передбачена відповідь користувача.

Для переходу в режим користувача необхідно перейти до закладки “Режим діагностики”. Якщо експертна система вже завантажена, то для запуску її на виконання достатньо натиснути кнопку “Почати” (4, рис.К.2). Якщо необхідна експертна система не завантажена, то необхідно її завантажити за допомогою команди Файл⇒Відкрити. Перед відкриттям нової системи, попередньо відкрити треба обов’язково закрити командою Файл⇒Закрити.

При роботі з системою в режимі користувача на кожному етапі (якщо відповідний вузол не є кінцевим) користувачу пропонується відповісти (точніше вибрати певну відповідь із запропонованих) на питання. Вибрана відповідь відповідає певному ребру, яке з'єднує вузол з наступним вузлом, наприклад: Яке стебло у рослини: дерев'янисте чи зелене? Стебло стоїть прямо чи стелиться? Відповідаючи на питання, користувач повідомляє про характеристики об'єкта. Процес консультації направляється по відповідному ребру. Коли об'єкт розпізнано (ідентифіковано), видається відповідне повідомлення про його тип, родину або рід.

Для оболонки DESS, що розглядається, є такі обмеження в роботі: кількість вершин в графі не повинна перевищувати 200, дві різні вершини може з'єднувати одне ребро.

Користуючись засобами експертної оболонки створюємо графову модель ідентифікації об'єктів експертною системою, записуємо питання, можливі відповіді на них тощо. Будемо використовувати такі позначення:

N. *Етап діагностики, на якому знаходиться система*.
N1)
… - етапи, на які переходить системи, в результаті відповідних відповідей,
Nn)

де N, N1, … Nn – імена етапів діагностики, що здійснюються системою.

Текстовий вміст графової моделі процесу діагностики, що здійснюється системою, наведеної на рис.К.2.

S0. Яке стебло у рослини: дерев'янисте чи зелене?

S2) зелене;
S1) дерев'янисте.
S2. Тип трав'яні. Питань більше немає.
S1. Яке положення стебла: він стелиться або стоїть прямо?
S3) стелиться;
S4) стоїть прямо.
S3. Тип ліани. Питань більше немає.
S4. Чи має рослина одне основне стебло?
S6) так;
S5) ні.
S6. Тип дерева. Чи має листя широку і плоску форму?
S7) так;

S8) ні.
S5. Тип чагарники. Питань більше немає.
S7. Клас покритонасінні. Питань більше немає.
S8. Клас голонасінні. Яка форма листа: лускоподібна чи голкоподібна?
S9) лускоподібна;
S10) голкоподібна.
S9. Родина кипарисові. Питань більше немає.
S10. Яка конфігурація розташування голок: хаотична чи в 2 ряди?
S11) хаотична;
S12) в 2 ряди;
S11. Родина соснові. Питань більше немає.
S12. Чи є сріблясta смуга знизу голки?
S11) так;
S13) ні.
S13. Родина болотний кипарис. Питань більше немає.

Оскільки експертна система є людинно-машинна система, вхідні данні для виводу, тобто факти, які мають місце в даній конкретній ситуації поступають в експертну систему від користувача. В процесі консультації система видає користувачу запит (питання) q_i, яке відповідає вершині S_i у вигляді альтернативного меню $q_i=\{a_{i1}Va_{i2}V\ldots Va_{ik}\}$, де V – знак “виключаючого Або”. Після вибору користувача якого-небудь факту a_{ij} перехід здійснюється по відповідному ребру до наступного вузла. Які саме факти можна запитати у користувача, встановлюється при розробці системи.

Приклади тем для створення експертних систем

1. *Визначення хвороби зернових культур.*

Якщо рослина уражена твердою сажкою, то уражений колос має інтенсивно зелений колір з синім відтінком.

Порошиста (летюча) сажка перетворює колос у чорну порошисту масу.

Бура іржа виявляється на стеблах і листках у вигляді довгастих іржаво-бурих порошистих подушечок, які пізніше набувають чорного кольору.

Жовта іржа уражає всі надземні частини рослини: листки, стебла, колоски. На уражених частинах рослини утворюються дуже дрібні лимонно-жовті порошисті урядопустули.

Аскохітоз уражує сходи, листки, стебла, на яких з’являються світло- або темно-коричневі плями, часто вдавлені.

2. Визначення хвороби овочевих культур.

Фітофтороз уражає всі надземні частини пасльонових рослин (картопля, помідор). На листках і стеблах з'являються бурі розпливчасті плями з буруватим нальотом спорошення на нижній поверхні листа.

При ураженні борошистою росою цибулі, на її листках з'являються жовті плями з сірувато-фіолетовим нальотом. При ураженні гарбузових культур з обох боків листків з'являється білий або рожево-сірий нальот.

Бактеріальний рак уражає пасльонові. На поперечному розрізі стебла судини мають вигляд темних крапок при дифузному ураженні. При місцевому ураженні плодів біля плодоножки виникають білі плями з темним центром подібно до "пташиного ока".

3. *Визначник дрібних хижаків.*

Якщо тварина маленька, з тонким тілом, черево, груди і горло її білі, а хвіст коротший за половину довжини тіла, одноколірний, то це - ласка.

Якщо тварина маленька, з тонким тілом, черево, груди і горло її білі, а хвіст приблизно дорівнює половині довжини тіла і кінець хвоста чорний, то це - горностай.

Якщо тварина півметра завдовжки. Забарвлення черева однотонне, блискуче, темно-коричневе. Голова приплюснута, вуха дуже маленькі, хвіст довший за половину довжини тіла, непухнастий, рівномірно звужується до кінця. Видра.

Якщо тварина менше, ніж півметра завдовжки (середніх розмірів). Тіло коричневе. Вуха досить великі, трикутні. На горлі й грудях велика біла, жовта або оранжева пляма. На губах і вусах білих плям немає. На горлі велика біла пляма. Куниця кам'яна.

Якщо тварина менше, ніж півметра завдовжки (середніх розмірів). Забарвлення тіла одноманітне, темно-буре. Вуха маленькі, заокруглені, без білих плям. На губах і горлі можуть бути невеликі білі плями. Норка.

4. *Приклади завдань для створення діагностичних експертних систем (ЕС) з математики.* Створити ЕС для визначення:

1) типу трикутника: прямокутний, рівносторонній, рівнобедрений;
2) геометричної фігури: квадрата, прямокутника, ромба, паралелепіпеда, трапеції;
3) фігур обертання: циліндр (прямий, нахилений), конус (прямий, нахилений), усічений конус, шар;
4) правильного багатогранника: правильний тетраедр, куб, октаедр, ікосаедр;
5) багатогранників: призма, паралелепіпед, куб, піраміда;
6) простих геометричних фігур: точка, пряма, відрізок, луч, площина.