DOI 10.31392/UDU-nc.series2.2024.23(30).11
УДК 371.3:510.22:004.4'7

Деканов Станіслав Якович
кандидат фізико-математичних наук, доцент,
доцент кафедри математичного аналізу, диференціальних рівнянь та теорії ймовірностей
Українського державного університету імені Михайла Драгоманова, м. Київ, Україна
ORCID ID: 0000-0002-0048-7192
s.ya.dekanov@udu.edu.ua

ВИВЧЕННЯ ЕЛЕМЕНТІВ ТЕОРІЇ МНОЖИН З ВИКОРИСТАННЯМ СИСТЕМИ КОМП'ЮТЕРНОЇ МАТЕМАТИКИ MATHEMATICA

Анотація. Однією з найпопулярніших та найпотужніших систем комп'ютерної математики є Mathematica. Її використовують як професійні науковці для своїх досліджень, так і викладачі й здобувачі освіти у процесі навчання. Будучи універсальним інструментом, Mathematica може використовуватися в різних галузях. Існує багато публікацій загального характеру про використання різних систем комп'ютерної математики (СКМ), зокрема Mathematica, для розв'язування математичних задач. Проте конкретній методиці вивчення окремих розділів математичного аналізу приділяється менше уваги. У цій статті запропоновано методику вивчення елементів теорії множин з використанням СКМ Mathematica. Показано, що за допомогою цієї СКМ можливо: 1) задавати множини різних типів, зокрема списки і області в $\mathbb{R}^1$ і $\mathbb{R}^2$; 2) перевіряти належність елемента до множини і включення множин; 3) виконувати операції об'єднання, перерізу, доповнення, віднімання; 4) зображувати множини графічно, зображувати круги Ейлера; 5) перевіряти правильність формул, які містять дії над множинами; 6) перевіряти правильність логічних тверджень. Описано порядок виконання перелічених дій. Наведено ілюстративні приклади. Використання такого інструменту як Mathematica не позбавляє необхідності добре знати теорію і самостійно розв'язувати задачі, а навпаки, стимулює до поглиблення знань, експериментування та дослідження. Водночас покращується комп'ютерна грамотність, алгоритмічне мислення, вміння програмувати. Значення цих результатів полягає у тому, що: 1) підвищується точність і надійність розв'язків задач; 2) підвищується ефективність розв'язування шляхом швидкого пошуку ідей, перевірки правильності окремих дій; 3) покращується розуміння матеріалу завдяки унаочненню; 4) розвивається алгоритмічне мислення і навички програмування; 5) зростає зацікавленість у вивченні матеріалу.

Ключові слова. Математичний аналіз, теорія множин, система комп'ютерної математики, Mathematica, Wolfram Cloud.

Вступ. Система комп'ютерної математики (СКМ) Mathematica була створена у 1988 році американським вченим Стівеном Вольфрамом. Згодом він заснував компанію Wolfram Research для продовження розробки своєї СКМ. З тих пір Mathematica пройшла 36-літню еволюцію і насьогодні є однією з найпопулярніших та найпотужніших у світі. Її використовують як професійні науковці для своїх досліджень, так і викладачі й студенти у процесі навчання. Будучи універсальним інструментом, Mathematica може використовуватися в різних галузях, різними способами, у тому числі в галузі освіти. Загальним питанням використання СКМ Mathematica у математиці, фізиці, економіці тощо приділено значну увагу. Разом з тим розроблення конкретної методики навчання окремих розділів і тем з використанням Mathematica чи інших СКМ далеко не завершено. У цій статті наведено різноманітні приклади ефективного і систематичного використання Mathematica під час вивчення вступу до аналізу, а саме початків теорії множин.

Аналіз останніх досліджень і публікацій. СКМ Mathematica тривалий період використовується в освітньому процесі закордонних закладів освіти, насамперед США. За останні 10-15 років вона набула популярності в закладах вищої освіти України. Цьому сприяла і зростаюча популярність онлайн-сервісу Wolfram Alpha від тих же розробників. М. І. Жалдак і Г. О Михалін подали ідею про необхідність створення сучасних навчальних посібників з курсу математичного аналізу з використанням СКМ. Водночас ставилось завдання детально, всебічно і широко демонструвати використання СКМ у кожній темі як з практичного боку, так і з теоретичного. У цьому напрямі було опубліковано ряд робіт (зокрема [1]), створено навчально-методичний посібник [2]. Автори, ймовірно, почали з

інтегрального числення як найбагатшого на застосування розділу. Разом з тим, інші розділи математичного аналізу теж містять багато можливостей застосування СКМ.

У дещо іншому контексті розглядали використання СКМ такі автори як О. О. Гриб'юк і В. Л. Юнчик. У роботі [3] вони аналізують, порівнюють між собою і вказують напрями використання кількох найпоширеніших математичних програмних засобів в контексті змішаного навчання.

І. В. Вакуленко у роботі [4] висвітлює на конкретних прикладах можливості ефективного використання деяких математичних програм для управління самостійною роботою майбутніх учителів інформатики під час навчання чисельних методів.

Ця стаття базується на матеріалах доповіді на конференції «Проблеми інформатизації навчального процесу в школі та вищому педагогічному навчальному закладі», що проходила в НПУ імені М.П. Драгоманова (нині Український державний університет імені Михайла Драгоманова) 10 жовтня 2017 року [5].

Метою написання статті є розроблення методики навчання майбутніх учителів математики теми «Елементи теорії множин» курсу математичного аналізу, що базується на використанні СКМ Mathematica 14.

Подання основного матеріалу. Робота з програмою Mathematica здійснюється у так званому робочому аркуші, або ноутбуці, на вигляд схожому на документ Word. Ноутбук складається з рядків введення команд (які позначаються In[1], In[2], …) і рядків виведення результатів (позначаються як Out[1], Out[2], …). На ці рядки можна робити посилання з інших рядків (рис. 1). Знак "%" позначає номер останнього рядка виведення. Для запуску на виконання команди в якомусь рядку введення потрібно натиснути комбінацію клавіш Shift + Enter. Під час наведення курсора на рядок виведення внизу цього рядка з'являється додаткове меню з пропозиціями щодо подальших перетворень.

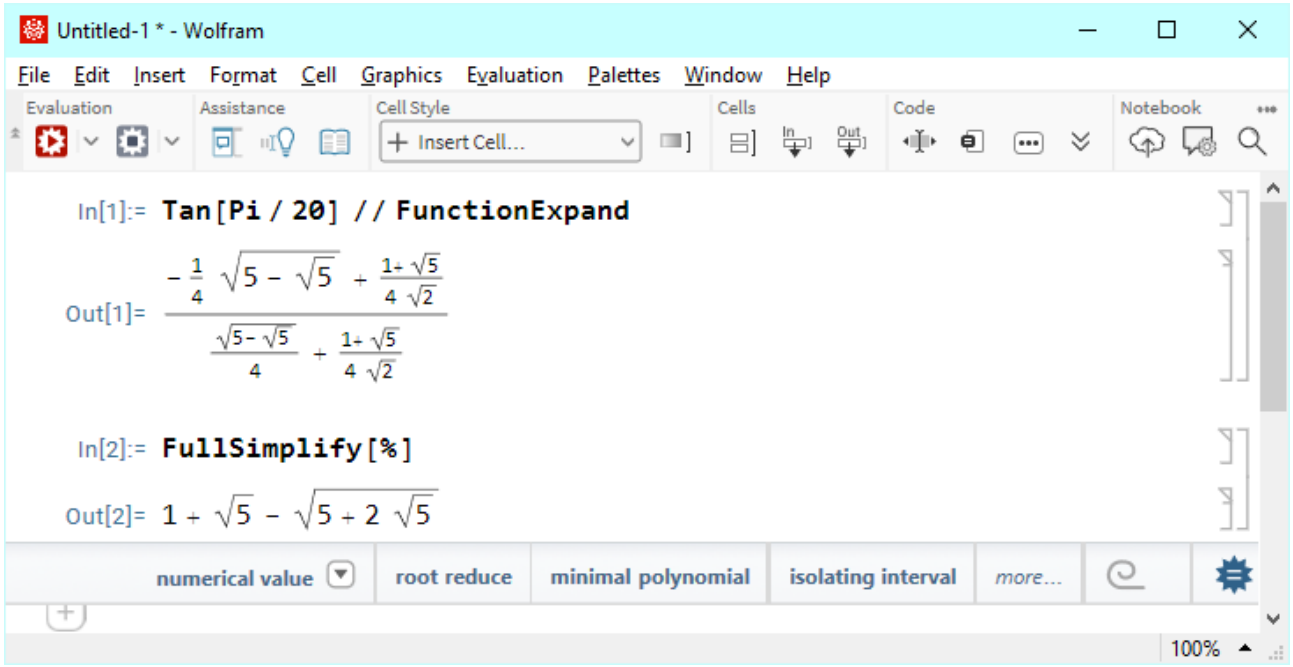


Рис. 1

Основними об'єктами в Mathematica є числа, змінні та функції. Функції поділяються на вбудовані і користувацькі. Є такі правила синтаксису: 1) назви всіх вбудованих функцій починаються з великої літери, 2) аргументи функцій записують у квадратних дужках, 3) фігурні дужки використовуються для введення списків або діапазонів. Круглі дужки використовують лише для групування членів виразів. Здебільшого вбудовані функції названі англійськими словами або словосполученнями, взятими із звичайної або математичної мови.

Сильною стороною Mathematica є її динамічна довідкова система (викликається клавішею F1), яка містить детальні описи всіх команд з прикладами і гіперпосиланнями.

Слід зазначити, що програма Mathematica 14 є комерційним продуктом, придбати який можна на сайті компанії [6]. Разом з цим, для популяризації Wolfram Language розробники надають безоплатний доступ до користування програмою Mathematica через хмарний сервіс Wolfram Cloud, щоправда з певними обмеженнями.

Система Mathematica містить розвинуті засоби для роботи з множинами. У ній є такі основні числові множини як: Primes (множина простих чисел), Integers (множина цілих чисел), Rationals (множина раціональних чисел), Reals (множина дійсних чисел), Algebraics (множина комплексних алгебраїчних чисел) і Complexes (множина комплексних чисел). Також є FullRegion[n] (множина $\mathbb{R}^n$), EmptyRegion[n] (порожня підмножина простору $\mathbb{R}^n$). Це у певному сенсі універсальні множини. Всі інші множини, які можна задати у даній СКМ, будуть переважно їх підмножинами. Розрізняються два основні типи множин – це списки (скінченні впорядковані набори) і області (множини з $\mathbb{R}^n$). Множини можна задавати, виконувати над ними операції, ілюструвати геометрично, використовувати їх в процесі обчислень. Завдяки логічним операціям, або так званій булевій алгебрі, у Mathematica можна виконувати операції і над довільними абстрактними множинами (що буде показано нижче).

Розглянемо детальніше роботу зі ***скінченними множинами***. Найбільш відповідним до поняття скінченної множини типом даних є список. Його задають командами List[a,b,…, c] або {a,b,…,c}, де аргументи можуть бути будь-якими об'єктами, які розпізнаються системою. Список, який не містить жодного елемента, називається *порожнім* і позначається {}. Слід мати на увазі, що список – це упорядкований набір елементів, і тому, наприклад, такі списки як {1,2}, {2,1}, {1,1,2} вважаються різними. У цьому є свої переваги. Разом з цим, якщо потрібно, можна дивитись на список як на множину, тобто не враховувати повторення елементів та їх порядок. Отже, почнемо роботу зі списками і задамо деякий список a:

```
In[1]:= a={1,4,6,9,12,15}
Out[1]= {1,4,6,9,12,16}
```

Для вибору зі списку a елемента з номером n використовують команду Part[a,n] або скорочений її варіант a[[n]]. Наприклад, виберемо четвертий елемент із заданого списку:

```
In[2]:= a[[4]]
Out[2]= 9
```

Для вибору зі списку a елементів з номерами від m до n використовують команду Part[a,m;;n] або скорочений її варіант a[[m;;n]]. Результат виводиться у вигляді нового списку. Наприклад, виберемо з заданого списку елементи з другого по четвертий:

```
In[3]:= a[[2;;4]]
Out[3]= {4,6,9}
```

Для вибору зі списку a елементи x, які задовольняють певний критерій P[x], використовують команду Select[a,P[#]&]. Наприклад, виберемо з заданого списку a всі елементи, які діляться на три:

```
In[4]:= Select[a,Mod[#,3]==0&]
Out[4]= {6,9,12,15}
```

Якщо потрібно знайти множину W всіх підсписків заданого списку $d=\{x_1,x_2,\ldots,x_n\}$, використовують команду Subsets[d]. Водночас зберігається порядок елементів і їх повторюваність. Наприклад:

```
In[5]:= d={1,3,3,3,2,2};
        Subsets[d]
        {{},{1},{3},{3},{3},{2},{2},{1,3},{1,3},{1,3},{1,2},{1,2},{3,3},
        {3,3},{3,2},{3,2},{3,3},{3,2},{3,2},{3,2},{3,2},{2,2},{1,3,3},{1,3,3},
        {1,3,2},{1,3,2},{1,3,3},{1,3,2},{1,3,2},{1,3,2},{1,3,2},{1,2,2},{3,3,3},
Out[6]= {3,3,2},{3,3,2},{3,3,2},{3,3,2},{3,2,2},{3,3,2},{3,3,2},{3,2,2},{3,2,2},
        {1,3,3,3},{1,3,3,2},{1,3,3,2},{1,3,3,2},{1,3,3,2},{1,3,2,2},{1,3,3,2},
        {1,3,3,2},{1,3,2,2},{1,3,2,2},{3,3,3,2},{3,3,2,2},{3,3,2,2},{3,3,2,2},
        {1,3,3,3,2},{1,3,3,3,2},{1,3,3,2,2},{1,3,3,2,2},{1,3,3,2,2},{3,3,3,2,2}, {1,3,3,3,2,2}}
```

У рядку In[5] після задання списку d поставлено оператор ";" для того щоб значення змінної d зайвий раз не виводилося на екран.

Перевіримо, скільки в останньому рядку було виведено підсписків списку d:

```
In[7]:= Length[%]
Out[7]= 64
```

Ця відповідь пояснюється так. Оскільки список d містить 6 пронумерованих елементів, то кількість усіх підмножин, утворених з цих елементів без їх перестановок, дорівнює $2^6 = 64$.

Як бачимо, список з повторюваними елементами трактується не як звичайна множина. Для того щоб перетворити список на множину, потрібно вилучити елементи, які повторюються. Це можна зробити за командами DeleteDuplicates[d] або Union[d]. Водночас остання з команд ще й відсортовує отриманий список. Застосуємо ці команди до заданого вище списку d:

```
In[8]:= DeleteDuplicates[d]
Out[8]= {1,3,2}
In[9]:= d=Union[d]
Out[9]= {1,2,3}
```

У рядку In[9] ми перепозначили список d, у результаті чого він став таким, як у рядку Out[9].

Команда Subsets має ще два варіанти. А саме: Subsets[d,n] видає за списком d усі його підсписки, що містять не більш ніж n елементів, а Subsets[d,{n}] знаходить усі n-елементні підсписки списку d:

```
In[10]:= Subsets[d,2]
Out[10]= {{},{1},{2},{3},{1,2},{1,3},{2,3}}
In[11]:= Subsets[d,{2}]
Out[11]= {{1,2},{1,3},{2,3}}
```

Над списками можна виконувати множинні операції: перевіряти належність елемента до списку, включення одного списку в інший, рівність двох списків як множин, знаходити об'єднання, переріз і різницю двох списків та ін.

Для прикладу задамо два списки a та b і виконаємо над ними вказані операції.

```
In[12]:= a={1,4,6,9,12,15}; b={2,4,7,10,15};
```

Належність елемента x до списку a перевіряють командою MemberQ[a,x]. Значенням цієї команди є True (так, твердження істинне) чи False (ні, твердження хибне). Закінчення "Q" у командах означає Question (запитання).

```
In[14]:= MemberQ[a,12]
Out[14]= True
```

Об’єднання, переріз і різницю списків a та b знаходять відповідно за командами Union[a,b], Intersection[a,b], Complement[a,b]. Водночас списки трактуються як множини і в результаті виводяться відсортовані списки без повторюваних елементів.

```
 In[15]:= Union[a,b]
Out[15]= {1,2,4,6,7,9,10,12,15}
 In[16]:= Intersection[a,b]
Out[16]= {4,15}
 In[17]:= Complement[a,b]
Out[17]= {1,6,9,12}
 In[18]:= Complement[b,a]
Out[18]= {2,7,10}
```

Включення списку b у список a (у розумінні включення множин), тобто умову $a \supset b$, перевіряють командою SubsetQ[a,b] або ContainsAll[a,b]. Наприклад,

```
 In[19]:= a={3,3,2,2,1,1,3};
          b={2,1,1,2,1};
 In[21]:= SubsetQ[a,b]
Out[21]= True
 In[22]:= ContainsAll[a,b]
Out[22]= True
```

Рівність двох множин, заданих списками a і b, можна перевірити двома способами, як показано у наступному прикладі. Перший спосіб:

```
 In[23]:= a={3,3,2,2,1,1,3};
          b={2,1,1,2,1,3};
 In[25]:= SubsetQ[a,b]&&SubsetQ[b,a]
Out[25]= True
```

У рядку In[25] символом “&&” задано логічну операцію “і”. Другий спосіб:

```
 In[26]:= Union[a]==Union[b]
Out[26]= True
```

У рядку In[26] знак “==” означає порівняння об’єктів між собою, на відміну від операторів присвоєння “=” чи “:=”.

Досі розглядався лише один спосіб задання списків – переліком його елементів. Насправді в Mathematica є й інші способи задання списків. Зокрема, для генерування списків часто використовують команди Range і Table з різними параметрами й опціями. Так, команда Range[m,n], де $m \le n$ – два довільних дійсних числа, створює список вигляду {m,m+1,…,m+k}, усі елементи якого належать відрізку [m;n], тобто їх останній номер k=[n-m] – ціла частина числа n-m.

```
 In[27]:= Range[-5,5]
Out[27]= {-5,-4,-3,-2,-1,0,1,2,3,4,5}
 In[28]:= Range[Sqrt[2],Sqrt[17]]
Out[28]= {√2, 1 + √2, 2 + √2}
```

Команда Range[m,n,h] виконує те саме, що й попередня, тільки з кроком h.

```
 In[29]:= Range[-5,5,3]
Out[29]= {-5,-2,1,4}
 In[30]:= Range[Sqrt[2],Sqrt[17],1/E]
Out[30]= {√2, √2 + 1/e, √2 + 2/e, √2 + 3/e, √2 + 4/e, √2 + 5/e, √2 + 6/e, √2 + 7/e}
```

У рядку In[30] літерою “E” позначено число $e = \lim_{n\to\infty}(1+1/n)^n$.

Команда Table[f[i],{i,m,n}] створює список зі значень функції f[i], де i=m,m+1,…,m+k, k=[n-m]. Взагалі ж, ця команда має багато варіацій.

```
In[31]:=  Table[i^2,{i,1,10}]
Out[31]=  {1,4,9,16,25,36,49,64,81,100}
```

Також новий список b можна отримати з уже побудованого списку a, застосовуючи функцію f. Для цього часто достатньо просто записати b=f[a], і функція f подіє на кожен елемент списку a. Наприклад, створимо список значень функції Sin[k], k=1,2,…,10, таким чином:

```
In[32]:=  listSin=Sin[Range[1,10]]
Out[32]=  {Sin[1],Sin[2],Sin[3],Sin[4],Sin[5],Sin[6],Sin[7],Sin[8],Sin[9],Sin[10]}
```

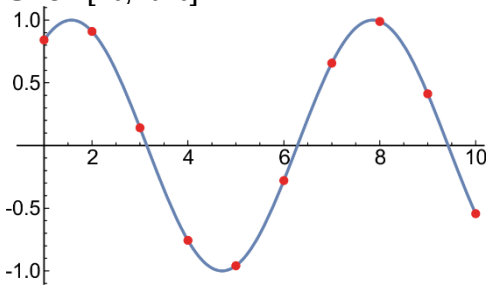
Список $a=\{y_1,y_2,\ldots,y_n\}$ з числовими значеннями можна зображати графічно у вигляді точок

$$M_k(k, y_k),\ k \in \overline{1,n},$$

що лежать на площині. Ці точки можна послідовно з’єднати прямолінійними відрізками або накласти на інший графік. Розглянемо приклад.

```
In[33]:=  ListPlot[listSin,PlotStyle->{PointSize[0.02],Red}];
In[34]:=  Plot[Sin[x],{x,1,10}];
In[35]:=  Show[%,%%]
```

Out[35]=



У рядку In[33] задано команду створити (але не виводити на екран) графік списку listSin, визначеного вище у рядку In[32]. Водночас вказано розмір і колір точок. У рядку In[34] задано команду побудувати синусоїду на відрізку [1;10] (і теж не показувати поки що). Команда Show у рядку In[35] означає показати на одному рисунку два графіки, створені у попередніх двох рядках. В результаті у рядку Out[35] виведено потрібний рисунок.

На цьому завершимо огляд оперування зі списками у Mathematica.

Перейдемо до так званих областей, або регіонів (Regions), які є підмножинами просторів $\mathbb{R}^1$, $\mathbb{R}^2$ … Цей тип даних призначений у першу чергу для задання геометричних об’єктів та виконання перетворень над ними. Разом з цим об’єкти типу Region мають аналітичний опис і можуть використовуватись у процесі обчислень. У СКМ Mathematica можна задавати і досліджувати десятки геометричних фігур. Зокрема, пряму, многокутник, коло, еліпс, многогранник, площину, круг, кулю, циліндр, конус і багато інших. Проте найбільший інтерес під час вивчення вступу до аналізу становлять команди ImplicitRegion і ParametricRegion, за допомогою яких можна задавати множини аналітично.

Розглянемо детальніше першу з них. Область A у просторі $\mathbb{R}^1$ задають так:

A=ImplicitRegion[P[x],{x}],

де P[x] – сукупність або система нерівностей з однією змінною. Наприклад, щоб задати множину $A = \{x : x^2(x^2-1)(x^2-1) \le 0\}$, потрібно записати:

```
In[1]:=  A=ImplicitRegion[x^2(x^2-1)(x^2-2)<=0,{x}];
```

Для перевірки належності числа x множині A використовують команди Element[{x},A] або RegionMember[A,{x}].

```
In[2]:= RegionMember[A,{0}]
Out[2]= True
In[3]:= Element[{2},A]
Out[3]= False
```

Команду Element[{x},A] можна задати у більш звичному для математики вигляді {x} ∈ A (знак ∈ набирається послідовним натисненням Esc el Esc).

```
In[4]:= {−1} ∈ A
Out[4]= True
```

Зауважимо, що елементом множини типу Region у просторі $\mathbb{R}^n$ вважається упорядкований набір $\{x_1, x_2, \ldots, x_n\}$, зокрема у просторі $\mathbb{R}^1$ це одноточковий набір $\{x\}$. Якщо ж розглядається стандартна числова множина, наприклад, Reals, то число x брати в дужки не потрібно:

```
In[5]:= Pi ∈ Reals
Out[5]= True
```

Наступною важливою операцією над множиною A є подання її у вигляді проміжків, якщо це можливо. Ця операція виконується за такою командою:

```
In[6]:= Reduce[Element[{x},A]]
Out[6]= −√2 ≤ x ≤ −1|| x == 0|| 1 ≤ x ≤ √2
```

Знак "||" у рядку Out[6] означає диз'юнкцію, тобто сполучник "або".

У заданні множини A крім рівнянь і нерівностей може бути умова належності елемента до деякої іншої множини. Наприклад, задамо множину $A = \{x \in \mathbb{Z} : 1/9 \le 3^x < 10\}$:

```
In[7]:= A=ImplicitRegion[1/9<=3^x<10 && x ∈ Integers, {x}];
```

і спростимо її:

```
In[8]:= Reduce[{x}∈A]
Out[8]= x == −2 || x == −1 || x == 0 || x == 1 || x == 2.
```

Визначимо ще одну множину і спростимо її:

```
In[9]:= A=ImplicitRegion[Log[1/x]/Log[1/3]<=1,{x}];
        Reduce[{x}∈A]
Out[10]= 0 < x ≤ 3
```

Над множинами типу Region можна виконувати операції об'єднання, перерізу та різниці за командами RegionUnion, RegionIntersection і RegionDifference відповідно. Для прикладу задамо дві множини A, B і виконаємо над ними зазначені операції:

```
In[11]:= A=ImplicitRegion[-1<x<=2,x]; B=ImplicitRegion[-2<=x<1,x];
         M1=RegionUnion[A,B]; M2=RegionIntersection[A,B];
         M12=RegionDifference[A,B]; M21=RegionDifference[B,A];
```

З'ясуємо, якими найпростішими нерівностями визначаються задані вище множини. Для цього виконаємо групове застосування команди Reduce до всіх цих множин:

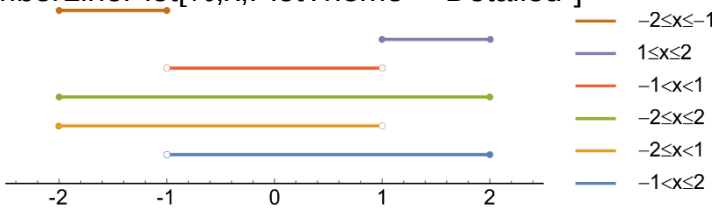
```
In[17]:= Map[Reduce[{x}∈#]&,{A,B,M1,M2,M12,M21}]
Out[17]= {−1 < x ≤ 2, −2 ≤ x < 1, −2 ≤ x ≤ 2, −1 < x < 1,

          1 ≤ x ≤ 2, −2 ≤ x ≤ −1}
```

Для зображення лінійної множини A ⊂ [a;b], що визначається предикатом P[x], використовують команду NumberLinePlot[P[x],{x,a,b}]. Якщо параметри a, b не вказати, то автоматично буде обрано найменший відрізок, який містить множину A. Під час застосування цієї команди до списку нерівностей виводяться зображення всіх відповідних множин на одному рисунку:

```
In[18]:= NumberLinePlot[%,x,PlotTheme->"Detailed"]
```

Out[18]=



Команда NumberLinePlot може мати багато опцій для налаштування вигляду рисунка. Зокрема, опція PlotTheme->"Detailed" відповідає за виведення детального опису кожної лінії справа від рисунка.

У просторі $\mathbb{R}^2$ область A задають аналогічно тому, як і в $\mathbb{R}^1$, а саме: і

A=ImplicitRegion[P[x,y],{x,y}],

де P[x,y] – сукупність або система нерівностей з двома змінними. Операції над областями в $\mathbb{R}^2$ виконуються за тими самими командами, які було розглянуто вище для областей в $\mathbb{R}^1$.

Перезапустивши сесію командою Exit, розглянемо деякі приклади оперування з множинами у просторі $\mathbb{R}^2$. Для початку задамо три круги, знайдемо їх об'єднання й переріз і зобразимо рисунок.

```
In[1]:= a=ImplicitRegion[(x+1/2)^2+y^2<1,{x,y}];
        b=ImplicitRegion[(x-1/2)^2+y^2<1,{x,y}];
        c=ImplicitRegion[x^2+(y+3/4)^2<1,{x,y}];
        d1=RegionUnion[a,b,c]; d2=RegionIntersection[a,b,c];
In[6]:= rys1= RegionPlot[{a,b,c},AspectRatio->Automatic,
        PlotLegends->{"a","b","c"}];
In[7]:= rys2=RegionPlot[d1,AspectRatio->Automatic,PlotStyle->
        {Opacity[0.5],Yellow},PlotLegends->{"a ∪ b"}];
In[8]:= rys3=RegionPlot[d2,AspectRatio->Automatic,PlotStyle->  Red,PlotLegends->{"a
        ∩ b"}];
In[9]:= Show[{rys1,rys2}]
Out[9]= див. рис. 2
In[10]:= Show[{rys1,rys3}]
Out[10]= див. рис. 3
```

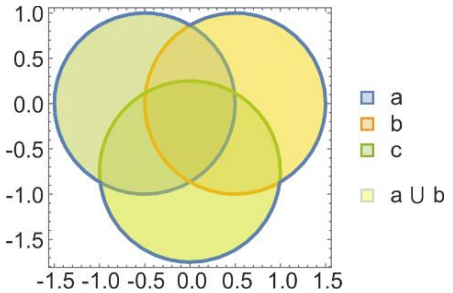


Рис. 2

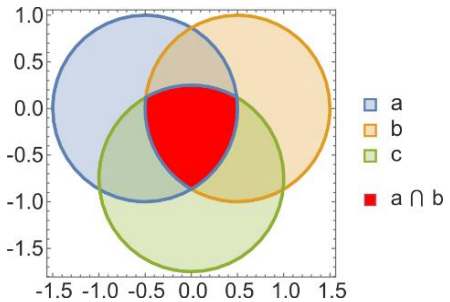
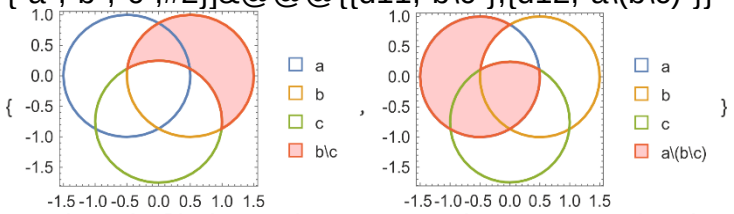


Рис. 3

Для створення графічного зображення області a в $\mathbb{R}^2$ призначена команда RegionPlot[a], яка може мати багато параметрів і опцій. Зауважимо, що за командою RegionPlot можна будувати лише області додатної міри. Саме цю команду було застосовано у рядках In[6] – In[8], тільки виведення проміжних графіків було заборонено оператором ";". Натомість за командою Show (у рядках In[9], In[10]) було виведено результуючі рисунки з накладеними один на одного двома графіками, а саме множини a, b, c та поверх них їх об'єднання і переріз.

Тепер продемонструємо на цих самих кругах виконання співвідношення a\(b\c)=a\((a∩b)\c):

```
In[11]:= d11=RegionDifference[b,c];
         d12=RegionDifference[a,RegionDifference[b,c]];
         d21=RegionIntersection[a,b];
         d22=RegionDifference[RegionIntersection[a,b],c];
         d23=RegionDifference[a,RegionDifference [RegionIntersection[a,b],c]];
In[16]:= RegionPlot[{a,b,c,#1},AspectRatio->Automatic,PlotStyle->
         {None,None,None,Lighter[Red,0.8]},PlotLegends->
         {"a","b","c",#2}]&@@@{{d11,"b\c"},{d12,"a\(b\c)"}}
```

Out[16]= 


```
In[17]:= RegionPlot[{a,b,c,#1},AspectRatio->Automatic,PlotStyle->
         {None,None,None,LightRred},PlotLegends->{"a","b","c",#2}]
         &@@@{{d21,"a∩b"},{d22,"(a∩b)\c"},{d23,"a\((a∩b)\c)"}}
Out[17]= див. рис. 4
```

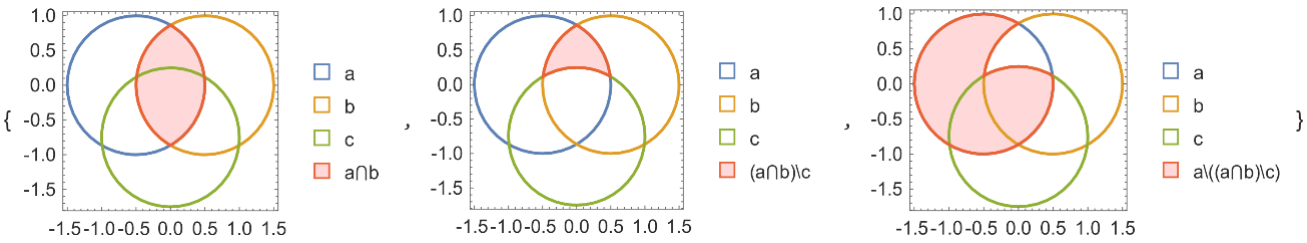


Рис. 4

У рядку In[11] покроково виконуються операції спочатку з лівої, а потім правої частини заданої рівності. У рядках In[15], In[17] вводяться команди послідовної побудови графіків з результатами операцій, виконаних вище. Для скорочення записів було введено слоти #1, #2, до яких послідовно підставлялися значення параметрів з вкладених списків, записаних у кінці команд. Зауважимо, що в математиці прийнято зображувати межі множин суцільною чи пунктирною лінією залежно від того, яка частина лінії належить множині, а яка – ні. На жаль, у команди RegionPlot немає опції, за якою така побудова виконувалася б автоматично. Тому за потреби користувач має самостійно уточнити вигляд межових ліній.

Далі перезапустимо сеанс і розглянемо команду, за якою перевіряють, чи містить область a у собі область b, тобто $a \supset b$, а саме: RegionWithin[a,b]. Наприклад:

```
In[1]:= ineq1=3x^2/2+2y^2<=5; ineq2=(2x)^2+y^2/2-x*y/2<=1;
        a=ImplicitRegion[ineq1,{x,y}]; b=ImplicitRegion[ineq2,{x,y}];
In[5]:= RegionWithin[a,b]
Out[5]= False
```

Ще один спосіб перевірити, чи виконується включення $a \supset b$, полягає у знаходженні різниці $b \setminus a$ і з'ясуванні, чи є вона порожньою. Застосування команди

FindInstance[P[x,y],{x,y}] дозволяє навіть знайти конкретну точку {x,y}, яка задовольняє умову P[x,y]. Якщо ж таких точок немає, результатом буде порожній список { }. Повернемось до нашого прикладу і визначимо різницю dif21=b\a, а в ній знайдемо яку-небудь (на вибір комп'ютера) точку dot1. В кінці побудуємо графічну ілюстрацію.

```
In[6]:= dif21=RegionDifference[b,a];
In[7]:= dot1=FindInstance[{x,y}∈dif21]
Out[7]= {{x → -1/9, y → -91/64}}

In[8]:= rys1=RegionPlot[{a,b},{x,-2,2},{y,-3/2,3/2},PlotStyle->{LightBlue,
        LightBrown},AspectRatio->Automatic,GridLines->Automatic, Axes->True];
In[9]:= rys2=Graphics[{PointSize[0.02],Red,Point[Values[dot1[[1]]]]}];
In[10]:= Show[rys1,rys2]
Out[10]= див. рис. 5
```

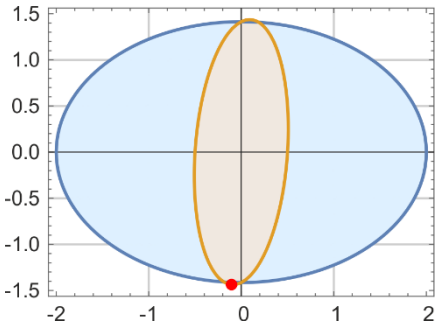


Рис. 5

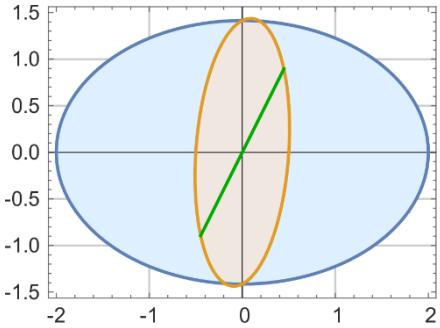


Рис. 6

У СКМ Mathematica можна перевірити включення множин ще й іншими способами. Зокрема, застосовуючи логічні операції або квантори, про що буде сказано трохи згодом.

Зазначимо ще одну особливість команд RegionMember і RegionWithin. За цими командами можна не лише перевіряти належність точки до області або включення областей, а й знаходити умови належності або включення. Нехай, скажімо, потрібно знайти відрізок прямої $y = 2x$, який міститься в заданій вище області *b*. Для цього достатньо виконати команду

```
In[11]:= RegionMember[b,{x,2x}]//Reduce
Out[11]= -1/√5 ≤ x ≤ 1/√5
```

Побудуємо цей відрізок разом з областями a, b:

```
In[12]:= ris3=Plot[2x,{x,-1/Sqrt[5],1/Sqrt[5]},PlotStyle->Darker[Green]];
In[13]:= Show[ris1,ris3]
Out[13]= див. рис. 6
```

Зауважимо, що множини можна задавати опосередковано певними умовами. За такого задання іноді навіть зручніше виконувати над ними операції і зображувати їх графічно. Потрібно лише мати на увазі таку таблицю відповідності між логічними і множинними операціями.

множинні операції	
назва	**позначення**
об'єднання	$A \cup B$
переріз	$A \cap B$
доповнення	$\overline{A}$

логічні операції		
назва	**позначення**	**у Mathematica**
диз'юнкція	$\vee$	And[a,b], a&&b
кон'юнкція	$\wedge$	Or[a,b], a||b
заперечення	$\neg$	Not[a], !a

Виконаємо цим способом ще одну вправу. Нехай потрібно зобразити на площині множини

$$A = \{(x, y) : y > \sqrt{x}\}, B = \{(x, y) : x^2 + (y - 1)^2 \le 1\},$$

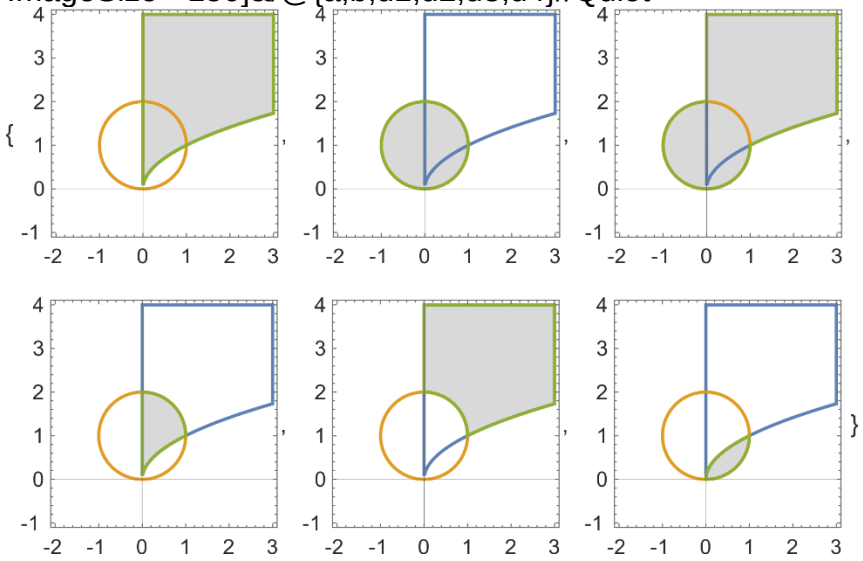
а також

$$A \cup B, A \cap B, A \backslash B \text{ і } B \backslash A.$$

Розв'язування цієї вправи в Mathematica матиме такий вигляд:

```
In[1]:= a=y>Sqrt[x]; b=x^2+(y-1)^2<=1;
        d1=a||b; d2=a&&b; d3=a&&!b; d4=b&&!a;
In[7]:= RegionPlot[{a,b,#},{x,-2,3},{y,-1,4},PlotStyle->{None,None,
        LightGray},Axes->True,PlotPoints->100,MaxRecursion->3,
        ImageSize->150]&/@{a,b,d1,d2,d3,d4}//Quiet
```

Out[7]=



Вище вже йшла мова про те, що у Mathematica можна оперувати з конкретними множинами, застосовуючи логічні операції. Завдяки тому, що в цій системі можна виконувати також операції над абстрактними висловленнями, з'являється можливість доводити або спрощувати загальні властивості операцій над множинами.

Перезапустимо сеанс Mathematica і розглянемо кілька прикладів.

Для початку доведемо закон двоїстості $\overline{A \cup B} = \overline{A} \cap \overline{B}$. Мовою логіки цей закон рівносильний такому твердженню:

$$\neg(a \vee b) \Leftrightarrow (\neg a) \wedge (\neg b).$$

Отже, уведемо останню формулу в рядок In[1] і нижче дамо запит, чи правильна вона.

```
In[1]:= !(a||b)⇔(!a&&!b);
In[2]:= TautologyQ[%]
Out[2]= True
```

Отже, твердження правильне. Зауважимо, що знак "⇔" набирається як Esc, equiv, Esc. Для перевірки правильності логічних формул використовують команду TautologyQ або LogicalExpand. Останню з них використовують також для спрощування логічних тверджень.

Доведемо розподільний закон об'єднання відносно перерізу:

$$(A \cap B) \cup C = (A \cup C) \cap (B \cap C).$$

```
In[3]:= TautologyQ[(a&&b)||c⇔(a||c)&&(b||c)]
Out[3]= True
```

Як відомо, одним із методів доведення тверджень є складання таблиць істинності. Цей метод має ефективну реалізацію у середовищі Mathematica. Так, повертаючись до доведення попередньої формули, складемо таблиці істинності її лівої і правої частин:

In[4]:= Prepend[Boole[BooleanTable[{a,b,c,a&&b,(a&&b)||c},{a,b,c}]],
{" a "," b "," c "," a∧b ","(a∧b)∨c"}]//Grid[#,Frame->All]&

Out[4]=

a	b	c	a∧b	(a∧b)∨c
1	1	1	1	1
1	1	0	1	1
1	0	1	0	1
1	0	0	0	0
0	1	1	0	1
0	1	0	0	0
0	0	1	0	1
0	0	0	0	0

In[5]:= Prepend[Boole[BooleanTable[{a,b,c,a||c,b||c,(a||c)&&(b||c)},
{a,b,c}]],{" a "," b "," c "," a∨c "," b∨c ","(a∨c)∧(b∨c)"}]//
Grid[#,Frame->All]&

Out[5]=

a	b	c	a∨c	b∨c	(a∨c)∧(b∨c)
1	1	1	1	1	1
1	1	0	1	1	1
1	0	1	1	1	1
1	0	0	1	0	0
0	1	1	1	1	1
0	1	0	0	1	0
0	0	1	1	1	1
0	0	0	0	0	0

Оскільки останні стовпчики в таблицях істинності лівої й правої частин однакові, то це означає, що $(a \wedge b) \vee c \Leftrightarrow (a \vee c) \wedge (b \vee c)$, а отже, і правильне співвідношення

$$(A \cap B) \cup C = (A \cup C) \cap (B \cap C).$$

Пояснимо призначення команд у рядках In[4], In[5]. Основною там є команда BooleanTable, яка видає список наборів усеможливих значень істинності заданих виразів в залежності від значень вказаних незалежних змінних. Для того щоб замість значень "True" чи "False" виводилися значення "1" і "0", відповідно, використовується функція Boole. Щоб додати до таблиці рядок заголовку, використовується команда Prepend. Нарешті, щоб список наборів вивести у вигляді розлінованої таблиці, в кінці задано команду Grid з опцією Frame->All.

Розглянемо кілька прикладів на спрощування логічних формул і відповідних формул для множин. Спростимо, наприклад, вираз $(A\backslash C)\backslash(B\backslash C)$. Врахуємо, що різниці множин $X\backslash Y$ відповідає логічна операція $x \wedge (\neg y)$.

In[6]:= LogicalExpand[(a&&!c)&&!(b&&!c)]

Out[6]= $a \;\&\&\; !b \;\&\&\; !c$

Отже, правильна рівність

$$(A\backslash C)\backslash(B\backslash C) = (A\backslash B)\backslash C.$$

Спростимо вираз

$$(A \cap B) \cup (A\backslash B) \cup (B\backslash A).$$

In[7]:= LogicalExpand[(a&&b)||(a&&!b)||(!a&&b)]

Out[7]= $a \;||\; b$

Дістаємо рівність $(A \cap B) \cup (A\backslash B) \cup (B\backslash A) = A \cup B$.

Нарешті, спростимо вираз $\overline{(\overline{A} \cup \overline{B})\backslash(A \cap B)}$.

```
In[8]:= LogicalExpand[!(!(a&&b)&&(!a||!b))]
Out[8]= a && b
```

Маємо рівність $\overline{(\overline{A} \cup \overline{B})\backslash(A \cap B)} = A \cap B$.

Окрім простих логічних операцій Mathematica підтримує логіку предикатів, тобто операції над висловленнями зі змінними, до яких можуть входити логічні квантори $\forall$ та $\exists$. Це значно розширює можливості застосування цієї СКМ у різних сферах, зокрема і в теорії множин.

Квантору $\forall$ відповідає команда ForAll, а квантору $\exists$ – команда Exists. Для спрощування тверджень з кванторами служать команди Resolve, Solve або Reduce. Так, наприклад, включення множини

$$A = \left\{(x,y) : 4x^2 - \frac{xy}{2} + \frac{y^2}{2} \le 1\right\} \text{ у множину } B = \left\{(x,y) : \frac{3x^2}{2} + 2y^2 \le 5\right\}$$

можна довести такими способами. По-перше, можна показати, що з першої нерівності випливає друга:

```
In[9]:= R1=(2x)^2+y^2/2-x y/2<=1; R2=3x^2/2+2y^2<=5;
In[11]:= ForAll[{x,y},R1⇒R2]//Resolve
Out[11]= True
```

Те саме дещо в іншій формі:

```
In[12]:= ForAll[{x,y},R1,R2]//Resolve
Out[12]= True
```

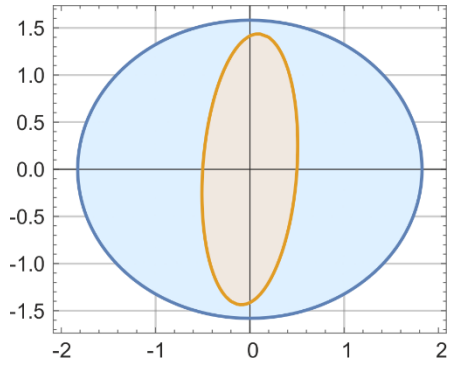
По-друге, – показати, що не існує точок, які задовольняють першу нерівність, але не задовольняють другу:

```
In[13]:= Exists[{x,y},R1&&!R2]//Resolve
Out[13]= False
```

Зобразимо розглянуті вище множини, щоб вочевидь переконатися, що $A \subset B$:

```
In[14]:= RegionPlot[{R2,R1},{x,-2,2},{y,-5/3,5/3},PlotStyle->{LightBlue,
         LightBrown},AspectRatio->Automatic,GridLines->Automatic, Axes->True]
```

Out[14]=



Використовуючи квантори, можна виконувати операції не тільки над скінченною кількістю множин, а й над нескінченною. Ось як, наприклад, можна знайти об'єднання $A = \bigcup_{n\ge1} E_n$ і переріз $B = \bigcap_{n\ge1} E_n$ піввідрізків $E_n = \left[\frac{2-3n}{n+1}; \frac{n^2+1}{n^2+n+1}\right)$, $n \in [1; +\infty)$.

```
In[1]:= A=Exists[n,n>=1,(2-3n)/(n+1)<=x<(n^2+2)/(n^2+n+1)];
In[2]:= Resolve[A]
Out[2]= −3 < x < 1
In[3]:= B=ForAll[n,n>=1,(2-3n)/(n+1)<=x<(n^2+2)/(n^2+n+1)];
In[4]:= Resolve[B]
Out[4]= −1/2 ≤ x < 2/3 (3 − √3)
```

Отже,

$$A = (-3\,;1),\ B = \left[-\tfrac{1}{2};\tfrac{2}{3}\left(3-\sqrt{3}\right)\right).$$

Зауважимо, що множини E_n проіндексовані дійсним параметром n. Якщо накласти умову, що параметр $n \in \mathbb{Z}$, то відповідь не вдасться отримати цим способом.

Висновки. За допомогою СКМ Mathematica можна розв'язувати задачі з теорії множин практично будь-якого типу, причому часто різними способами і з графічними ілюстраціями. У даній статті було показано, як це робиться. Такий інструмент як Mathematica не позбавляє необхідності добре знати теорію і самостійно розв'язувати задачі, а навпаки, стимулює до поглиблення знань, експериментування та дослідження. Одночасно відбувається підвищення рівня комп'ютерної грамотності, розвиток алгоритмічного мислення та вдосконалення навичок програмування.

Значущість цих результатів полягає у тому, що: 1) забезпечується зростання точності та надійності під час розв'язування задач; 2) підвищується ефективність розв'язування завдань завдяки оперативному пошуку ідей та перевірці правильності окремих етапів; 3) покращується розуміння матеріалу через його наочне подання; 4) формується алгоритмічне мислення та розвиваються програмістські вміння; 5) зростає мотивація до вивчення матеріалу.

Список використаних джерел:

[1] Жалдак М. І., Михалін Г. О., Деканов С. Я. Навчання майбутніх учителів математики інтегрального числення функцій однієї змінної з використанням комп'ютерних засобів математики. *Науковий часопис НПУ імені М. П. Драгоманова. Серія № 2. Комп'ютерно-орієнтовані системи навчання*. 2011. № 10 (17). С. 3–25.

[2] Жалдак М. І., Михалін Г. О., Деканов С. Я. Математичний аналіз. Інтегральне числення функцій однієї змінної з елементами інформаційних технологій*:* навч. посіб. Київ: НПУ імені М. П. Драгоманова, 2013. 268 с.

[3] Гриб'юк О. О., Юнчик В. Л. Використання систем комп'ютерної математики у контексті моделі змішаного навчання *Математика. Інформаційні технології. Освіта*. СНУ імені Лесі Українки. Луцьк – Світязь, 2015. С. 52–71.

[4] Вакуленко І. В. Управління самостійною роботою майбутніх вчителів в процесі навчання інформатики з використанням систем комп'ютерної математики. *Науковий часопис Національного педагогічного університету імені М.П. Драгоманова. Серія 2. Комп'ютерно-орієнтовані системи навчання*. 2020. № 22 (29). С. 181–196. DOI: 10.31392/NPU-nc.series2.2020.22(29).25.

[5] Деканов С. Я. Вивчення теорії множин з використанням СКМ Mathematica: матеріали всеукраїнської науково-практичної конференції *Проблеми інформатизації навчального процесу в школі та вищому педагогічному навчальному закладі*. Київ: Вид-во НПУ імені М. П. Драгоманова, 10 жовт. 2017. С. 57–58. URL: http://enpuir.npu.edu.ua/handle/123456789/38765

[6] Wolfram Research. URL: http://www.wolfram.com/

STUDYING ELEMENTS OF SET THEORY USING THE COMPUTER MATHEMATICS SYSTEM MATHEMATICA

Stanislav Dekanov

Abstract. One of the most popular and powerful computer mathematics systems is Mathematica. It is used both by professional researchers for their studies and by educators and students in the learning process. Being a universal tool, Mathematica can be applied across various fields. There are many general publications about using different computer mathematics systems, including Mathematica, to solve

mathematical problems. However, less attention is given to specific methodologies for studying individual sections of mathematical analysis. This paper proposes a methodology for studying elements of set theory using Mathematica. It demonstrates that this system enables users to: 1) define sets of various types, including lists and regions in $\mathbb{R}^1$ and $\mathbb{R}^2$; 2) verify element membership in sets and set inclusions; 3) perform operations such as union, intersection, complement, and subtraction; 4) graphically represent sets and draw Euler circles; 5) verify the correctness of formulas involving set operations; and 6) check the validity of logical statements. The procedure for performing these actions is described, and illustrative examples are provided. In our view, a tool like Mathematica does not eliminate the need to understand the theory and solve problems independently. On the contrary, it stimulates more profound knowledge, experimentation, and exploration. It also enhances computer literacy, algorithmic thinking, and programming skills. The significance of these results lies in the following: 1) increased accuracy and reliability of problem-solving; 2) improved efficiency through rapid idea generation and verification of individual steps; 3) better comprehension of material through visualization; 4) development of algorithmic thinking and programming skills; and 5) greater engagement in studying the material.

Keywords: Mathematical analysis, set theory, computer mathematics system, Mathematica, Wolfram Cloud.

References (translated and transliterated)

[1] Zhaldak M. I., Mykhalin H. O., Dekanov S. Ya. (2011) Teaching future mathematics teachers' integral calculus of one-variable functions using computer mathematics tools. *Scientific Journal of National Pedagogical Dragomanov University. Series № 2. Computer-oriented learning systems*. № 10 (17). Pp. 3–25. (in Ukrainian)

[2] Zhaldak M. I., Mykhalin H. O., Dekanov S. Ya. (2013) Mathematical Analysis. Integral Calculus of One-Variable Functions with Elements of Information Technologies: textbook. Kyiv: NPU named after M. P. Dragomanov. 268 p. (in Ukrainian)

[3] Hrybiuk O. O., Yunchyk V. L. (2015) The use of computer mathematics systems in a blended learning model context. *Mathematics. Information Technologies*. *Education*. Lesya Ukrainka Volyn National University. Lutsk – Svityaz. Pp. 52–71. (in Ukrainian)

[4] Vakulenko I. V. (2020) Management of future teachers' independent work in studying computer science using computer mathematics systems. *Scientific Journal of Mykhailo Dragomanov. Series № 2. Computer-oriented learning systems.* № 22 (29). Pp. 181–196, doi: 10.31392/NPU-nc.series2.2020.22(29).25. (in Ukrainian)

[5] Dekanov S. Ya. (2017) Studying set theory using the Mathematica system: Proceedings of the All-Ukrainian Scientific and Practical Conference *Problems of Informatization of the Educational Process in Schools and Higher Pedagogical Institutions*. Kyiv: Publishing House of NPU named after M. P. Dragomanov. 10 Oct. 2017. Pp. 57–58. [Online]. Available: http://enpuir.npu.edu.ua/handle/123456789/38765 (in Ukrainian)

[6] Wolfram Research [Online]. Available: http://www.wolfram.com/ (in English)