**МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ**
**ДЕРЖАВНИЙ ВИЩИЙ НАВЧАЛЬНИЙ ЗАКЛАД**
**«ДОНЕЦЬКИЙ НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ»**
**КАФЕДРА «ЕЛЕКТРИЧНОЇ ІНЖЕНЕРІЇ»**

# Методичні вказівки до виконання лабораторнихробіт «Методи розв'язання математичних задач в енергетиці на базі програмного пакету Maple»

**Красноармійськ – 2015**

УДК №

ББК №

Методичні вказівки до виконання лабораторних робіт «Методи розв’язання математичних задач в енергетиці на базі програмного пакету Maple» Колларов О.Ю., Сторож В.В., Власенко М.М. – Красноармійськ: ДонНТУ, 2015 рік. 54 сторінки.

Методичні вказівки містять теми лабораторних занять, алгоритми розрахунків і методичні вказівки до їх виконання. Допоможуть студентам закріпити теоретичний матеріал курсу.

Укладачі: _____ Колларов О.Ю., в.о. зав. каф. «Електричної інженерії»,

_____ Сторож В.В., доц. каф. «Електричної інженерії»,

_____ Власенко М.М., доц. каф. «Електричної інженерії».

Рецензент: _____ Артеменко Ю.А., доц. каф. «Електричної інженерії»

Відповідальний за випуск: _____ Колларов О.Ю., в.о. завідувача кафедри «Електричної інженерії», к.т.н., доцент кафедри

Затверджено навчально-методичним відділом ДонНТУ,
протокол № _____від_____

Розглянуто на засіданні кафедри «Електричної інженерії»,
протокол №_____ від _____.

**ЗМІСТ**

# Вступ

В останні роки показником інтелектуальних можливостей комп'ютерів стали новітні програмні системи символьної математики комп'ютерної алгебри. Створені для проведення символьних перетворень математичних виразів, ці системи були доведені до рівня, що дозволяє значно полегшити, а часом і замінити працю математиків, причому як теоретиків, так і математиків, що працюють в прикладній галузі. Вже з'явилися математичні відкриття, зроблені за допомогою таких програм. Навряд чи сьогодні є хоча б один серйозний науковий проект, зв'язаний з математичними дослідженнями, де б не використовувались системи комп`ютерної математики.

Системи символьної математики довгий час були орієнтовані на великі комп'ютери. З появою персональних комп'ютерів та зростанням їх можливостей, ці системи були адаптовані для персональних комп`ютерів і доведені до рівня масового комерційного продукту.

Система **Maple** була створена групою вчених, керівництвом **Кейта Геддеса (Keith Geddes)** і **Гастона Гонэ (Gaston Gonnet)** у 1980 році, в університеті Waterloo, Канада. Спочатку вона була реалізована на великих комп'ютерах і пройшла довгий шлях апробації, увібравши в себе велику частину математичних функцій і правил їхніх перетворень, вироблених математикою за сторіччя розвитку. Є реалізації цієї програми на платформах ПК Macintosh, Unix, Sun, а зовсім недавно спрощена версія Maple для операційної системи Windows. СЕ стала використовуватися в мініатюрних комп'ютерах фірми **Casio** за назвою **Cassiopeia**. Цій системі присвячені сотні книг.

Особливо ефективним є використання програми **Maple** при навчанні математики. Найвищий «інтелект» цієї системи сполучається з прекрасними засобами чисельного моделювання і відмінними можливостями графічної візуалізації. Застосування системи **Maple**, можливо як при викладанні, так і при самоосвіті, причому від самих основ і до вершин математики.

Перше знайомство з програмою багатьох користувачів просто зачаровує, переконавши користувача у безмежних можливостях можливостей системи і відсутності якого –небудь систематизованого опису. В результаті багато користувачів відкладають вивчення системи на потім.

**Maple** — система комп'ютерної математики, розрахована на широке коло користувачів. Донедавна її називали системою комп'ютерної алгебри, що вказувало на особливу роль символьних обчислень і перетворень, що здатна здійснювати ця система. Але така назва звужує сферу застосування системи. Насправді, по-перше, вона здатна виконувати швидко й ефективно чисельні розрахунки, причому з необмеженою точністю, а подруге, її можливості виходять далеко за рамки алгебри і вже покривають, наприклад, велику частину математичного аналізу.

Система **Maple** має широку сферу застосування від моделювання складних фізичних об'єктів, систем і пристроїв, і до створення художньої графіки (наприклад, фракталів). Заслуженою популярністю система Maple користаються у вузов — понад 300 самих великих університетів світу взяли її на озброєння. А число тільки зареєстрованих користувачів системи вже давно перевищило мільйон.

**Maple** — одна із самих могутніх інтегрованих систем символьної математики. Лідерство ця система завоювала в гострій конкурентній боротьбі з 6 іншою математичною системою — Mathematica. Кожна з цих двох систем має свої особливості, але в цілому вони практично рівноцінні. Однак треба відзначити, що поява нових версій Maple від 7 до 9 означає черговий виток у змаганні цих систем за місце лідера світового ринку. Причому ривок цього разу зробила система **Maple**.

**Maple** — типова **інтегрована** система. Вона характеризується наступними властивостями:

**- має вбудоване ядро для перетворення математичних виразів;**

**- потужну довідкову систему з великою кількістю прикладів;**

**- чисельний і символьний процесори;**

**-візуалізація обчислень, наукова та інженерна графіка;**

**- бібліотека вбудованих функцій і додаткових пакетів;**

**- вбудована мова програмування.**

**Maple** може застосовуватися як для проведення як класичних, так і прикладних математичних досліджень майже в усіх галузях науки, які використовують математичні моделі, включаючи економіку, фізику, екологією, біологію, соціологію . Іншими словами,**Maple**- ефективний , інструмент для математичного моделювання.

## РОЗДІЛ 1Типи об'єктів системи MAPLE та операції з ними

Система **MAPLE** може виконувати операції з об'єктами різної математичної природи, зокрема числами, константами, строками, змінними, невідомими величинами, виразами, функціями, списками, множинами, масивами,задавання чисел та виконання дій з ними.У системі **MAPLE** можна задавати цілі числа. звичайні дроби, радікали, числа з плаваючою точкою, комплексні числа. Перші три типи чисел дозволяють виконувати над ними точні обчислення, не використовуючи заокруглення.Числа з плаваючою точкою є наближеними, у яких число значущих цифр є обмеженим і які використовуються для апроксимації точних чисел, зокрема дійсних.Комплексні числа можуть бути як точними, так і наближеними, в залежності від того, точними або наближеними є дійсна і уявна частина цих чисел.

### 1.1Цілі числа

Цілі числа задаються у вигляді послідовностей цифр від 0 до 9. Від'ємні числа задаються зі знаком (-) перед числом. Максимальне ціле число, яке можна задати у системі **MAPLE** обмежується величиною 228. Обчислення з цілими числами обмежуються арифметичними діями: додавання +, віднімання - , множення *, ділення /. та обчислення факторіалу !.

Окрім звичайних арифметичних операцій над цілими числами можна виконувати деякі команди:

-розвинення цілих чисел на прості множники: **ifactor();**

-обчислення частки двох чисел **iquo(m,n);**

-обчислення залишку ділення двох чисел **irem(m,n);**

-перевірка, чи є ціле число простим **isprime(m);**

```
> ifactor(34276);                    (2)^2 (11) (19) (41)
> iquo(49,8);                        6
> irem(49,8);                        1
> isprime(34277);                    false
> isprime(347);                      true
```

## 1.2 Звичайні дроби

Звичайні дроби задаються за допомогою операції ділення двох цілих чисел m/n. У випадку, коли дріб скорочується, система **MAPLE** проводить це скороченняі записує результат. Якщо запис результату у вигляді звичайного дробу не дужезручний, то його можна перетворити у десятковий дріб за допомогою оператора**evalf(m/n,r),** де параметр r задає кількість десяткових знаків мантиси. За замовченням ця кількість дорівнює 10 знакам.

**> 65/7+2/3; 209**

**> evalf(%,3); 9.95**

У останньому прикладі символ % у першому аргументі функції evalf вказує на результати попереднього обчислення.

## 1.3Радикали

Радикали задаються як результат піднесення у дробову степінь цілих або дробових чисел, а також обчислення з них квадратного кореня.

**> 1/4*3**(2/3)*4**(2/5); 3(2/3 )(2/5 )**

Система **MAPLE** прагне до виконання дій над радикалами точно, дивись наступний приклад:

**> sqrt(3+2*sqrt(2))-sqrt(3-2*sqrt(2));2**

### 1.4 Дійсні числа с плаваючою точкою.

Дійсні числа з плаваючою точкою задаються у вигляді цілої і дробової частини, що розділені між собою десятковою точкою 24.3456 і перед яким може знаходитись знак + або -.

Часто числа з плаваючою точкою задаються у експоненціальному вигляді,наприклад: 24.347 e-4 або 5.357 e 5. Частина числа, що знаходиться перед знаком e, називається **мантисою** числа, яка множиться на 10 у степені значення якої розташоване за знаком e.

### 1.5Комплексні числа

Комплексні числа задаються в системі **MAPLE** у вигляді суми дійсної та уявної, яка містить спеціальну константу **I**, наприклад:

**> (2/3+3*I); +23 3 I**

Над комплексними числами можна виконувати арифметичні операції: +, -, *,/

**> (2/3+2*I)+(1/3-I); 1 + I**

**> (2/3+2*I)-(1/3-I); +13 I**

**> (2/3+2*I)*(1/3+I); +-643 I**

**> (2/3+2*I)/(1/3-I); +-8565 I**

Для отримання з комплексного числа його дійсної та уявної частини можна використовувати функції **Re()** та **Im().** Комплексно спряжене число можнаотримати за допомогою функції **conjugate().** Модуль і аргумент комплексногочисла можна отримати за допомогою функцій **abs(),та argument():**

```
> conjugate(-8/5+6/5*I);
> abs(-8/5+6/5*I);
> argument(-8/5+6/5*I);
> Re(-8/5+6/5*I);Im(9-8/5+6/5*I);
```

$$\frac{-8}{5}-\frac{6}{5}I$$

$$2$$

$$-\arctan\left(\frac{3}{4}\right)+\pi$$

$$\frac{-8}{5}\,;\,\frac{6}{5}$$

Для обчислення над комплексними числами використовується функція **evalc().**

> evalc(argument(((1/2)^(1/2)-I*(1/2)^(1/2))^(1/4))) − π16

## 1.6Задавання констант

Система **MAPLE** використовує набір загально прийнятих констант, таких як:

**-false- “істина”;**

**-true- “хиба”;**

**-gamma- константа Ейлера;**

**-Pi- число π;**

**-I- уявна одиниця ;**

**-infinity- нескінченість** ∞ .

Константи **Pi** та **gamma** можна обчислювати з певною точністю, використовуючи функцію **evalf()**

**> evalf(Pi,7);**

**> evalf(gamma,12);**

## 1.7Задавання стрічок

Система **MAPLE** дозволяє працювати зі стрічками – довільним набором символів, який записується у подвійні лапках. Кожний символ в стрічці представляє самого себе. Довжина стрічки практично не обмежена і може досягати майже 268 млн. символів. За допомогою функції length() можна обчислити довжину стрічки. **MAPLE** трактує стрічку як масив символів, тому є можливість звернутися як до окремого символу масиву, так і до частини стрічки.

**> S:="FRAZA";**

**> length(S);**

**> S[3];**

**> "FRAZA"[2..4];**

Для об'єднання двох стрічок в єдину використовується операція конкотенації || або функція **cat().**

**> "FRAZA"||"235";**

**> cat( "FRAZA","235");**

### 1.8 Змінні, невідомі, вирази та функції в MAPLE

Змінні в системі **MAPLE** мають своє ім’я, яке починається з літери. Великі і малі літери в імені змінної відрізняються між собою. Імена змінних не повинні містити пробілів, які можна заміняти знаком підкреслювання. Для іменування змінних не можна використовувати зарезервовані слова **MAPLE** та назви констант. При використанні цих слів для іменування змінних, система **MAPLE**повідомляє про помилку. Для того, щоб дізнатися про список зарезервованих слів,можна звернутися до довідки системи **MAPLE** за допомогою оператора **,? protect**.

Вирази у **MAPLE** представляють собою комбінацію імен змінних, чисел, функцій, що з’єднуються між собою знаками допустимих арифметичних операцій.

Якщо вираз використовує змінну, якій не присвоєно ні якого числового значення, то така змінна вважається невідомою величиною, а такий вираз називається символьним виразом. Саме для таких виразів і розроблялася система **MAPLE**.

Важливою операцією в **MAPLE** є операція присвоювання, яка має наступний синтаксис:

**змінна: = вираз.**

За замовченням кожна змінна **MAPLE** має тип **symbol**, представляє символьну змінну, а її значення м є її ім’я. При записі виразів система **MAPLE**дозволяє використовувати дуже багатий набір функцій. Елементарних:(експоненціальна, логарифмічна, квадратний корінь, модуль, знак числа, факторіал, тригонометричні функції, гіперболічні функції, обернені тригонометричні функції) та спеціальних функцій: (функцій Бесселя, еліптичних ін-

тегралів, Дірака, гамма та інші). Для знаходження необхідної функції користувач може використати довідникову систему **MAPLE** за допомогою команди **?inifunction**.

Для задавання функції користувача та її багаторазового використання можна використовувати функціональний оператор у вигляді **vars->result**.:

**> ff:=x->(sin(x)+cos(x))/x^2;**

**> evalf(ff(2));**

**> ff1:=(x,y)->sin(x)*sin(y)/(sin(x)+sin(y));**

**> evalf(ff(2,3));**

Для створення функцій, що приймають різні аналітичні вирази у різних областях зміни незалежної змінної, так званих кускових функцій у системіMAPLE використовується спеціальний оператор

**piecewise(cont_1, f_1, cont_2, f_2, ….cont_n, f_n, ),**

де cont_1,cont_2,…cont_n – логічні вирази, f_1,f_2,…f_n,f_other – вирази, що задаютьфункціональні залежності.

```
> piecewise(-infinity <x and x<=0,x^2,
0<x,x);
```

$$\begin{cases} x^2 & -\infty < x \text{ and } x \le 0 \\ x & 0 < x \end{cases}$$

```
> piecewise(x^2+y^2<=1, x^2+y^2,
1/sqrt(x^2+y^2));
```

$$\begin{cases} x^2+y^2 & x^2+y^2 \le 1 \\ \dfrac{1}{\sqrt{x^2+y^2}} & otherwise \end{cases}$$

## 1.9Складні типи виразів системи MAPLE

Крім перелічених вище простих типів даних, система **MAPLE** ефективно працює з складними типами даних. До яких відносяться :

1.Послідовності виразів;

2.Списки виразів;

3.Множини виразів;

4.Масиви;

5.Матриці.

Складні типи даних утворюються з простих типів даних за допомогою спеціальних синтаксичних конструкцій.

## 1.10 Послідовності виразів

**Послідовність виразів** є самостійним базовим об'єктом **MAPLE**, за допомогою якого будуються інші складні об’єкти. Послідовність виразів , або просто послідовність, - це група виразів, що розділена комами. Базовою властивістю послідовності є збереження порядку слідування виразів.

Послідовність може містити вирази, що повторюються, які розташовані удовільному порядку. На послідовність можна дивитися як на послідовність виразів, перенумерованих натуральними числами починаючи з одиниці.

Для послідовності її тип має ім’я **exprseq**. Для задавання послідовності використовується оператор присвоювання, у правій частині якого використовується послідовність виразів, або в лівій частині використовується послідовність змінних.

Для послідовностей є можливість звертатися як до послідовності в цілому, так і до окремих її елементів, вказуючи у квадратних дужках поруч з іменем змінної номер елемента послідовності, або діапазон номерів послідовності, що вказується у квадратних дужках у вигляді **[n..m].**

В той же час присвоїти нове значення елементу послідовності,використовуючи індексну форму елемента послідовності у лівій частині оператора, не можна.

Для задавання довгих послідовностей використовується оператор **seq** з таким синтаксисом:

**-seq(f, i= n..m);**

**-seq(f, i= s);**

де f - вираз, що залежить від змінної i ;

i - змінна, що визначає порядковий номер елемента послідовності;

n,m – числа, що задають діапазон зміни величини і з кроком 1 ;

s – може бути списком, множиною, послідовністю, стрічкою

```
> s1:=1,x,x^2,x^3;
```

$$s1 := 1, x, x^2, x^3$$

```
> whattype(s1);
```

$$exprseq$$

```
> s1[2..4];
```

$$x, x^2, x^3$$

```
> s1[2]:=y;
```

```
invalid assignment (1, x, x^2, x^3)[2]:= y;
  cannot assign to an expression sequence
```

```
> a,b,c,d:=s1;
```

$$a, b, c, d := 1, x, x^2, x^3$$

```
> a;b;c;d;
```

$$1$$
$$x$$
$$x^2$$
$$x^3$$

```
>   s2:=seq(1/(j^(3/2)),
j=1..10);
```

```
s2 := 1, 1/4*2^(1/2), 1/9*3^(1/2), 1/16*4^(1/2), 1/25*5^(1/2),
1/36*6^(1/2), 1/49*7^(1/2), 1/64*8^(1/2), 1/81*9^(1/2),
1/100*10^(1/2)
```

```
> s3:=seq(1/i,i=s1);
```

$$s3 := 1, \frac{1}{x}, \frac{1}{x^2}, \frac{1}{x^3}$$



## 1.11Списки та множини

**Списком** будемо називати впорядковану послідовність виразів, що записується у квадратні дужки. **Множиною**будемо називати не впорядковану послідовність елементів, записану у фігурних дужках. Якщо в послідовності присутні елементи, що повторюються, то кожний з них є окремим елементом списку, і одним і тим же елементом у множині.

Порядок елементів у списку е важливим і список зберігає порядок слідування елементів. За допомогою індексної форми списку можна звернутися до будь – якого елемента списку, та присвоїти йому нове значення. При задаванні множини елементів порядок елементів є неважливим і може змінюватися системою **MAPLE** самостійно.

Індексну форму можна використовувати і для елемента множини для вибору відповідного елемента множини, але за допомогою індексної форми не можна присвоїти елементу множини нове значення.

Для вибору декількох елементів списку або множини за допомогою індексної форми можна використовувати діапазон, при цьому додатне значення

індексу відповідає відліку елементів списку, або множини зліва направо, а від'ємне значення індексу з права наліво. Для задавання усіх елементів списку S можна записати індексний вираз S[1..-1], тобто індекс – 1 відповідає останньому елементу списку, а індекс –2 передостанньому елементу списку або множини.

```
> sp1:=[1,x,x^2,x^3,x^4];
```
$$sp1 := [1, x, x^2, x^3, x^4]$$

```
> sp1[2];
```
$$x$$

```
> sp1[3..-1];
```
$$[x^2, x^3, x^4]$$

```
> sp1[3]:=y^2;
```
$$sp1_3 := y^2$$

```
> sp1:=sp1;
```
$$sp1 := [1, x, y^2, x^3, x^4]$$

```
> Mn1:={a,b,c,a,b,c,d,f,a};
```
$$Mn1 := \{a, b, c, d, f\}$$

```
> Mn1[2..4];
```
$$\{b, c, d\}$$

```
> Mn1[4];
```
$$d$$

```
> Mn1[4]:=dd;
cannot reassign the entries in a set
```

Над множинами можна виконувати теоретико – множинні операції, а самеоперації об`єднання множин union, різниці множин minus, перетину множин intersect.

```
> Mn2:=Mn1 union {e,g};
```
$$Mn2 := \{a, b, c, d, f, e, g\}$$

```
> Mn3:= Mn2 minus {b};
```
$$Mn3 := \{a, c, d, f, e, g\}$$

```
> Mn4:=Mn2 intersect Mn3;
```
$$Mn4 := \{a, c, d, f, e, g\}$$

## 1.12Масиви

Для створення масиву у системі **MAPLE** необхідно використати оператор **array**(границі, список),де параметр границі задає діапазон, або діапазони зміни індексу (індексів) масиву.

Для багатовимірних масивів відповідні діапазониповинні задаватися підряд через кому. Параметрсписок, призначений длязадавання елементів масиву у вигляді вкладених списків з глибиною вкладання, що дорівнює роз-

мірності масиву. Перший і другий параметри функції array є необов'язковими, однак або перший або другий повинен бути заданий. Якщо значення елементу масиву не задані функцією array, то ці значення можна присвоїти, використовуючи індексну форму елементів масиву в лівій частині оператора присвоювання.

При введенні імені масиву **MAPLE** лише повторює його ім'я, але не зображає його елементи. Для отримання елементів масиву необхідно використати оператор **print**(ім'я_масиву), який приведе до зображення на лістингу **MAPLE** елементів масиву. Для двовимірних масивів з індексами, що починаються з одиниці зображення цих масивів здійснюється у вигляді матриць.

```
> MM:=array(1..4,1..4,[[1,2,3,4],
[2,3,4,5],[3,4,5,6],[4,5,6,7]]);
```

$$MM := \begin{bmatrix} 1 & 2 & 3 & 4 \\ 2 & 3 & 4 & 5 \\ 3 & 4 & 5 & 6 \\ 4 & 5 & 6 & 7 \end{bmatrix}$$

```
> MM1:=array(1..4,1..4,[seq([seq(
i+j,i=1..4)],j=1..4)]);
```

$$MM1 := \begin{bmatrix} 2 & 3 & 4 & 5 \\ 3 & 4 & 5 & 6 \\ 4 & 5 & 6 & 7 \\ 5 & 6 & 7 & 8 \end{bmatrix}$$

```
> MM[2,1];MM;
```

$$2 \quad MM$$

```
> MM[3,3]:=100;
```

$$MM_{3,3} := 100$$

```
> print(MM);
```

$$\begin{bmatrix} 1 & 2 & 3 & 4 \\ 2 & 3 & 4 & 5 \\ 3 & 4 & 100 & 6 \\ 4 & 5 & 6 & 7 \end{bmatrix}$$

```
MM2:=array(1..2,1..2,1..2,[[[a,b],
[c,d]],[[a1,b1],[c1,d1]]]);
```

**MM2 := ARRAY([1 .. 2, 1 .. 2, 1 .. 2], [(1, 1, 1) = a, (1, 1, 2) = b, (1, 2, 1) = c, (1, 2, 2) = d, (2, 1, 1) = a1, (2, 1, 2) = b1, (2, 2, 1) = c1, (2, 2, 2) = d1])**



```
> SM:=[seq([seq(m[i,j],j=1..3)], i=1..3)];
```

$$SM := [[m_{1,1}, m_{1,2}, m_{1,3}], [m_{2,1}, m_{2,2}, m_{2,3}], [m_{3,1}, m_{3,2}, m_{3,3}]]$$

```
>SM:=[seq([seq(m[i,j],j=1..3)],i=1..3)]:
> MMM:=array(SM);
```

$$MMM := \begin{bmatrix} m_{1,1} & m_{1,2} & m_{1,3} \\ m_{2,1} & m_{2,2} & m_{2,3} \\ m_{3,1} & m_{3,2} & m_{3,3} \end{bmatrix}$$

## 1.13Команди перетворення виразів

Головна превага системи полягає в наявності можливості здійснювати не тільки обчислення над числовими виразами та функціями, але й проводити символьні обчислення над виразами, які містять змінні величини та функції.

Команди перетворення виразів мають наступний синтаксис: **команда(пар_1,пар_2, ….пар_n);** або **команда(пар_1, пар_2, ….пар_n):.** У першому випадку,коли команда закінчується крапкою з комою,команда виконується, і після неї виводиться результат та виконання у другому випадку, коли команда закінчується двокрапкою, команда виконується, але результат команди не виводиться.

## 1.14Команда спрощення виразів simplify

Ця команда призначена для здійснення операції спрощення виразів, які включають в себе раціональні функції, тригонометричні функції, логарифмічні та експоненціальні функції.

Команда має декілька форм використання : **simplify (вираз);** У дужках задається вираз, який повинен бути спрощений. Для спрощення окремих типів виразів раціонально використовувати додаткові параметри команди **simplify (вираз, n1,n2,…).** Тут n1,n2,… є імена процедур спрощення виразів, а саме:

**exp** - для спрощення функцій з експоненціальними виразами;

**ln** - для спрощення виразів з логарифмами;

**sqrt** - для спрощення виразів з квадратними коренями;

**trig** - для спрощення тригонометричних виразів;

**radical** - для спрощення виразів з дробовими степенями;

**power**- для спрощення виразів з степенями, експонентами, логарифмами.

Крім того, в якості параметра процедури simplify можна використовувати параметр виду **assume= властивість**, де параметр**властивість** може приймати одне з наступних значень:

**complex** - комплексна область;

**real** - дійсна область;

**positive** - додатні дійсні числа;

**integer** - цілі числа;

**RealRange(a,b)** - інтервал (a,b) дійсних чисел.

Команда **спрощення** дозволяє задавати правила спрощення користувача, які користувач пропонує використати при виконанні спрощення відповідного виразу.

Ці правила задаються у команді **simplify()** другим параметром і мають наступний вигляд **{ рівність 1, рівність 2,…}.**

**Розглянемо приклади використання оператору спрощення виразів:**

**> simplify(((x-a)^2+x*(x-a)+x^2)/((xa)^2-x*(x-a)+x^2)-19/7);**

**> R1:=((x-a)*sqrt(x-a)+(x-b)*sqrt(xb))/(sqrt(x-a)+sqrt(x-b))+a+b;**

**> simplify(R1,sqrt,symbolic);**

**>R2:=log[a](sqrt(a^2-1))*(log[1/a](sqrt(a^2-1)))^2/(log[a^2](a^2-1)*log[a^(1/3)]((a^2-1)^(1/6))):**

**> simplify(R2);**

**>simplify(%,{ln(1/a)=-ln(a),ln(a^2)=2*ln(a)});**

**> r4:= (1-2*(cos(v))^2)/(sin(v)* cos(v))+ 2*tan(2*v)+cot(4*v)-3;**

**>simplify(r4,trig);**

## 1.15Команди розкриття дужок у виразах expand

Команда **expand()** виконує розкриття дужок і перетворює добутки у суми у будь – яких алгебраїчних виразах. Ця команда виконується для будь – яких поліномів та для частки двох поліномів. У випадку частки двох поліномів ця команда розкриває дужки у чисельнику і ділить кожний член отриманого виразу на знаменник, який вона не змінює. Крім того, команда expand() вміє розкривати дужки для більшості математичних функцій: sin(x), cos(x), tg(x), ln(x), exp(x), sh(x), ch(x) та інших.

**> expand(sin(x+y)*sin(y+z));**

**> expand((a*x^2+b*x^3)*(b*x^3+3*a*x*2));**

## 1.16Команда приведення декількох членів виразу до одного, команда combine

Команда **combine()** приводить декілька членів, представлених сумою, добутком, або степенями до одного члена , використовуючи різноманітні правила, які є фактично оберненими правилам перетворення команди expand().

**>d1:=sin(x)*cos(y)+cos(x)*sin(y);**

**> combine(d1);d1 := sin( x ) cos( y ) + cos( x ) sin( y )sin( x + y)**

**> assume(x>0);assume(y>0);**

**> d2:=2*ln(x)+3*ln(y);**

**> combine(d2);**

**> d1:=exp(2*x)*exp(y)/exp(z);**

**> combine(d1);d2 := 2 ln( x~) + 3 ln( y~)ln( x~ ) 2 y~3d1 :=e(2 x )eyeze(2 x + y − z)**

## 1.17Розвинення полінома на прості множники, функція factor()

Призначення функції **factor()** провести розвинення на множники поліному від декількох змінних.

Під поліномом у системі **MAPLE** розуміють вирази, що містять невідомі величини, у яких кожний член має вигляд добутку цілих невід'ємних степенів невідомих величин з числовими або алгебраїчними коефіцієнтами. Невідома величина у поліномі може бути представлена викликом деякої функції, аргумент якої є невідома величина. Команди розвинення поліному має вигляд **factor( exp,K),**де exp – вираз, що підлягає розвиненню на множники, а K – необов'язковий параметр, який уточнює над яким полем числових коефіцієнтів здійснюється розвинення на множники. Цей параметр може приймати значення **real complex**, мати значення одного радикалу, або списку радикалів.

**> factor(x^3+3,3^(1/3));**

**>factor(x^3+3,real);**

**> factor(x^3+3,complex);**

**> factor(x^3-y^3);**

**>factor(x^3-y^3,(-3)^ (1/2));**

**>factor(x^3-3,complex);**

**>x^2+x*3^(1/3)+3^(2/3);**

**> evalf(%);**

**> factor(%,complex);**

### 1.18Скорочення алгебраїчного дробу команда normal()

Призначення команди **normal()** полягає у приведенні виразів, що містять алгебраїчні дроби до спільного знаменника і спрощення отриманого алгебраїчного дробу шляхом скорочення чисельника і знаменника. Командnormal() має наступний вигляд **normal(f, expanded).** Де f – алгебраїчний дріб, анеобов'язковий параметр expanded вказує на те, що після скорочення дробу учисельнику і знаменнику розкриваються дужки.

**> gg1:= 1/(x+1)^2 +3/(x-1)^2+ 4/((x-1)*(x+1));**

**> normal(gg1); > normal(gg1,expanded);**

## 1.19Команда приведення подібних членів collect

Команда **collect()** працює з узагальненими поліномами декількох змінних – поліномами, в яких в якості невідомих можуть виступати функції з аргументами,що є невідомими величинами в **MAPLE**. Синтаксис цієї команди має три форми:

**-collect(exp,x) ; collect(exp,x,form,func)** ;

-collect(exp,x,func) exp - вираз, у якому необхідно провести комплектування виразу постепеням невідомих;

-x - ім’я невідомої величини, список, множина невідомих величин у випадку поліному декількох змінних, або ім’я функції з аргументом – невідомим.

-form - має два значення ,**recursive** або **distributed**,має зміст для многочленів декількох змінних і впливає на алгоритм приведення подібних членів відносно обраного списку змінних.

-func - параметр що визначає ім’я команди, яка використовується по відношенню до отриманих коефіцієнтів, зазвичай **simplify()** або **factor().**

| | |
|---|---|
| `> s:= 2*x^3*a^2*y^2+ 3*x^3*y^2*b+2*x^2*y^3*a^2+4*a *x^2*y^3;` | $s := 2x^3a^2y^2+3x^3y^2b+2x^2y^3a^2+4a$ |
| `> collect(s,[y,x]);` | $(2a^2+4a)x^2y^3+(2a^2+3b)y^2x^3$ |
| `> collect(s,[x,y], factor);` | $2a(a+2)x^2y^3+(2a^2+3b)y^2x^3$ |

| | |
|---|---|
| `>s1:=2*a*(sin(x))^2+3*b^ 2*(sin(x))^2*x^2+15*x^2* sin(x)+2*c*sin(x);` | $s1 := 2a\sin(x)^2+3b^2\sin(x)^2x^2+15x^2\sin(x)+2c$ |
| `> collect(s1,sin(x));` | $(2a+3b^2x^2)\sin(x)^2+(15x^2+2c)\sin(x)$ |

## 1.20Структура виразів та робота з ними

Дуже часто для виконання певних дій з виразами з’являється необхідність виділити з виразу його частину та провести з нею певні перетворення , або замінити її на деякий інший вираз. В таких випадках треба знати струк-

туру виразу і вміти з нею працювати шляхом звернення до будь - якої частини.

Будь – який вираз у системі **MAPLE** представляється у вигляді ієрархічної структури, в якій кожний об’єкт розділяється на підоб’єкти першого рівня, які у свою чергу діляться на підоб’єкти другого рівня і так далі. Процес виділення підоб’єктів продовжується до тих пір , поки не будуть виділені базові прості елементи **MAPLE**, такі як цілі числа, дійсні числа, дроби, змінні величини символьного типу, функціїБ списки, послідовності, множини, масиви та інші.

Для роботи з виразами корисним будуть функції:

-**lhs(egn)-** виділяє ліву частину рівності egn;

-**rhs(egn**)- виділяє праву частину рівності egn;

-**numer(exp)-** виділяє чисельник дробу exp;

-**denom(exp)-** виділяє знаменник дробу exp.

**>eq1:=(a*x^2+b*x+c)/tan(x)=x*sin(x); eq1 := =a x + + 2 b x ctan( x) x sin( x)**

**> rhs(eq1);**

**> lhs(eq1);**

**> numer(rhs(eq1));**

**> denom(rhs(eq1));**

Для більш детального вивчення структури виразу використовуються наступні функції **MAPLE**:

1.**whattype(exp**)повертає тип виразу exp;

2.**op(exp)** повертає список об’єктів першого рівня виразу exp;

3.**op(k..n,exp) (op(k,exp))**повертає список від к –го до n –го об’єкту (k–й об’єкт) першого рівня виразу exp, при к = 0 повертається тип виразу exp;

4.**nops(exp)** повертає число об’єктів першого рівня в виразі exp;

**> op(0..nops(eq1),eq1);**

**> eq11:=op(1,eq1);**

**> eq12:=op(2,eq1);**

**> op(0..nops(eq11),eq11);**

**> op(0..nops(eq12),eq12);**

**> eq121:=op(1,eq12);**

**> op(0..nops(eq121),eq121);**

**> op(1,eq121);**

**> op(2,eq12);**

**> op(op(2,eq12));**

**> op(1,op(2,eq12));**

## 1.21Обчислення виразів

У програмі символьних обчислень **MAPLE** користувач зустрічається з поняттям **обчисленнясимвольнихімен**. Подібна функція відсутня для систем численних обчислень. Здебільшого, система **MAPLE** використовує алгоритм повного обчислення символьної змінної. Тобто, у випадку, коли необхідно обчислити значення символьної змінної перевіряється, чи присвоювались цій змінній які - небудь значення, якщо да , то воно підставляється замість імені змінної і перевіряється, чи містять підставлені значення невідомі змінні. Якщо містять, то перевіряється, чи були присвоювання цим змінним певних значень, якщо були то вони підставляються замість значень змінних.

Цей процес продовжується рекурсивно, поки замість усіх імен не будуть підставлені присвоєні їм значення, або якщо деяким змінним нічого не присвоювалось, то такі змінні залишаться в виразі як невідомі величини. Для повного обчислення значення будь – якої змінної можна просто вказати її ім’я і в результаті система **MAPLE** виконає повне обчислення значення цієї змінної, або можна використати команда **eval**(x) де x ім’я відповідної змінної, яку необхідно обчислити.

Правило повного обчислення значення змінних при задаванні імені змінної діє для змінних усіх типів за винятком змінних, які містять масиви, матриці таблиці та процедури. Для цих змінних діє правило обчислення до значення останнього присвоєного символьного імені. Для виконання повного обчислення такої змінної необхідно явно ініціювати команду **eval**(Y).

У випадку, коли необхідно просто обчислити ім’я змінної не використовуючи алгоритм повного обчислення значення цієї змінної, можна використати оператор **evaln(x);** або той же самий результат зможемо отримати за допомогою запису імені відповідної змінної у одинарних кавичках ‘x’. Вказаний оператор можна використовувати для того, щоб анулювати усі попередні присвоювання здійснені по відношенню до цієї змінної, що дозволить використовувати її просто як невідому величини з значенням, що дорівнює імені змінної.

**> x:=a*c; c:=b^2+2; a:=(d+e)^3;**

**> x; > eval(x);**

**> y:=d+e;**

**> x*y;**

**> eval(x*y);**

**> x/y;**

**> evaln(x);**

**> evaln(y);**

**> x:=evaln(x);**

**> x;**

**> y:='y';**

**> y;**

На практиці, в математичних дослідженнях дуже часто виникає задача обчислення значення математичного виразу або функції при конкретному значені змінних величин, що входять до відповідного виразу. Для того, щоб зберегти сам вираз і в той же час обчислити його значення при вказаних значення параметрів можна скористатися оператором **eval(expr, [eq1,eq2,…eqn])**; де expr – математичний вираз, що містить змінні величини, Параметри eq1,eq2,…eqn – рівності, що задають значення змінним величинам, при яких повинен бути обчисленим відповідний вираз expr.

**> g:=2*x^3*y+3*x*y^3+5*x*y;**

**> gg:=eval(g,x=1);**

**> gg1:=eval(gg,y=1);**

**> eval(g,[x=1,y=1]);**

Система **MAPLE** прагне провести обчислення будь – якої величини точно і використовує арифметику дробових чисел та радикалів там де це можливо. Якщо при проведені обчислень результат можна отримати лише у виглядііраціонального числа, або більш доцільно отримувати числа у вигляді звичайнихдесяткових дробів з заданою кількістю знаків після коми, то для обчисленнязначення виразу можна використовувати оператор **evalf(f,n),**де f вираз щонеобхідно обчислити , а n – число значущих цифр, які використовуються приобчисленнях (за замовченням n=10).

Система **MAPLE** використовує ще цілий ряд функцій призначених для проведення обчислень виразі:

**evalc**– обчислення виразів, що приймають комплексні значення;

**evalb**– обчислення виразів, що приймають булєві значення;

**evalm**– обчислення матричних виразів;

**evalr**– обчислення інтервальних виразів.

Для аналітичного перетворення виразів і зокрема для їх спрощення в математиці, використовуються операція підстановок, де замість одних змінних та виразів здійснюється підстановка інших змінних та виразів. В системі **MAPLE**існують декілька команд, які виконують відповідні функції. До цих команд відносяться:

**subs(x = a, exp)** – у виразі exp заміняє x на a.

**subs(s1,s2…sn, exp)** - у виразі exp заміняє одні підвирази на інші, обираючи їх зі списку s1,s2…sn, кожний елемент списку має вигляд x=a.Підстановки виконуються послідовно.

**subsop(N1=a1,…,Nk =ak, exp)** – у виразі exp заміняє підвираз з номером N1 на підвираз a1…з номером Nk на підвираз ak, де номер підвиразу визначається за допомогою аналізу структури оператором op(n, exp). Усі підстановки виконуються послідовно.

**algsubs(x=a,exp)** - у виразі exp заміняє підвираз x на підвираз a.

## 1.22Обмеження на невідомі

Для проведення різноманітних математичних обчислень і перетворень дуже часто необхідно накладати на відповідні змінні певні обмеження, робити відносно невідомих певні припущення. У системі **MAPLE** існує декілька команд, які дозволяють вводити і перевіряти обмеження, накладені на змінні або цілі вирази.

Команда **assume (x, властивість)**накладає на змінну x відповідну властивість. Тут x представляє собою змінну, або вираз який включає таку змінну. Властивість може приймати такі найбільш широковживані значення.

## 1.23Властивість. Опис властивості

Від'ємні дійсні числа- **negative**;

Невід'ємні дійсні числа- **nonnegative**;

Додатні дійсні числа- **positive**;

Натуральні числа (0,1,2,3…)- **natural**;

Цілі, строго більше 0- **posint**;

Непарні числа- **odd**;

Парні числа- **even**;

Комплексні числа- **complex**;

Дійсні числа- **real**;

Раціональні числа- **rational**;

Цілі числа- **integer**.

Пару параметрів (x,властивість) можна замінити математичним відношенням, наприклад: assume(x>0); assume (x>=0).

Якщо на змінну накладені обмеження , то у результатах перетворень з цією змінною ця змінна виводиться за замовченням зі знаком (~).

Команда **assume** може отримувати декілька пар (x,властивість) або математичних відношень, наприклад:**assume(x>1, x<=2**).

Нове обмеження, що накладається новою командою **assume()** на змінну відміняє усі попередні обмеження.

Для послідовного додавання додаткових обмежень по ходу розв'язування задач можна використовувати команду **additionally ()** з параметрами, аналогічнимипараметрам команди assume, наприклад assume (x>=0), additionally(x<=100).

Таким чином, обмеження, накладені двома останніми операторами, накладають на змінну x обмеження: 0 <=x<=100.

За допомогою функції **is(),** яка вертає значення true або false в залежності від того,задовольняє змінна відповідній властивості або ні.

**>assume(x>5, y<-10);**

**> is(x*y < 50);**

**>assume(x>y,y>0); is(x<0);**

**>assume((2*i+2)/2,integer); is(i,integer);**

### 1.24Перетворення виразів в тотожні форми, команда convert()

Більшість математичних виразів мають різноманітні математичні форми запису. Здебільшого, перетворення виразу з однієї форми в іншу дозволяє отримати вираз більш зручний для подальшого використання. Крім того різні функції **Maple** працюють з різними формами виразів і різними типами даних.

Велике значення має цілеспрямоване перетворення виразів та даних.

Основною функцією для таких перетворень є функція **convert:**

**convert(expr, form, arg3. ...)**

Тут expr — довільний вираз, form — найменування форми, arg3, ... — необов'язкові додаткові аргументи.

**Convert** — досить проста і разом з тим потужна функція.. ЇЇ потужність полягає у можливості множину параметрів. Їх повний перелік нараховує (76 штук!) можна знайти в довідці по функції convert.

## РОЗДІЛ 2Розв`язування задач математичного аналізу.

Система аналітичних обчислень **MAPLE** має багатий набір вбудованих функцій, призначених для розв`язування класичних задач математичного аналізу,а саме:

-обчислення границь послідовностей;

-обчислення границь функцій однієї змінної та декількох змінних;

-обчислення похідних та диференціалів функцій однієї змінної та багатьох змінних;

-розвинення функцій за формулою Тейлора та за формулою Лорана;

-знаходження невизначених інтегралів;

-обчислення визначених інтегралів;

-обчислення сум та добутків;

-дослідження збіжності рядів;

- виконання заміни змінних в диференціальних та інтегральних виразах;

-обчислення багатовимірних інтегралів;

-обчислення криволінійних інтегралів;

-обчислення поверхневих інтегралів та інші

## 2.1Обчислення границь послідовностей та функцій

Для обчислення границь функцій використовується стандартна функція **limit** або **Limit**, друга функція представляє собою **пасивну** форму функції.

Загальна форма функції limit або Limit має вигляд:

**limit (f(x),x=a,dir), Limit(f(x),x=a), Limit (f(x),x=a,dir),**

де f(x) – алгебраїчний вираз, що залежить від x, x – ім`я змінної, a – точка в якій обчислюється значення границі, dir – параметр, який вказує на напрямок пошуку границі він приймає значення (left – пошук односторонньої границі зліва, right – пошук односторонньої границі справа, real – пошук границі в дійсній області, complex – пошук границі в комплексній області.

Обчислення границі послідовностей здійснюється за допомогою функції **limit(f(n),n=infinity),**для змінної n можна використати функцію **assume(n, integer).**

**> assume(n,integer);**

**> limit((1/nsin(1/n))*(n^3),n=infinity);**

**> limit((1+1/n)^(7*n),n=infinity);**

**> Limit((1+1/n)^(7*n),n=infinity);**

**> limit(tan(x),x=Pi/2,left);**

**> Limit(tan(x),x=Pi/2,right);**

**> value(%);**

**> limit(((5-x)/(6-x))^(x+2),x=infinity).**

## 2.2Обчислення похідних функцій

Для диференціювання функцій в системі **MAPLE** використовуються стандартні функції**diff(f,x1,x2…xn), Diff(f,x1,x2…xn)**, де f – алгебраїчний вираз, який диференціюється , зокрема функція змінних x1,x2,….xn. Друга функція**Diff**представляє собою пасивну форму функції.

Слід зауважити, що при використанні функції diff або Diff результатом обчислення є алгебраїчний вираз, який безпосередньо не може бути обчислений для фіксованих значень аргументів. Для обчислення похідних старших порядків при використанні функцій **diff(f,x1,x2…xn), Diff(f,x1,x2…xn)** можна використовувати однакові значення аргументів, або використовувати конструкцію **x$k**, де x – змінна по якій проводиться диференціювання, k – порядок похідної по змінній x, $ - спеціальний символ повтору елементу x k разів.

Другий спосіб обчислення похідних функцій пов`язаний з використанням диференціального оператора D, який має наступний вигляд **D(f)** або **D[i](f),** де f – вираз або функція, i - додатне ціле число, вираз, який має ціле значення, або послідовність цілих чисел, яка вказує номери аргументів по яким здійснюється диференціювання функції. На відміну від функції diff(f,x1,x2…xn), операторD[i](f) після обчислення результату диференціювання зразу продукує функцію відповідної кількості аргументів, що дозволяє

провести безпосереднє обчислення відповідної похідної для будь – якого значення аргументів.

**>Diff(sin(x)/x,x)=diff(sin(x)/x,x);**

**> diff(sin(x)/x,x$3);**

**> diff(sin(x)/x,x,x,x);**

**> g:=sin(x)/x+tan(x);**

**> diff(g,x);**

**>diff(x*y/sqrt(1+x^2+y^2),x,x)+diff(x*y/sqrt(1+x^2+y^2),y,y);**

**>f:=(x,y,z)->sin(x*y*z);**

**>g1:=D[1,2,2](f);**

**> g1(1,1,Pi);**

### 2.3Обчислення похідних від неявно заданих функцій

Дуже часто виникає необхідність обчислювати похідні від неявно заданих функцій, тобто таких функцій, які задаються у вигляді одного або декількох рівнянь, які містятять як незалежні змінні, так і саму функцію. При заданні неявної функції слід обов`язково визначити які змінні є незалежними, а які є функціями. Для обчислення похідних від неявно заданих функцій використовується функція **implicitdiff( ).**Яка може мати один з можливих форматиів:

**implicitdiff(f, y, x)**

**implicitdiff(f, y, x1,...,xk)**

**implicitdiff({f1,...,fm}, {y1,...,yn}, u, x)**

**implicitdiff({f1,...,fm}, {y1,...,yn}, u, x1,...,xk)**

**implicitdiff({f1,...,fm}, {y1,...,yn}, {u1,...,ur}, x)**

**implicitdiff({f1,...,fm}, {y1,...,yn}, {u1,...,ur}, x1,...,xk)**

Де f, f1,...,fm – алгебраїчні рівняння (одне рівняння), що задають неявнуфункцію, або систему неявних функцій; y, y1,...,yn - імена залежної змінної, або змінних; u,u1,...,ur - імена залежних змінних, похідні від яких будуть знайдені; x,x1,...,xk . – імена змінної, або змінних по яким відбуватиметьсяобчис-

лення похідної, порядок частинної похідної визначається кількістю таіменами змінних.

Наступний приклад демонструє обчислення першої та другої похідної від неявно заданої функції однієї змінної.

**> f:=sin(x+y)- x*y=0; f := sin(x + y) − x y = 0**

**>implicitdiff (f,y,x);**

**>implicitdiff (f,y,x,x).**

Наступний приклад демонструє обчислення частинних похідних першого та другого порядку від неявної функції двох змінних.

**>g:=z^3+3*x^2*z=2*x*y;**

**> implicitdiff(g,z,x);**

**>implicitdiff(g,z,y);**

**>implicitdiff(g,z,y,y);**

**>implicitdiff(g,z,x,x);**

**>implicitdiff(g,z,x,y).**

Наступний приклад демонструє знаходження частинних похідних першого та другого порядку від двох неявних функцій двох незалежних змінних u=u(x,y),v=v(x,y), які задані системою рівностей:

**> fu:=x*u-y*v=0;**

**> fv:=y*u+x*v=1.**

Далі обчислюються частинні похідні першого порядку від функцій u=u(x,y), v=v(x,y).

**>implicitdiff({fu, fv},{u,v},{u,v},x,notation=Diff);**

**>implicitdiff({fu, fv}, {u,v},{u,v},y,notation= Diff);**

В наступному прикладі провадиться обчислення другої змішаної похідної від функцій u=u(x,y), v=v(x,y).

**>implicitdiff({fu,fv},{u,v} ,{u},x,y,notation=Diff);**

**>implicitdiff({fu,fv},{u,v} ,{v},x,y,notation=Diff);**

## 2.4Заміна змінних у диференціальних виразах

В системі **MAPLE** існує можливість проводити заміну змінних в диференціальних виразах, як однієї незалежної змінної, так і декількох незалежних змінних. Для цього можна використовувати функцію **dchange()** пакету **PDEtools.**Яка має наступний вигляд:**dchange(tr, expr),** де tr –множина алгебраїчнихрівностей, які містять в лівій частині старі незалежні змінні, а в правій частині вирази від нових незалежних змінних.

В наступному прикладі для звичайного диференціального рівняння третього порядку введена заміна змінних z=et:

```
>with(PDEtools):
>assume(z>0);
>g:=diff(y(z),z$3)=6*y/ z^2;
>simplify(dchange(z=exp (t),g)*exp(3*t));
```

В наступному прикладі для лінійного диференціального рівняння першого порядку з двома незалежними змінними введена заміна змінних $\xi = x + y$, $\eta = x - y$:

```
>pr:={xi=x+y,eta=x-y};
>pr1:=solve(pr,{x,y});
>dd:=diff(z(x,y),x)= diff(z(x,y),y);
>dchange(pr1,dd);
> -(rhs(%)-lhs(%)=0)/2;
```

Вказані функції виконують заміну лише незалежних змінних в диференціальних виразах. В той же час іноді виникає необхідність провести заміну змінних в більш складному випадку.

## 2.5Обчислення сум послідовностей та сумування числових рядів

Для обчислення сум послідовностей в системі **MAPLE** існує стандартні функції **sum** та її інертна форма функція **Sum**. Функції мають наступну форму:

**sum(f,k) sum(f,k=m..n), sum(f,k=alpfa), Sum(f,k) Sum(f,k=m..n), Sum(f,k=alpfa)**

Тут f – функція, що задає члені послідовності, яка залежить від k, k – індекс-сумування, m та n – цілочисельні границі зміни індексу сумування., alpfa – виразтипу RootOf(z). В цьму випадку індекс k приймає значення з множини кореніврівняння z=0. Перша форма суми sum(f,k) еквівалентна формі sum(f,k=1..k-1).

**>sum('k'^3,'k');**

**>sum('k'^3,'k'=1..'k-1');**

**>Sum(-(-1)^'k'/'k','k'=1..infinity)=sum(-(-1)^'k'/'k','k'=1..infinity);**

**> Sum(1/('n'*('n'+1)),'n'=1..infinity)=sum(1/('n'*('n'+1)),'n'=1..infinity);**

**>Sum(sin('k'*x)/(2^'k'),'k'=1..infinity)=simplify(sum(sin('k'*x)/(2^'k'),'k'=1..infinity));**

**>Sum(1/'k','k'='m'..'n')=sum(1/'k','k'='m'..'n').**

Слід зауважити, що при обчисленні сум послідовностей необхідно строго додержуватися прямого (зростаючого) порядку задання значення індексної змінної суми.

## 2.6Обчислення добутків членів послідовностей

Для обчислення добутків членів послідовності в системі **MAPLE**використовуються стандартні функції:

**Product(f,k) product(f,k=m..n), product(f,k=alpfa); Product(f,k) Product(f,k=m..n), Product(f,k=alpfa)**

Параметри функцій product мають зміст аналогічний параметрам функції sum.

**>Product(('n'^2-4)/('n'^2-1), n=3..infinity)=product(('n'^2-4)/('n'^2- 1), n=3..infinity);Product((1-x^2/('n'^2*Pi^2)),'n'=1..infinity)= product((1-x^2/('n'^2*Pi^2)),'n'=1..infinity);**

**>Product(1-1/'n'^2,'n'=2..'k')=product(1-1/'n'^2,'n'=2..'k');**

**>Product(1-1/'n'^2,'n'=2..infinity)=product(1-1/'n'^2,'n'=2..infinity).**

Слід зауважити, що при обчисленні добутків послідовностей необхідно строго додержуватися прямого (зростаючого) порядку задання значення індексної змінної суми.

## 2.7Розвинення функцій в ряди

Для розвинення аналітичних функцій в ряд Тейлора або ряд Маклорена в заданій точці в системі **MAPLE** використовується стандартні функції **taylor(),mtaylor().** Перша функція використовується для функцій однієї змінної , а другадля функцій багатьох змінних.

Функція **taylor()** має формат **taylor(exp, x=a, n),** де exp – вираз, що залежить від x, або функція, x=a, задає точку для якої здійснюється розвинення в ряд, n –необов`язковий параметр, що задає кількість членів розвинення, при його відсутності за замовчанням кількість членів розвинення дорівнює 6.

Функція **mtaylor()** має вигляд **mtaylor(f,v,n),** де f – вираз або функція, що залежить від декількох змінних. v – список рівностей виду [x=a,y=b,…z=c], обо просто список змінних [x,y,…z], n – необов`язковий параметр, який вказує на максимальну сумарну ступінь в розвиненні, на якій обривається розвинення.

**>taylor(ln(cos(x)),x,8);**

**>taylor((sin(x^3))^(1/3),x,20);**

**> taylor(x^x-1,x=1);**

**>taylor(sqrt(1+x^2)-x,x=infinity).**

## 2.8Обчислення невизначених інтегралів

Невизначені інтеграли в **MAPLE** обчислюються за допомогою стандартноїпроцедури **int(exp,x),** яка має також інертну форму **Int(exp,x),** де exp – підінтегральна функція, x – змінна інтегрування. Інертна форма процедуриінтегрування використовується для символьного запису невизначеного інтегралу.

Розглянемо приклад застосування процедур знаходження невизначеного інтегралу.

**>Int(1/(x^2+1)^(3/2),x)=int(1/(x^2+1)^(3/2),x);**

**> f:=x->(x^2+5*x+4)/(x^4+5*x^2+4);**

**> Int(f(x),x)=int(f(x),x)**

## 2.9Обчислення визначених інтегралів

Для обчислення визначених інтегралів в системі **MAPLE** використовується процедура int() та її пасивна форма Int() з такими параметрами виклику**. int(exp, x=a..b**), де exp підінтегральна функція, x – змінна інтегрування, a,b – відповідно верхня та нижня межа інтегрування.

Розглянемо приклад обчислення визначених інтегралів.

**>Int(1/(sin(x)^4+cos(x)^4),x=0..2*Pi)=int(1/(sin(x)^4+cos(x)^4),x=0..2*Pi);**

**>assume(a>=0);**

**> Int(x^2*sqrt(a^2-x^2),x=0..a)=int(x^2*sqrt(a^2-x^2),x=0..a);**

**> assume(b<1 and b>0);**

**> Int(x*abs(x-b),x=0..1)=int(x*abs(x-b),x=0..1).**

Дуже часто виникає потреба в обчисленні невласних інтегралів першого або другого роду. Нагадаємо, що невласними інтегралами першого роду називають визначені інтеграли, в яких хоча б одна межа інтегрування є нескінченою.

Невласними інтегралами другого роду називається **визначені інтеграли**, в яких підінтегральна функція принаймні в одній точці інтервалу інтегрування перетворюється в нескінченість.

Розглянемо приклад обчислення невласного інтегралу першого роду.

**>Int(1/(x^2+x+1)^2,x=-infinity..infinity)=int(1/(x^2+x+1)^2,x=-infinity..infinity);**

**>int(ln(1+x)/x^n,x=0..infinity);**

**>Int(x^2*cos(exp(x)),x=0..infinity)=int(x^2*cos(exp(x)),x=0..infinity);**

Розглянемо приклади обчислення невласних інтегралів другого роду.

**> int(ln(sin(x)),x=0..Pi/2);**

У цьому прикладі підінтегральна функція є необмеженою в точці x = 0:

**> int(1/(1-cos(2*x)),x=0..Pi/4).**

Цей приклад демонструє, що інтеграл є розбіжним за рахунок сильного зростання підінтегральної функції в точці x = π / 4:

**>Int(1/(3-x)^(1/2),x=1..3)=int(1/(3-x)^(1/2),x=1..3).**

Підінтегральна функція необмежено зростає в точці x = 3, проте інтеграл приймає скінчене значення.

## 2.10Головне значення Коші для невласних інтегралів

Іноді невласний інтеграл від необмеженої функції не існує, але він може існувати в дещо послабленому розумінні, а саме може існувати його головне значення за Коші. Це головне значення можна позначити та записати у вигляді.

**∫ ∫ ∫+−→+= +bccabav p f x dx f x dx f x dxεεε. . ( ) lim [ ( ) ( ) ]0.**

Для невласних інтегралів на необмеженому інтервалі теж можна розглядати певне послаблення невласного інтегралу, а саме його головне значення за Коші,яке можна позначити та записати у вигляді:

**∫ ∫− →+∞∞−∞=aaav.p. f (x)dx lim f (x)dx.**

Для обчислення головного значення за Коші невласного інтеграла першого або другого роду використовується функція

**int(exp, x=a..b ,'CauchyPrincipalValue'),**

де параметр 'CauchyPrincipalValue'' вказує на необхідність розглядати відповідний невласний інтеграл за його головним змачення за Коші.

**>int(cos(x)/x^3, x=-Pi/2..Pi/2,'CauchyPrincipalValue');**

**>int(cos(x)/x^3, x=-Pi/2..Pi/2);**

Попередній приклад демонструє, що головне значення інтегралу за Коші, дорівнює нулю, але в звичайному розумінні невласний інтеграл не існує.

**> int(1/(x^2-3*x+2),x=0..infinity,'CauchyPrincipalValue').**

Цей приклад демонструє наявність головного значення за Коші невласного інтегралу другого та першого роду одночасно. Як інтеграл другого роду підінтегральна функція має необмежене зростання в точках x = −1, x = −2. В зви-

чайному розумінні невласний інтеграл не існує, про що свідчить наступне обчислення в системі **MAPLE**.

**>int(1/(x^2-3*x+2),x=0..infinity).**

## 2.11Обчислення подвійних інтегралів

Для обчислення подвійних інтегралів в системі **MAPLE** не існує спеціальної окремої процедури, тому обчислення усіх подвійних інтегралів у випадку, коли область інтегрування є простою, тобто обмежена контуром першого або другогороду, згідно до загальної теорії зводиться до обчислення повторних інтегралів зі змінними верхніми границями. У випадку, коли область інтегрування є більшскладною, то вона представляється у вигляді декількох простих областей.

Розглянемо приклад обчислення подвійних інтегралів шляхом їх зведення до повторних.

Приклад: Обчислити подвійних інтеграл ∫∫ +E(x y )dxdy 2 2 по множині E,обмеженій кривими: y = 0, x = y, x =1.

Неважко зрозуміти, що такий інтеграл зводиться до повторного інтегралу одним з двох способів:

**∫ ∫ ∫ ∫ + = +1020210202 ( (x y )dy)dx ( (x y )dx)dyx y.**

**Int(Int((x^2+y^2),y=0..x),x=0..1)=int(int(x^2+y^2 ,y=0..x),x=0..1);**

**>Int(Int(x^2+y^2 ,x=0..y),y=0..1)=int(int(x^2+y^2 ,x=0..y),y=0..1);**

Хоча в системі **MAPLE** немає спеціалізованої функції обчисленняподвійних інтегралів, але в пакеті student існує функція Doudleint, яка має лишенейтральну (пасивну) форму і призначена для запису подвійного інтегралу.

**>with(student):**

**>D1:=Doubleint(y^2*sqrt(R^2-x^2),=-sqrt(R^2-x^2)..sqrt(R^2-x^2),x=-R..R);**

Слід зауважити, що порядок слідування змінних, в межах інтегрування, а саме y, а потім x є дуже важливим для коректного обчислення цього інтегра-

лушляхом зведення до повторного за допомогою функції value(exp), де в якостізмінної exp беремо подвійний інтеграл D1.

**> value(D1);**

Цей приклад демонструє можливості функції **Doudleint** лише для символічного запису подвійного інтегралу по деякій області Ω.

Другим досить ефективнім методом обчислення подвійних інтегралів є метод заміни змінних інтегрування, який можна виконати вже відомою намфункцією changevar.

## 2.12Обчислення потрійних інтегралів

Для обчислення потрійних інтегралі в системі **MAPLE**,так само як і для подвійних інтегралів, використовується метод зведення до повторних інтегралів зі змінними межами інтегрування. Крім того у пакеті student існує нейтральна функція **Triplein**, яка використовується для символічного запису трьохвимірного інтегралу, або в сукупності з функцією **value** для його обчислення шляхом зведення до повторного.

Розглянемо приклад обчислення потрійних інтегралів

Приклад: Обчислити потрійний інтеграл dxdydz

**x y z V∫∫∫ + + 3 ( )1 ,**

де V-область, обмежена площинами x = 0, y = 0, z = 0, x + y + z =1

Обчислення цього інтегралу зводиться до трьохкратного повторного інтегрування

**∫ ∫ ∫– – –– – –x x yx y dz dx dy1031010 (1 )** , що і демонструє наступні обчислення.

**>Tripleint(1/(1+x+y+z)^3,z=0..1-x-y,y=0..1-**

**x,z=0..1)=int(int(simplify(int(1/(1+x+y+z)^3,z=0..1-x-y)),y=0..1-x),x=0..1);**

## 2.13Обчислення криволінійних інтегралів

Обчислення криволінійних інтегралів в системі **MAPLE** здійснюєтьсяшляхом зведення їх до звичайних визначених інтегралів. Спосіб зведення залежить від того, чи є криволінійний інтеграл інтегралом першого, чи інтеграломдругого роду, а також від способу завдання контуру по якому здійснюєтьсяінтегрування.

Таке зведення можна зробити безпосередньо, використовуючи відповідну формулу математичного аналізу, або застосувавши функцію Lineint() пакету student. Ця функція має декілька форматів використання в залежності від способу завдання контуру інтегрування, а саме:

**Lineint(f(x,y), x, y), Lineint(f(x,y), x=x(t),**

**y=y(t)), Lineint(f(x,y), x, y =a..b), Lineint(f(x,y), x=x(t), y=y(t), t=a..b);Lineint(f(x,y,z),**

(x, y, z). f(x,y)- підінтегральна функція, x, y – аргументи функції, a,b –нижня таверхня межі інтегрування.

Приклад: Обчислити $\int$ **+C(x y )ds 2 2 ,**

де C – крива.

**x = a(cost + tsin t) , y = a(sin t − t cost), 0 ≤ t ≤ 2π**

Наведений інтеграл представляє собою криволінійний інтеграл першого роду при параметричному способі завдання контуру інтегрування.

**>assume(a>0); >x:=t->a*(cos(t)+t*sin(t));>y:=t->a*(sin(t)-t*cos(t);>Lineint(x(t)^2+y(t)^2,x,y,t=0..2*Pi);>factor(value(%));**

Обчислення поверхневих інтегралів першого та другого роду в системі **MAPLE** зводиться фактично до подвійного інтегралу згідно відповідних формул математичного аналізу. В свою чергу подвійні інтеграли зводяться до повторних інтегралів.

## РОЗДІЛ 3Розв`язування рівнянь та нерівностей.

### 3.1Основна функція solve

Для знаходження розв`язку системи алгебраїчних рівнянь можна використовувати стандартну потужну функцію **MAPLE,solve**, яка може бути застосована не тільки до лінійних систем рівнянь, а і до систем нелінійних рівнянь та окремих рівнянь. Функція solve, має наступний формат виклику:

**solve(eqn,var) або solve({eqn1,eqn2,…},{var1,var2…}),**

де eqn – рівняння, якемістить змінні, var, ({eqn1,eqn2,… }- система рівнянь,{var1,var2…}- список змінних, відносно яких ця система буде розв`язана.

Характер розв`язків можна змінювати за допомогою глобальних системних змінних:

- **EnvExplicit** - при значенні true видає розв`язок без використання конструкції RootOf
- **EnvAllSolutions** - при значенні true видає усі розв`язки;
- **SolutionsMayBeLost** – при значенні true видає розв`язок, який при звичайному застосування функції solve повертає значення NULL.

## 3.2Системи лінійних алгебраїчних рівнянь

Для розв`язування систем лінійних алгебраїчних рівнянь існують потужні матричні методи з своїми спеціальними функціями. В той же час для розв`язування системи лінійних алгебраїчних рівнянь невисокого порядку можна використовувати і функцію solve.

Для розв`язування задач лінійної алгебри взагалі, та систем лінійнихалгебраїчних рівнянь зокрема в системі **MAPLE** існує спеціальний пакет **LinearAlgebra**, в якому серед набору функцій є функція:

**LinearSolve(A,b,method=’name_method’);**

де A- матриця системи рівнянь, b- вектор стовпець, ’name_method’- один зметодів знаходження розв`язування систем рівнянь:

**Метод LU – декомпозиції (mehtod=’LU’);**

**Метод QR –декомпозиції (mehtod=’QR’);**

**Метод декомпозиції Холеского (mehtod=’Cholesky’);**

**Метод оберненої підстановки (mehtod=’subs’).**

Функція LinearSolve здатна обчислювати розв`язки неповних систем рівнянь, у яких кількість рівнянь менша за кількість невідомих. При цьому розв`язок системи рівнянь виражається через вільні системні змінні, кількість яких залежить від рангу матриці.

## 3.3Знаходження коренів многочленів

При використання процедури solve для знаходження коренів многочленів функція знаходить усі корені многочлена як дійсні, так і комплексні.

**>P1:=x->x^4+2*x^3+2*x^2+x+5=0; P1 := x → x + + + + = 4 2 x3 2 x2 x 5 0**

**> solve(P1(x),x);84− − , , , 12−1 + 2 I 192 − +12−1 + 2 I 192 − −12−1 − 2 I 192 − +12−1 − 2 I 192**

У випадку, коли система **MAPLE** не в змозі провести обчислення коренів многочлена в радікалах, або коли таке представлення не можливе, **MAPLE**використовує для запису розв`язків функцію **RootOf**.

Наступний приклад демонструє використання функцій solve, RootOf та evalf для наближеного обчислення коренів многочлена

**>P2:=x->x^5+3*x^3+32=0;P2 := x → x + + =5 3 x3 32 0**

**>solve(P2(x),x);**

**RootOf(_Z + + , ) 5 3 _Z3 32 index = 1 RootOf(_Z + + , ) 5 3 _Z3 , 32 index = 2 ,**

**RootOf(_Z + + , ) 5 3 _Z3 32 index = 3 RootOf(_Z + + , ) 5 3 _Z3 , 32 index = 4 ,**

**RootOf(_Z + + , ) 5 3 _Z3 32 index = 5**

**>evalf(%);1.362408512 + 1.307370837 I, -0.4907275853 + 2.215266865 I, -1.743361853,-0.4907275853 − 2.215266865 I, 1.362408512 − 1.307370837 I**

## 3.4Розв`язування систем алгебраїчних рівнянь

Функція **solve** може використовуватись для розв`язування не тільки окремих нелінійних рівнянь, а і систем цих рівнянь. Розглянемо наступні системи рівнянь та способи їх розв`язку в системі **MAPLE.**

**> exp3:=y^2-x=5; exp3 := y − = 2 x 5**

**> exp4:=1/(y-1)-1**/(y+1)=1/x; exp4 := − = 1y − 11y + 11x

> solve({exp3,exp4},{x,y}); {x = 4, y = 3}, {x = 4, y = -3}

Для перевірки правильності розв`язку можливо виконати його перевірку шляхом підстановки за допомогою функції eval.

**> eval({exp3,exp4},S1[1]); {5 = 5, = } 1414**

**> eval({exp3,exp4},S1[2]); {5 = 5, = } 1414**

### 3.5Розв`язування тригонометричних рівнянь

Розв`язування тригонометричних рівнянь здійснюється за допомогою звичайної процедури solve з урахуванням тієї особливості, що як правилотригонометричні рівняння мають злічену кількість розв`язків, а процедура solve видає лише один корінь рівняння, який лежить на проміжку [0, 2π ]. Длязнаходження усіх розв`язків тригонометричного рівняння необхідно задаватизначення глобальної змінної **_EnvAllSolutions := true.**

**>solve(sin(x)=sqrt(3)/2,x);π3>_EnvAllSolutions:=true: >solve( sin(x)=sqrt(3)/2, x ); + +13 π13 π _B1~ 2 π _Z1~**

### 3.6Розв`язування рекурентних рівнянь

Для розв`язання рекурентних співвідношень в системі **MAPLE** існує спеціальна функція **rsolve**, яка дозволяє обчислити як загальний так і частинний-розв`язок рекурентного співвідношення. Параметрами функції rsolve є множина,яка містить одне рекурентне рівняння, або систему рівнянь, а також початковідані.

### 3.7Розв`язування нерівностей та їх систем

Для розв`язування нерівностей та їх систем використовується відома процедура **solve(ineq,exp)**,в якій параметр ineq – використовується для запису однієї нерівності, або системи нерівностей, а другий параметр exp – використовується для запису однієї або декількох змінних відносно яких розв`язується відповідна система нерівностей.

Розглянемо приклади розв`язування нерівностей.

**> e:=(x^2-5*x+6)/(x-4)<0; e := <x − + 2 5 x 6 x − 4 0 > solve(e,x); RealRange(−∞, Open(2)), RealRange(Open(3), Open**

Розглянемо приклад розв`язування системи нерівностей відносно однієїзмінної

**>ee2:={x^2-5*x+6>=0,abs(x- 4)>2}; ee2 := {2 < x − 4 , 0 ≤ x − + } 2 5 x 6**

**> solve(ee2,x); {6 < x}, {x < 2 }**

Розв`язок системи нерівностей дається у вигляді послідовності множин, яку треба розглядати як об`єднання множин x∈(−∞,2) ∪ (6,∞).

## РОЗДІЛ 4Візуалізація обчислень.

### 4.1Побудова двовимірних та тривимірних поверхонь

Система **MAPLE** має у своєму складі функції для побудови графічних об`єктів. Це, в першу чергу, функція для побудови двовимірних графіків **plot**() та функція для побудови тривимірних графіків **plot3d**. Для візуалізації більш складних математичних об`єктів в системі **MAPLE** існують спеціальні математичні пакети, які містять додаткові графічні функції.

### 4.2Побудова двохвимірних графіків.Основна функція побудови двовимірних графіків plot

Для побудови двовимірних графіків в системі **MAPLE** використовується функція **plot(f, h, v)plot(f, h, v, o),** де f, - функція, що візуалізується (або функції), h – змінна з завданням області її зміни ,v – необов`язкова змінна з за-

вданням області її зміни, o – параметр або набір параметрів, що задає стиль побудови графіків (товщину та колір кривих, тип кривих, мітки та інше).

### 4.3Побудова графіку однієї функції

При побудові графіку однієї функції за допомогою функції plot().Вона задається в явному вигляді на місті шаблону f, задається також область зміни незалежної змінної.

Для управління діапазоном завдання незалежної та залежної змінних,застосовуються параметри h та v, які задається у вигляді **var= expr1..expr2**, де var –ім`я змінної, а expr1 та expr2 верхня та нижня межі їх зміни. У розглянутому прикладі завдання діапазону має вигляд x=0..10,y=-1..1.

При побудові графіку функції на нескінченому проміжку в діапазоні змінної може з`явитися константа **infinity** та-**infinitю**.

### 4.4Графіки функцій з розривами

При побудові графіків функцій з розривами другого роду, коли функція може прямувати до нескінченості при підході до точки розриву, система MAPLE не завжди може чітко здійснити побудову графіку. Для урахування особливостей побудови графіків таких функцій використовується параметр discont=true.

### 4.5Управління стилем і кольором ліній двовимірних графіків

Графік в системі **MAPLE** можна задавати за допомогою лінії або точок , для цього використовується параметр **style**, який може приймати значення **point** або **line**, що відповідає стилю виведення графіку у вигляді суцільної лінії або окремих точок. При виведені графіку у вигляді окремих точок для завдання типу точки використовується параметр **symbol** , який може приймати значення :

**CROSS** - хрест;

**POINT** - крапка;

**DIAMONT** - ромб;

**CIRCLE** - коло;

**BOX** - квадрат;

Для завдання кольору ліній або символів використовується параметр color, який може приймати значення :

**red**– червоний;

**green**– зелений;

**blue**– блакитний;

**black**– чорний;

**white** - білий ;

**khaki**– хакі;

**gold**– золотистий;

**orange**– оранжевий;

**violet** - фіолетовий ;

**yellov**– жовтий.

Для визначення розміру символів відображення графіку використовується параметр **symbolsize**, цей параметр не впливає на розмір символів, якщо значення параметру **symbol = POINT**.

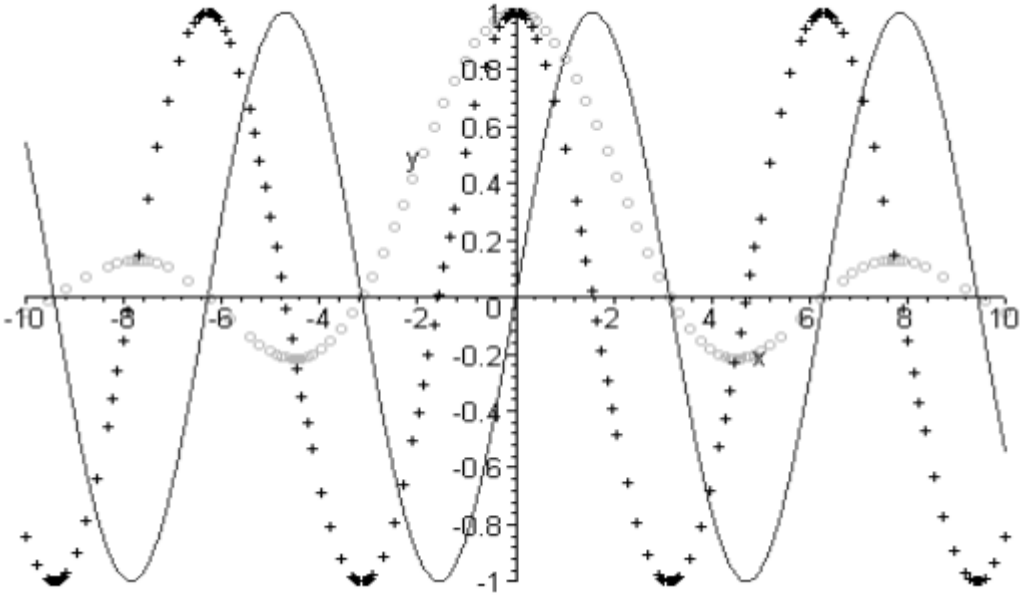
### 4.6Побудова графіків декількох графіків на одному малюнку

При побудові графіків декількох функцій на одному малюнку необхідно першим параметром функції plot задати список функцій та встановити для них спільні інтервали зміни залежної та незалежної змінних. Для кращого розпізнавання окремих графіків при їх зображенні можна використати різні стилі символів та різні кольори. Розглянемо приклад побудови трьох графіків на одному малюнку, для зображення яких використаємо стилі лінії для першого та точки для другого та третього графіків. Для зображення точками другого

та третього графіків використаємо символи хрест та коло. Кожний графік відобразимо окремим кольором: червоним, чорним, зеленим відповідно.

**>plot([sin(x),cos(x),sin(x)/x],x=-10..10,y=-1..1,color=**

**[red,black,green],style=[line,point,point],symbol=[CROSS,CIRCLE]);**




### 4.7Побудова графіка функції заданої сукупністю точок

Досить часто виникає потреба побудова графіка функції, що задана в табличній формі, тобто задана у вигляді сукупності окремих точок Така сукупність точок може бути задана списком координат незалежної змінної та значень функції в цих точках.

При побудові графіка відповідної функції список значень аргументу та значень функції передається в якості першого параметру функції plot.

Розглянемо приклад побудови деякої функції заданої набором своїх точок.

Сформуємо відповідний список значень для функції x ln x на проміжку від 1 до 2 з кроком 0.05.

**>FF:=[seq([1+i*0.05,(1+i*0.05)*ln(1+i*0.05)],i=0..20)];**

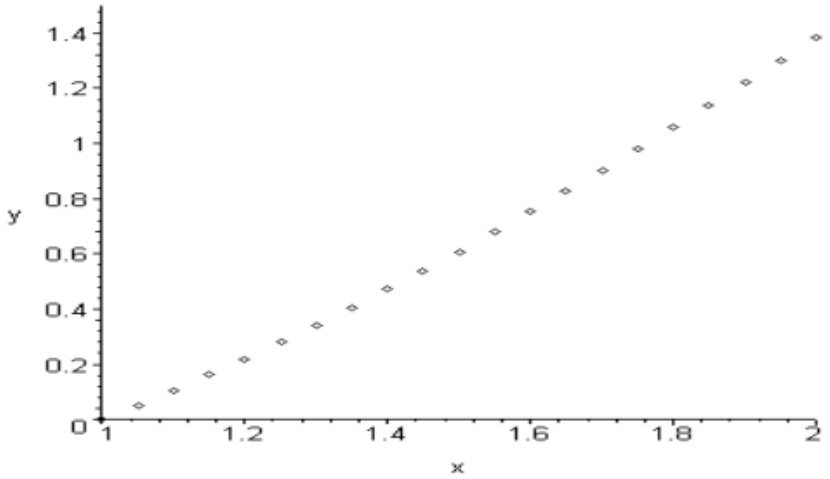
**FF := [[1., 0.], [1.05, 0.05122967238 ], [1.10, 0.1048411978 ], [1.15, 0.1607262338 ],**

**[1.20, 0.2187858682 ], [1.25, 0.2789294391 ], [1.30, 0.3410735438 ],**

**[1.35, 0.4051411999 ], [1.40, 0.4710611312 ], [1.45, 0.5387671568 ],**
**[1.50, 0.6081976622 ], [1.55, 0.6792951429 ], [1.60, 0.7520058067 ],**
**[1.65, 0.8262792250 ], [1.70, 0.9020680269 ], [1.75, 0.9793276288 ],**
**[1.80, 1.058015997 ], [1.85, 1.138093432 ], [1.90, 1.219522384 ],**
**[1.95, 1.302267277 ], [2.00, 1.386294361 ]]**

Виведемо графік функції використовуючи стиль виведення – точка і блакитний колір зображення.

**> plot(FF,x=1..2,y=0..1.5,style=point,color=blue);**




## 4.8Побудова полігонів

Функція **plot** можна використовувати для побудови не тільки графіків , а і різноманітних полігонів, заданих сукупністю окремих точок.

Розглянемо приклад побудови п`ятикутника за координатами його вершин.

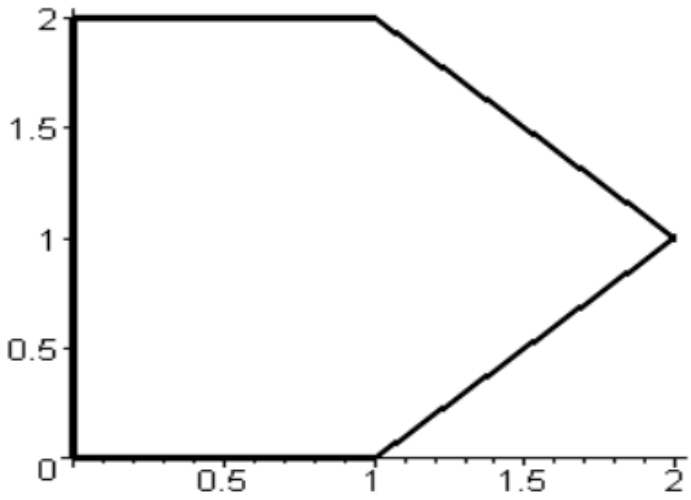
Задамо координати вершин п`ятикутника, та прямі, що з`єднують відповідні вершини.

**> A1:=[0,0]: B1:=[1,0]: C1:=[2,1]: D1:=[1,2]: E1:=[0,2]:**

**>ab:=[A1,B1]:bc:=[B1,C1]:cd:=[C1,D1]:e:=[D1,E1]:ea:=[E1,A1]:**

Виведемо п`ятикутник у вигляді полігону, виберемо параметр товщини лінії, що дорівнює 3.

**> plot([ab,bc,cd,de,ea],color=black,thickness=3);**



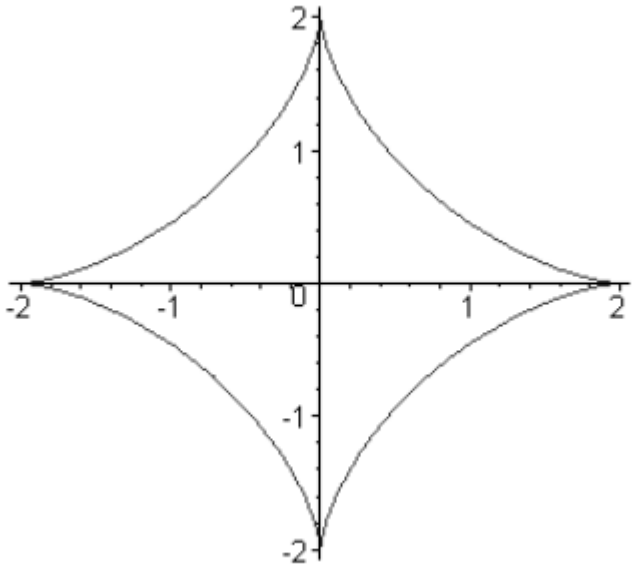

## 4.9Побудова графіків функцій заданих параметрично та функцій в полярній системі координат

В цілому ряді випадків функція може задаватися в параметричному вигляді а саме у вигляді **( ), ( ) 1 2 x = f t y = f t** . При цьому значення параметру **[ , ] 0 1 t ∈ t t** .

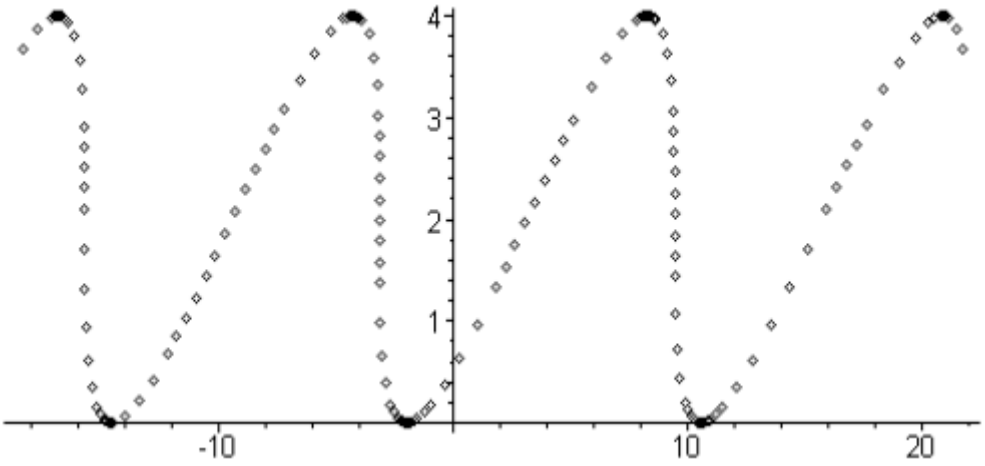
Розглянемо приклади побудови графіків для функцій заданих параметрично на приастероїди.

**> a:=2; a := 2**

**> plot([a*cos(t)^3,a*sin(t)^3,t=-Pi..Pi]);**



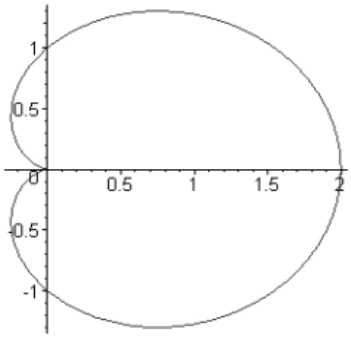

**>plot([a*(t-cos(t)),a*(1-cos(t)),t=-10..10],style=point, color=black);**



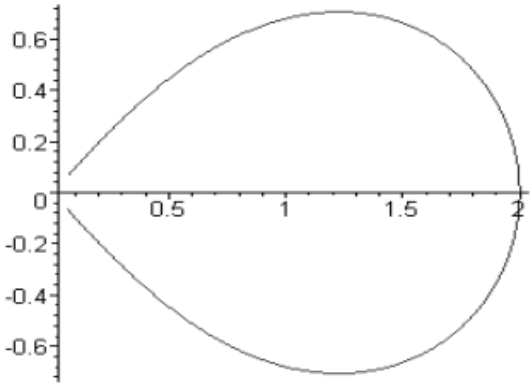

Для побудови графіків функцій в полярній системі координат використовується функція plot з додатковим параметром coords=polar .

Розглянемо приклади побудови графіків функцій в полярній системі координат.Побудуємо карбоїду.:

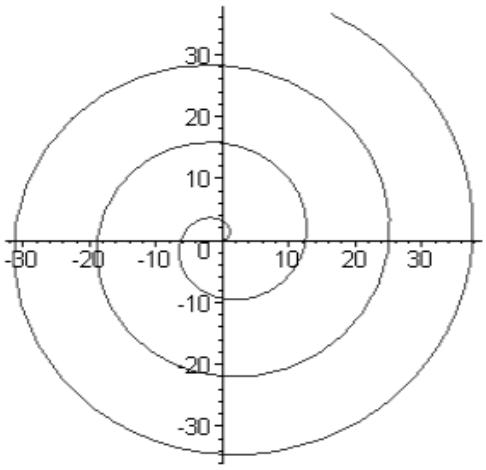
**> plot([(1+cos(Phi)),Phi,Phi=0..2*Pi],coords=polar);**




**> plot([a*sqrt(cos(2*Phi)),Phi,Phi=-Pi/4..Pi/4],coords=polar);**




**> plot([a*Phi,Phi,Phi=0..20],coords=polar);**




## 4.10Побудова трьохвимірних графіків

Трьохвимірними графіками називають графіки поверхонь, які можна задавати у одному з вигляді **в : z = z(x, y), y = y(x, z), x = x(y, z).**Цей спосіб називається **явним завданням поверхні**., або у спосіб, коли визначаються три функції **x = x(u,v), y = y(u,v), z = z(u,v)** від двох параметрів u, v. Крім того для задання поверхні можна використовувати криволінійні ортогональні системи координат: циліндричні, сферичні, параболічні, еліптичні та інші.

## 4.11Особливості застосування функції plot3d

Для побудови графіків трьохвимірних поверхонь в системі MAPLE існує вбудована функція plot3d, яка може бути використана в одному з форматів.

**plot3d(expr1, x=a..b,y=c..d,p)**

**plot3d([exprf,exprg,exprh],u=a..b,v=c..d,p)**

Де epr1 – вираз, що задає поверхні від змінних x , y . eprf, eprg ,-eprh - вирази, що задають рівняння поверхні в параметричній формі від змінних u, v. a, b, c, d – верхня та нижня межі зміни незалежних змінних., p – управляючіпараметри.

## 4.12Побудова поверхонь різними стилями

При побудові поверхонь **MAPLE**пропонує користувачу цілу низку можливостей для надання графіку потрібних властивостей. Для цього користувачможе безпосередньо задавати такі параметри як **style**, **color**, **axes**, надаючи цім

параметрам відповідних значень. В той же час при активізації відповідного графічного об`єкту за допомогою миші у вікні лістінгу **MAPLE** користувач має можливість за допомогою пунктів головного меню **STYLE**, **COLOR**, **AXES**обирати потрібні йому значення відповідних параметрів.

### 4.13Побудова фігур в різних системах координат

Система **MAPLE** дозволяє будувати поверхні , які задані в найбільш поширених відомих криволінійних системах координат, наприклад: циліндричній, сферичній, еліпсоїдальній, тороідальній та багатьох інших. Для зав дання відповідної системи координат користувачу необхідно присвоїти параметру **coords** відповідне значення :

**spherical** - сферичні;

**cylindrical** - циліндричні;

**toroidal** - тороідальні;

**ellipsoidal** - еліпсоїдальні;

### 4.14Зображення поверхонь заданих в параметричній формі

Для зображення поверхонь, заданих в параметричній формі в системі MAPLE використовується наступна форма виклику функції:

**plot3d([exprf,exprg,exprh],u=a..b,v=c..d, p**

Для цієї форми необхідно задати функціональні залежності exprf, exprg, exprh, як функції параметрів u, v.

Розглянемо приклади зображення поверхонь в параметричній формі.

В першому прикладі зобразимо поверхонь гелікоїда, яких задається за допомогою співвідношень: **x = r cosϕ, y = rsinϕ, z = 4ϕ, 0 < r < 2, 0 <ϕ < 4π**

**> plot3d([r*cos(phi),r*sin(phi),4*phi],r = 0..2,phi=0..4*Pi);**

Другий приклад демонструє зображення параметрично заданого тору, параметрична форма якого має вигляд: **x = (b + a cosψ)cosϕ, y = (b + a cosψ)sinϕ, z = a sinψ, 0 <ϕ < 2π , 0 <ψ < 2π**

Параметри **a=1, в=2.**

## Список рекомендованої літератури

1. Аладьев В. Эффективная работа в Maple 6 и 7. М.: Лаборатория Базовых Знаний. 2002.

2. Аладьев В., Богдявичюс М. Maple 6: Решение математических, статистическихи инженерно-физических задач. М.: Лаборатория Базовых Знаний. 2001.

3. Аладьев В., Шишаков М. Автоматизированное рабочее место математика. М.:Лаборатория базовых знаний. 2000.

4. Васильев А. Maple 8. Самоучитель, М.: Диалектика, Вильямс, 2003.

5. Говорухин В., Цибулин В. Введение в Maple. Математический пакет для всех.М.: Мир. 1997.

6. Говорухин В., Цибулин В. Компьютер в математическом исследовании: Maple,MATLAB, LaTeX. СПб.: Питер. 2001.

7. Голоскоков Д. Уравнения математической физики. Решение задач в системеMaple. СПб: Питер, 2004.

8. Дьяконов В. Компьютерная математика. Теория и практика. М.: Нолидж. 2000.

9. Дьяконов В. Maple 8 в математике, физике и образовании. М.: СОЛОН-Пресс,2003.

10.Дьяконов В. Maple 9 в математике, физике и образовании. М.: СОЛОН-Пресс,2004.

11.Кирсанов М. Решебник. Теоретическая механика. М.: Физматлит, 2002.

12.Матросов А. Maple 6. Решение задач высшей математики и механики. БХВ-Петербург, 2001.

13.Очков В. Физические и экономические величины в MathCAD и Maple. М.: Финансы и статистика, 2002.

14.Прохоров Г., Леденев М., Колбеев В. Пакет символьных вычислений Maple. М:Компания Петит, 1997.

15.Сдвижков О.А. Математика на компьютере: Maple 8. М.: Солон-пресс, 2003.