Функції

Функції – унікально іменовані об’єкти, що виконують визначені перетворення своїх аргументів, і при цьому повертають результати своїх перетворень. Результат обчислення функції з одним вихідним параметром підставляється на місце її визову, що дозоляє використовувати функції в математичних виразах. Функції в загальному випадку мають список аргументів (параметрів). Якщо функція повертає декілька значень, то вона записується у вигляді **[Y1, Y2, …] = func(X1, X2, …)**

де Y1, Y2, … - список вихідних параметрів, і X1, X2, … - список вхідних аргументів.

Функції можуть бути *вбудованими* і *зовнішніми*, або m-функціями. Вбудованими є найбільш поширені елементарні функції, наприклад, sin(x) і exp(x), тоді як функція sinh(x) є зовнішньою функцією. Зовнішні функції містять свої означення в m-файлах.

Вбудовані функції зберігаються в відкомпільованому ядрі системи Matlab, тому вони виконуються дуже швидко.

Одна з таких можливостей залючається в застосуванні функції ***inline***, аргументом якої треба в апострофах задати вираз, що задає функцію однієї чи декількох змінних. В приведеному прикладі задана функція двох змінних:

```
>> func1=inline('sin(x).^2+cos(y).^2')
func1 =
Inline function:
func1(x,y) = sin(x).^2+cos(y).^2
>> func1(1,0)
ans =
1.7081
```

Можна також задавати свої функції у вигляді m-файлів. У вікні редактора m-файлів створити m-файл з ім’ям func3 і лістингом:

```
function y=func3(x,y)
y=sin(x).^2+cos(y).^2
```

Записавши його на диск, командою **type func3** можна вивести лістинг функції. Звертання до функції відбувається як до звичайної, підставляємо в список параметрів потрібні змінні.

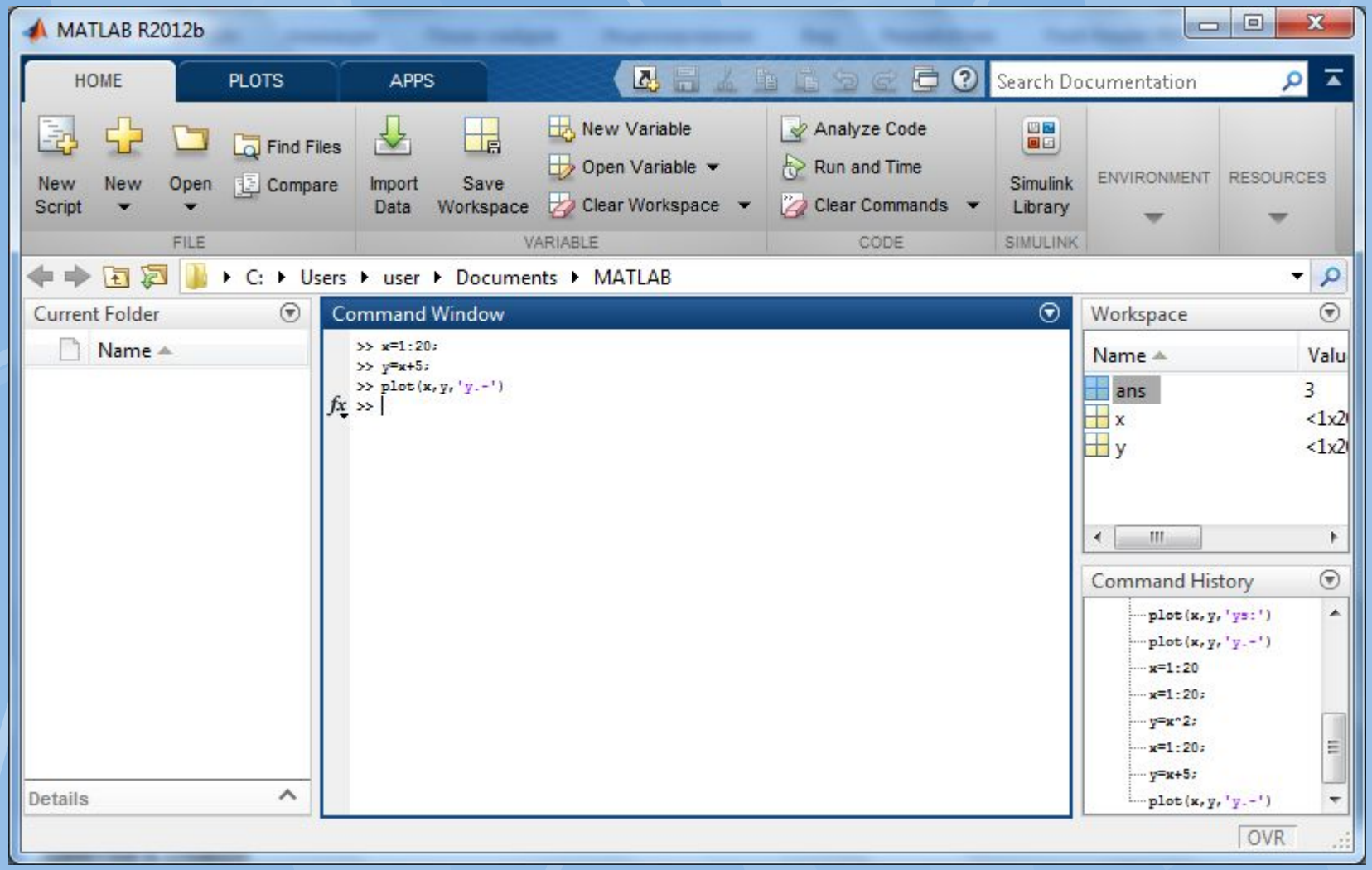
4.3. Побудова графіків функції на площині

Одна з переваг системи MATLAB – широкий спектр засобів графіки: від команд побудови простих графіків функцій однієї змінної в декартовій системі координат і до комбінованих презентаційних графіків з елементами анімації.

Команда **plot(x,y,'s')** використовується для побудови графіків функцій однієї змінної у декартовой системі координат на площині.

Координати точок **(x, y)** беруться з векторів однакового розміру **x,y; s** - рядок з трьох символів:

1-й символ	2-й символ	3-й символ
керування кольором	відмітка символом	відображення лінії
y	.	пробел
m	o	-
c	x	:
r	+	-.



Приклад.

>>x=1:20;

>>y=x+5;

>>plot(x,y,’y.-’)

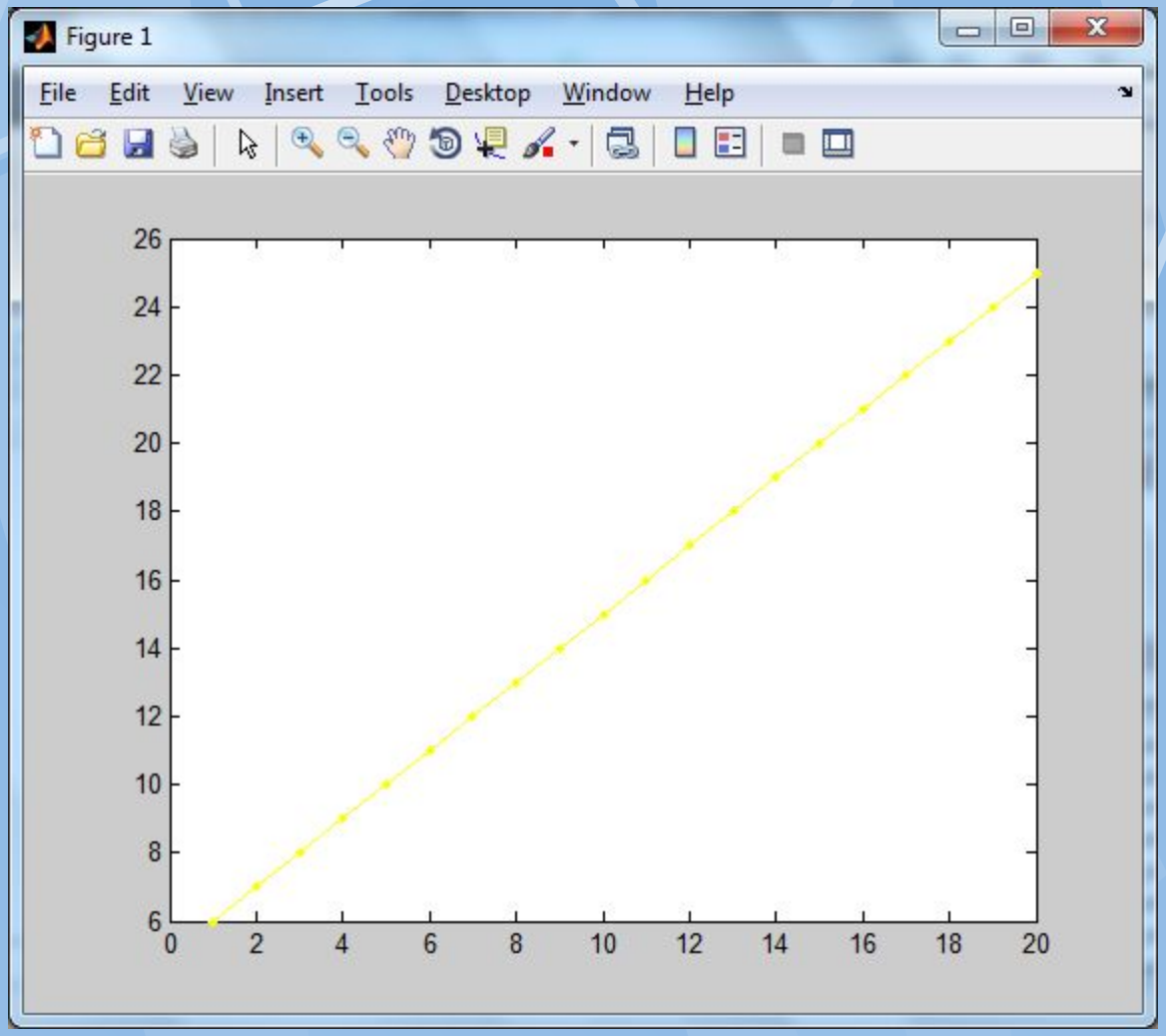


Рис.5. Графік функції з параметром s='y.-'

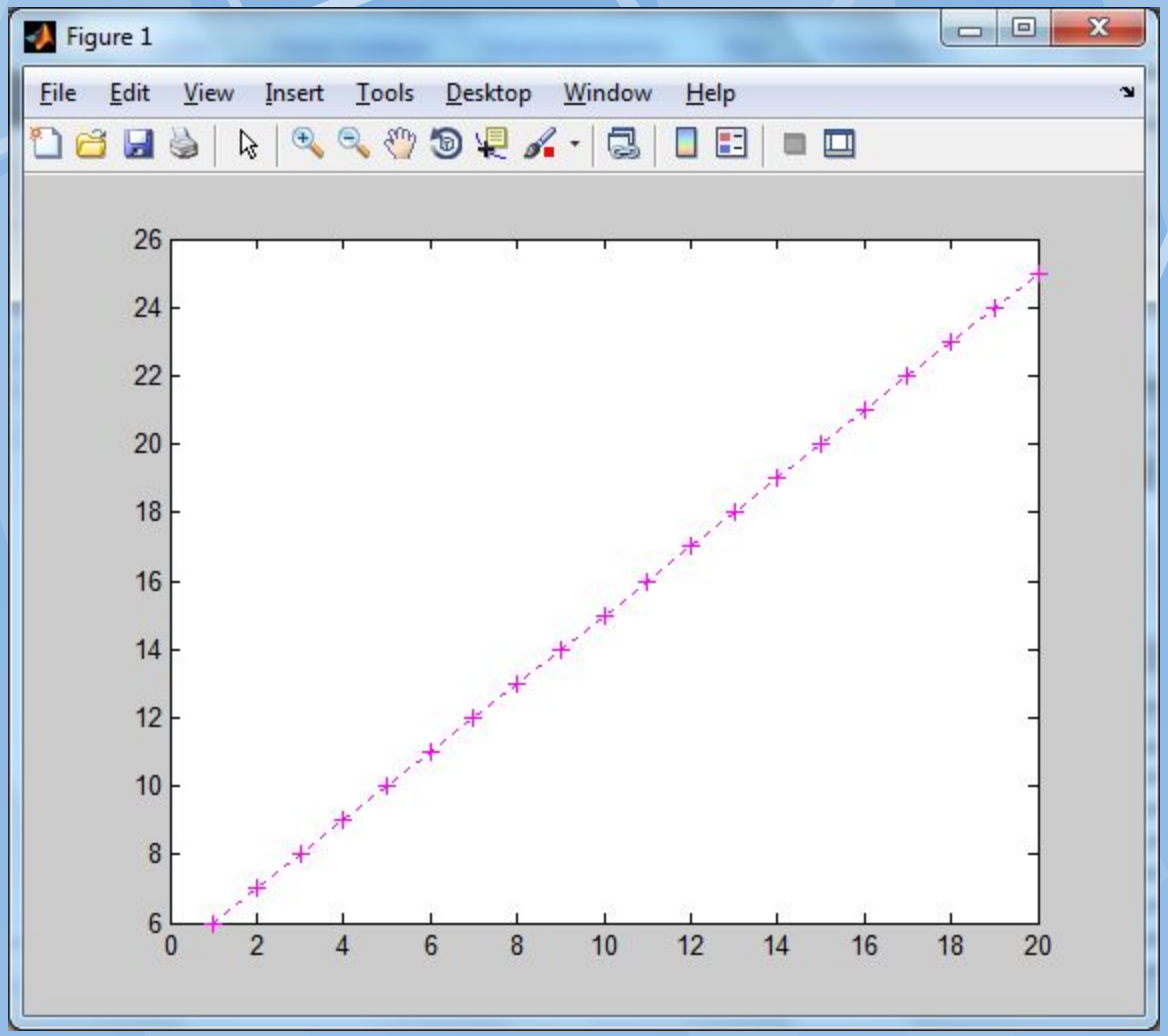


Рис.6. Графік функції з параметром s=‘m+:’

Додаткові команди при побудові графіків

grid on/off– побудова/зняття ліній сітки.
title(‘текст‘)– нанесення тексту на графік(зверху).
xlabel(‘текст‘)– нанесення тексту на підписX-вісі.
ylabel(‘текст‘)– нанесення тексту на підписY-вісі.
legend()– опис елементів графіка.
text()– підпис графіка.

Можуть також будуватись спеціальніX-Y графіки:
bar, comet, hist, polar, compass, errorbar, feather, fplot, rose, stairs, stem.

Приклад.

```
x=1:20;
y=x+5;
plot(x,y,'r.-') ;
grid on;
title ('текст1');
text(10,10,'\leftarrow  (10;10)','FontSize',18)
```

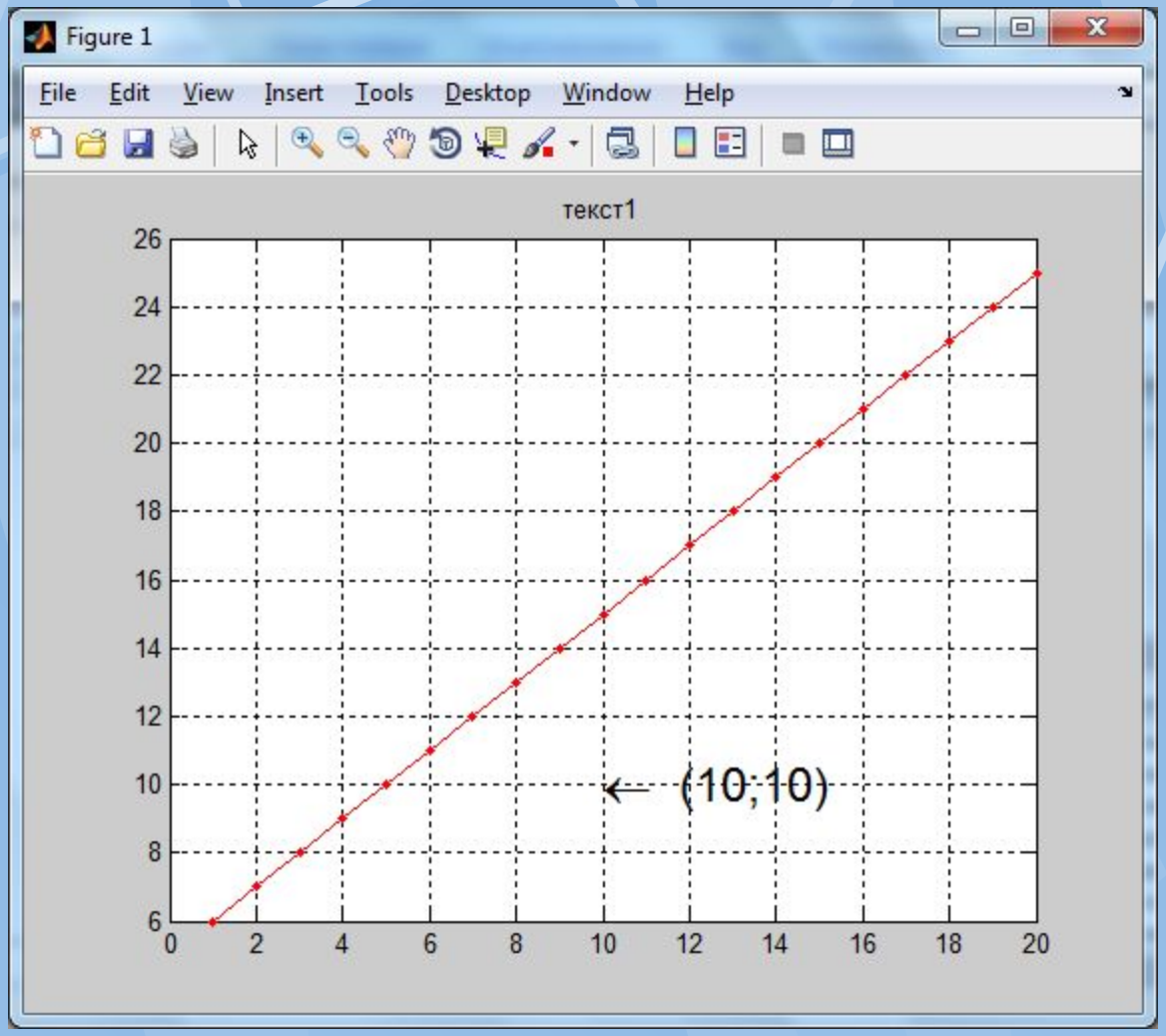
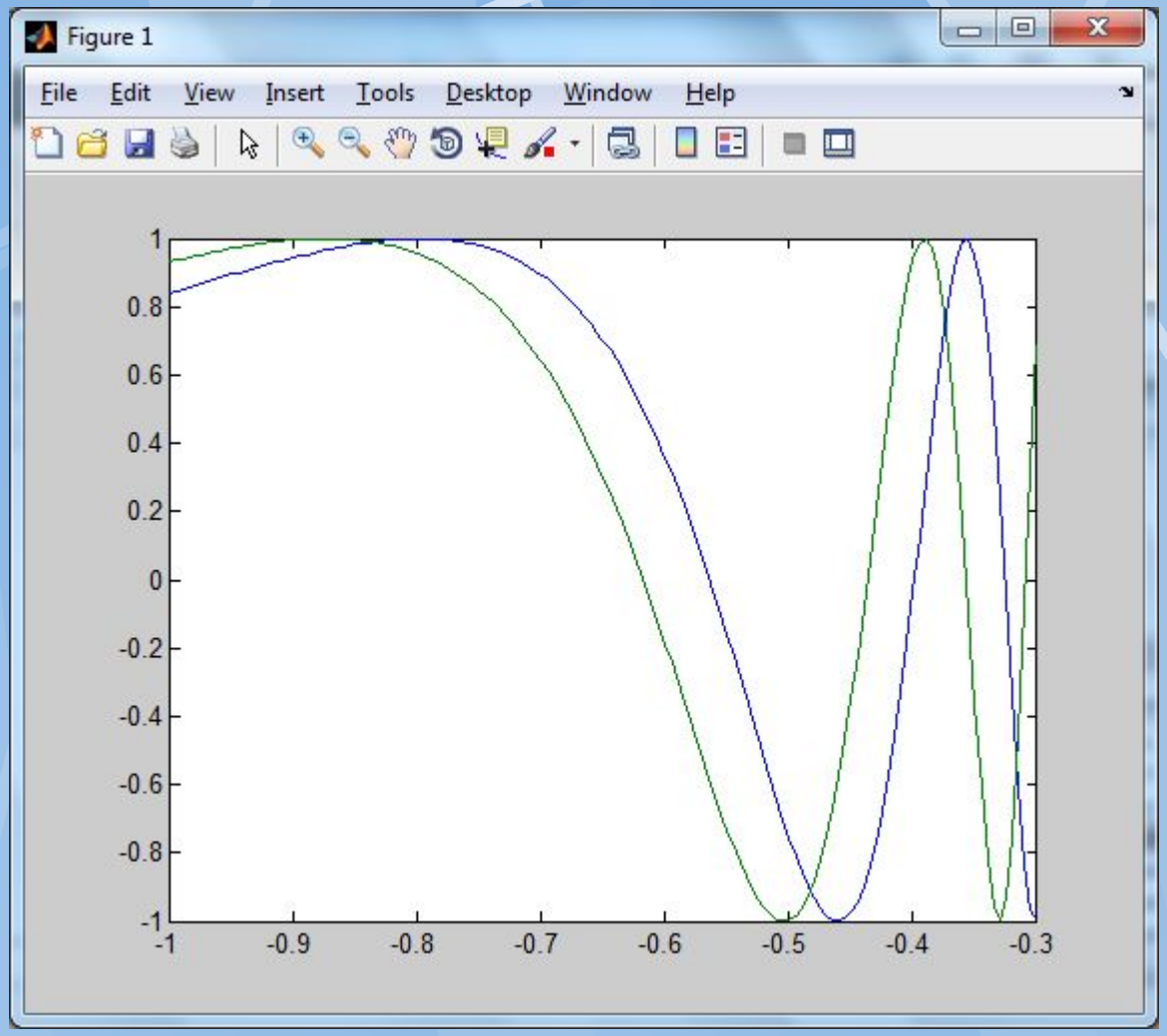


Рис.7. Графік функції у з використанням додаткових команд

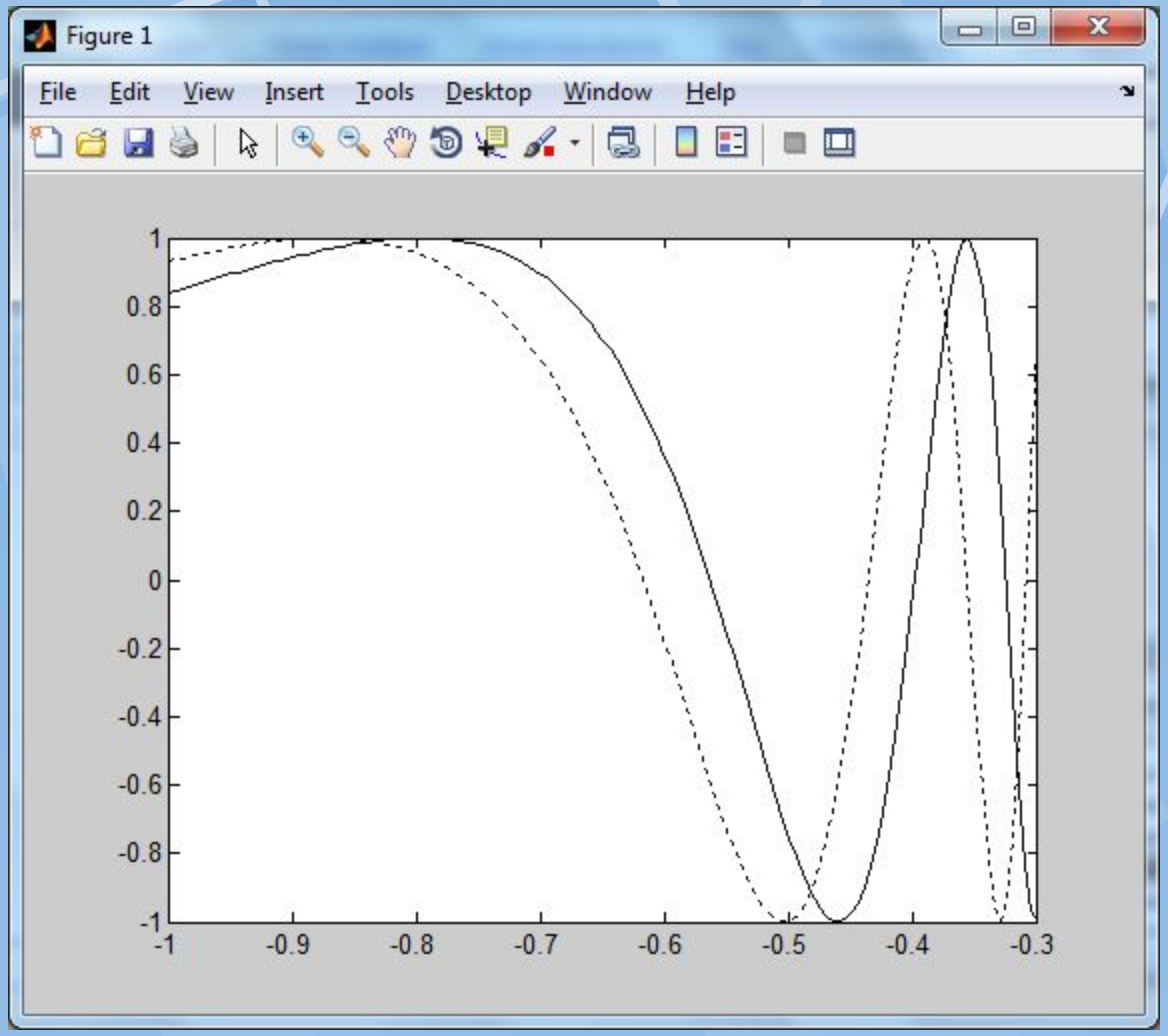
Порівняння декількох функцій зручно виконувати, якщо їх графіки відобразити в одній системі координат.




```
>> x  = -1:0.005:-0.3;
>> f = sin(x.^-2);
>> g = sin(1.2*x.^-2);
>> plot(x, f, x, g)
```

Також з допомогою команди plot можна задати стиль та колір ліній, наприклад:

>> plot(x, f, ‘k-‘, x, g, ‘k:’)



Побудова графіків функції в тривимірному просторі

Тривимірні(3-D) графіки малюються приблизно так само як і X-Y графіки. Замість команди plot використаються її 3-D аналог **plot3(x,y,z)** – побудова ліній і точок в 3-D просторі.

Можуть також будуватись спеціальні3-D графіки:

contour – лінії рівня,

contour3 - 3-мірні лінії рівня,

mesh – 3-D сітчата поверхня,

meshc -3-D сітчата поверхня з проекцією ліній постійного рівня,

meshz - 3-D сітчата поверхня з площиною відліку на нульовому рівні,

surf – затінена3-D сітчата поверхня,

surfc - затінена3-D сітчата поверхня з проекцією ліній постійного рівня,

surfl - затінена3-D сітчата поверхня з підсвіткою.

4.4. Мова програмування і побудова М-файлів.

Основні керуючі структури алгоритмічної мови MATLAB

Команди мови, як і команди командного вікна, записуються по одній в рядок. Для продовження її на наступний рядок використовуєм<…>.

Розгалуження

a)If умова
Оператори;
end

b)If умова
Оператори1;
else
Оператори2;
end

Оператори циклу

a) з передумовою
while умова
Оператори;
end

b) з лічильником
for k=e1:e2:en
Оператори;
end

Завдання. Обчислимо значення синуса при 20 значень аргумента від 0.2 до 4 з кроком 0.2:

```
>> i=1;
>> while i<= 20 x=i/5;
si=sin(x);
disp([x si]);
i=i+1;
end
   0.2000    0.1987

   0.4000    0.3894

   0.6000    0.5646

   0.8000    0.7174

   …

   3.6000   -0.4425

   3.8000   -0.6119

   4.0000   -0.7568
```

Опис функції

Опис функції здійснюється в окремому **М-файлі** з назвою функції.

Синтаксис цього файлу (**назва.m**) :

Function [R1 R2 … Rn]=назва(x1,x2,…,xn)

xi – вхідні формальні параметри

Ri – вихідні формальні параметри

% коментар

оператори

return

end

Всі поіменовані данні, які можуть використовуватись в тілі функції є або локальними або формальними вхідними або вихідними параметрами.

Інформацію, розміщену в тексті коментаря можна відобразити командою **help** назва, а весь текст М-функції можна надрукувати командою **type** назва.

Локальні та глобальні змінні

Глобальними називаються змінні означенні з допомогою атрибуту **global.** Наприклад, **global z1 z2 z3.**

Локальними називаються змінні, які використовуються лише у данній М-функції, М-файлі.

Для збереження інформації між викликом М-функції (статичну) використовується атрибут **persistent.**
Наприклад, **persistent t1 t2 t3.**

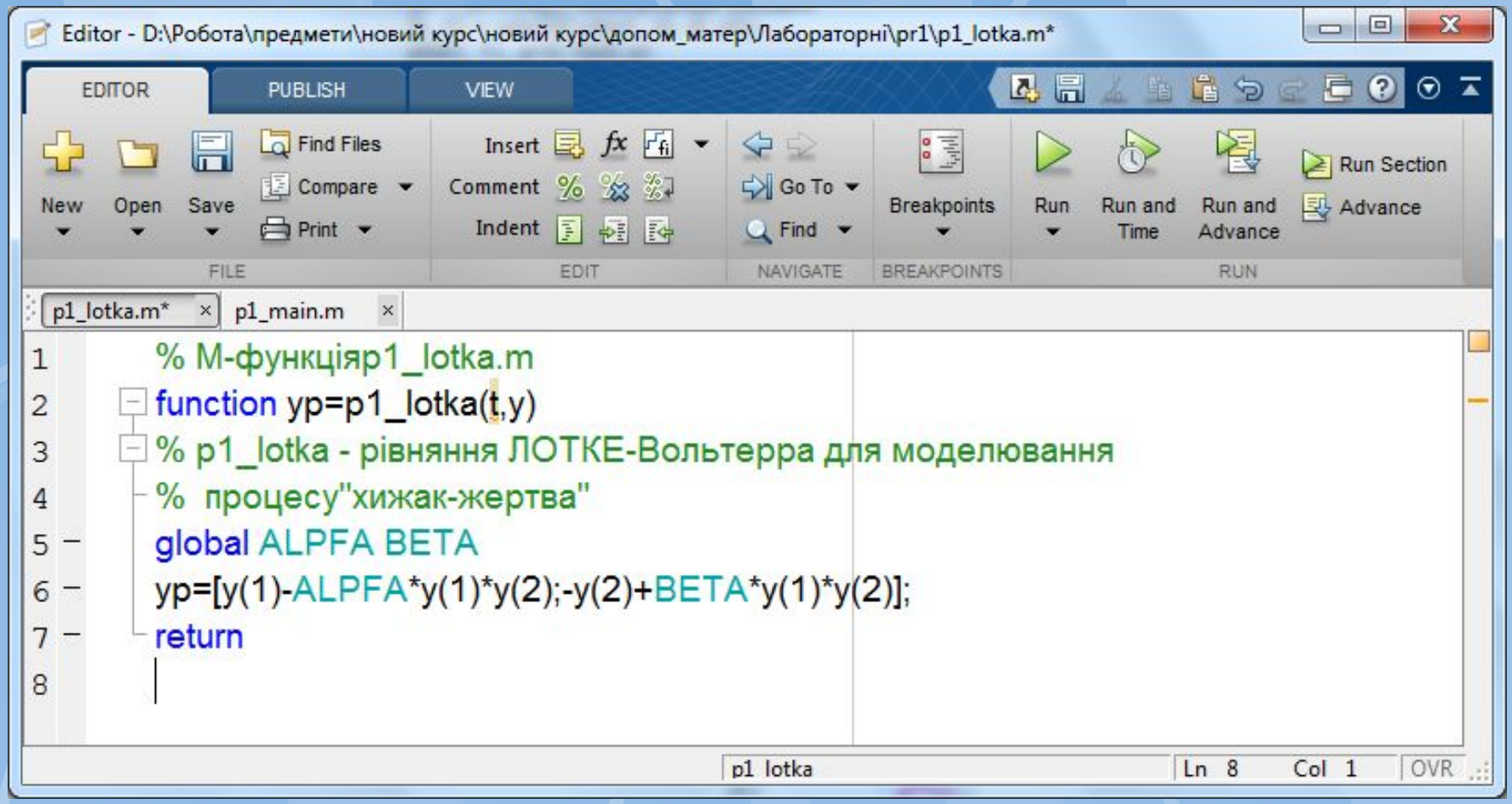
М-сценарії та їх виклик

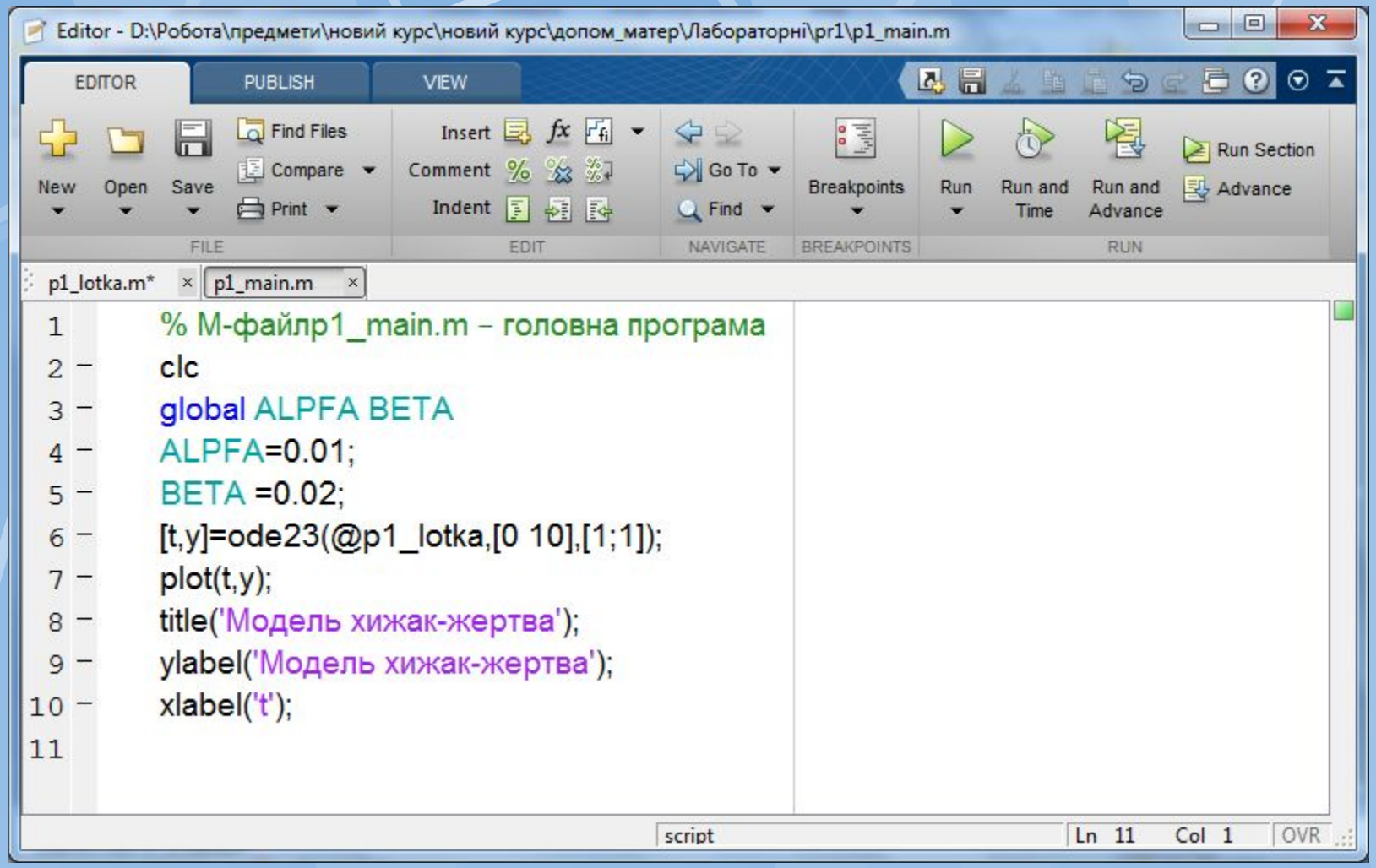
Треба зазначити що М-файли, які створені користувачем, умовно можна поділити на дві групи:

- ***файли-сценарії,*** які не мають вхідних параметрів;
- ***файли-функції***, які мають вхідні параметри.

Файл-функція відрізняється від файла-сценарія тим що, в функції обмін інформації виконується через апарат формальних-фактичних параметрів. Змінні в середині тіла функції, як правило локальні. В М-файлі доступна вся інформація, яка є в робочій області.

Виклик М-файлу відбувається аналогічно виклику команди.


Editor - D:\Робота\предмети\новий курс\новий курс\допом_матер\Лабораторні\pr1\p1_lotka.m*
EDITOR
PUBLISH
VIEW
New
Open
Save
Find Files
Compare
Print
FILE
Insert
Comment
Indent
EDIT
Go To
Find
NAVIGATE
Breakpoints
BREAKPOINTS
Run
Run and Time
Run and Advance
Run Section
Advance
RUN
p1_lotka.m*
p1_main.m
1 % M-функціяp1_lotka.m
2 function yp=p1_lotka(t,y)
3 % p1_lotka - рівняння ЛОТКЕ-Вольтерра для моделювання
4 % процесу"хижак-жертва"
5 global ALPFA BETA
6 yp=[y(1)-ALPFA*y(1)*y(2);-y(2)+BETA*y(1)*y(2)];
7 return
8
p1_lotka
Ln 8
Col 1
OVR


Editor - D:\Робота\предмети\новий курс\новий курс\допом_матер\Лабораторні\pr1\p1_main.m
EDITOR
PUBLISH
VIEW
New
Open
Save
Find Files
Compare
Print
FILE
Insert
Comment
Indent
EDIT
Go To
Find
NAVIGATE
Breakpoints
BREAKPOINTS
Run
Run and Time
Run and Advance
Run Section
Advance
RUN
p1_lotka.m*
p1_main.m
1 % M-файлр1_main.m – головна програма
2 clc
3 global ALPFA BETA
4 ALPFA=0.01;
5 BETA =0.02;
6 [t,y]=ode23(@p1_lotka,[0 10],[1;1]);
7 plot(t,y);
8 title('Модель хижак-жертва');
9 ylabel('Модель хижак-жертва');
10 xlabel('t');
11
script
Ln 11
Col 1
OVR

Функція ode23 призначена для чисельного інтегрування систем ЗДР. Вони застосовуються як для розв'язування простих ДР так і для моделювання складних динамічних систем.

Функція [t, X] = ode23('< ім'я функції >', t0, tf, x0, tol, trace)

Вхідні параметри:

'<ім'я функції>' - стрічкова змінна, яка являється ім'ям М-файла, в якому обчислюються праві частини системи ЗДР;

t0 - початкове значення часу;

tf - кінцеве значення часу;

x0 - вектор початкових умов;

tol – задана точність; за замовчуванням для ode23 tol = 1.e-3, для ode45 tol = 1.e-6);

trace – прапорець, який керує вивід проміжних результатів; за замовчуванням рівний 0, що відміняє вивід проміжних даних.

Вихідні параметри:

t – поточний час;

X – двох-вимірний масив, кожний стовбець якого відповідає одній змінній.

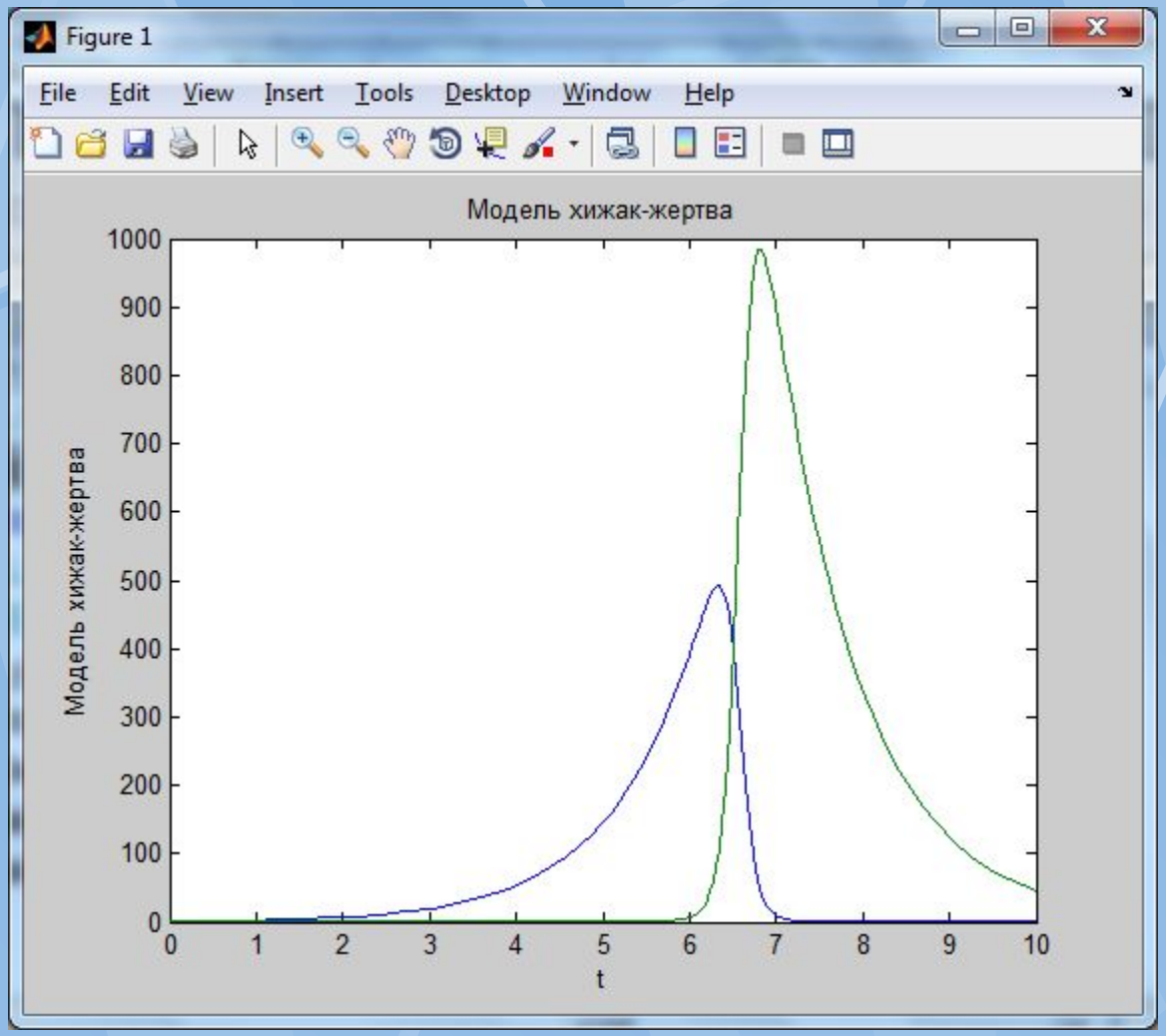


Figure 1
File Edit View Insert Tools Desktop Window Help
Модель хижак-жертва
Модель хижак-жертва
t
0
100
200
300
400
500
600
700
800
900
1000
0 1 2 3 4 5 6 7 8 9 10

Лекція №2

Тема 3: Математичний пакет Mathematica

План лекції

1. Загальна характеристика пакету.
2. Основи роботи в пакеті.
3. Елементи програмування.

1. Загальна характеристика пакету.

Mathematica – система комп'ютерної алгебри, яка використовується в багатьох наукових, інженерних, математичних і комп'ютерних областях.

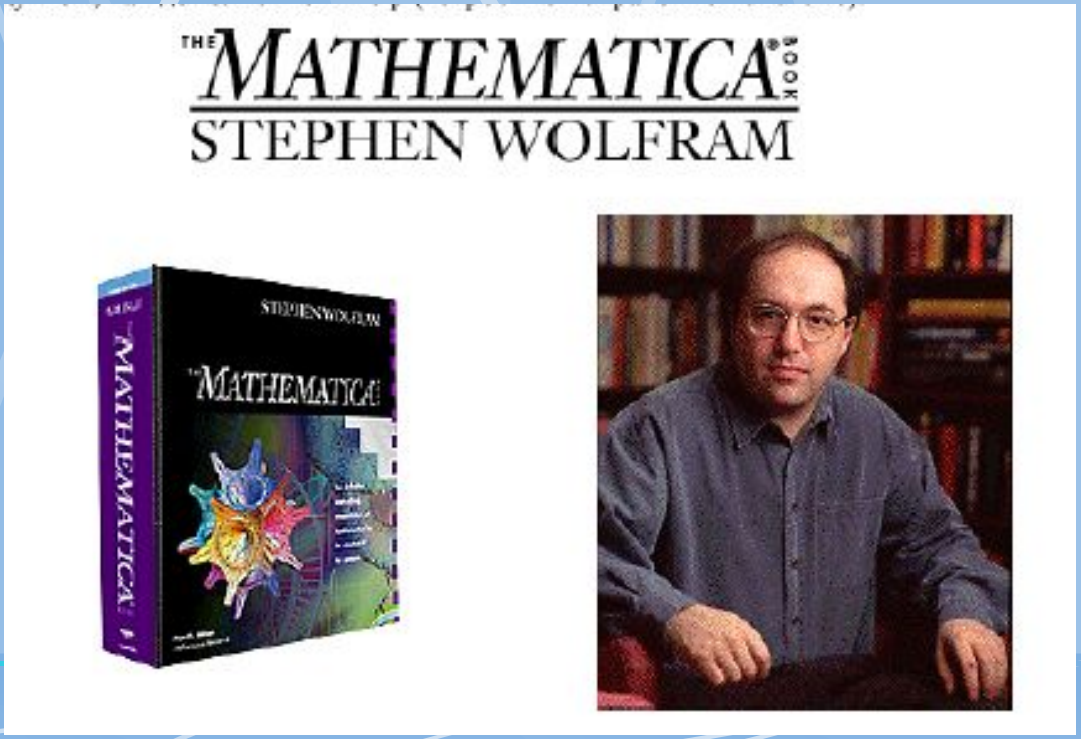
Спочатку система була придумана Стівеном Вольфрамом, в даний час розробляється компанією WolframResearch.

WolframMathematica – це програмне забезпечення, не тільки для математичних обчислень, це набагато більше: від моделювання та симуляції, візуалізації, документації, до створення веб-сайтів.

Mathematica володіє можливістю здійснювати виклики функцій і приймати виклики з C, .NET, Java та інших мов, генерувати C код, компілювати автономні бібліотеки і виконувані файли.

Перша версія Mathematica з’явилась 1 червня 1988 року.

Перша версія - 1988 рік
Друга версія - 1991 рік
Третя версія - 1996 рік
Четверта версія - 1999 рік
П’ята версія - 2003 рік
…
Десята версія - 2015 рік
Одинадцята - 2018 рік



При розробці і експлуатації самих різних виробів використовується система Mathematica.

Система Mathematica складається з

- ядра **Mathematica Kernel** (обчислювальний механізм)
- зовнішньої оболонки **Mathematica** (візуальний інтерфейс).

Після установки пакета в головному меню створюються ярлики на два файли: Mathematica і Mathematica Kernel.

Справа в тому, що ярлик Mathematica Kernel запускає ядро пакету, яке робить всі обчислення, а ярлик Mathematica запускає інтерфейсну частину пакету рис.1.

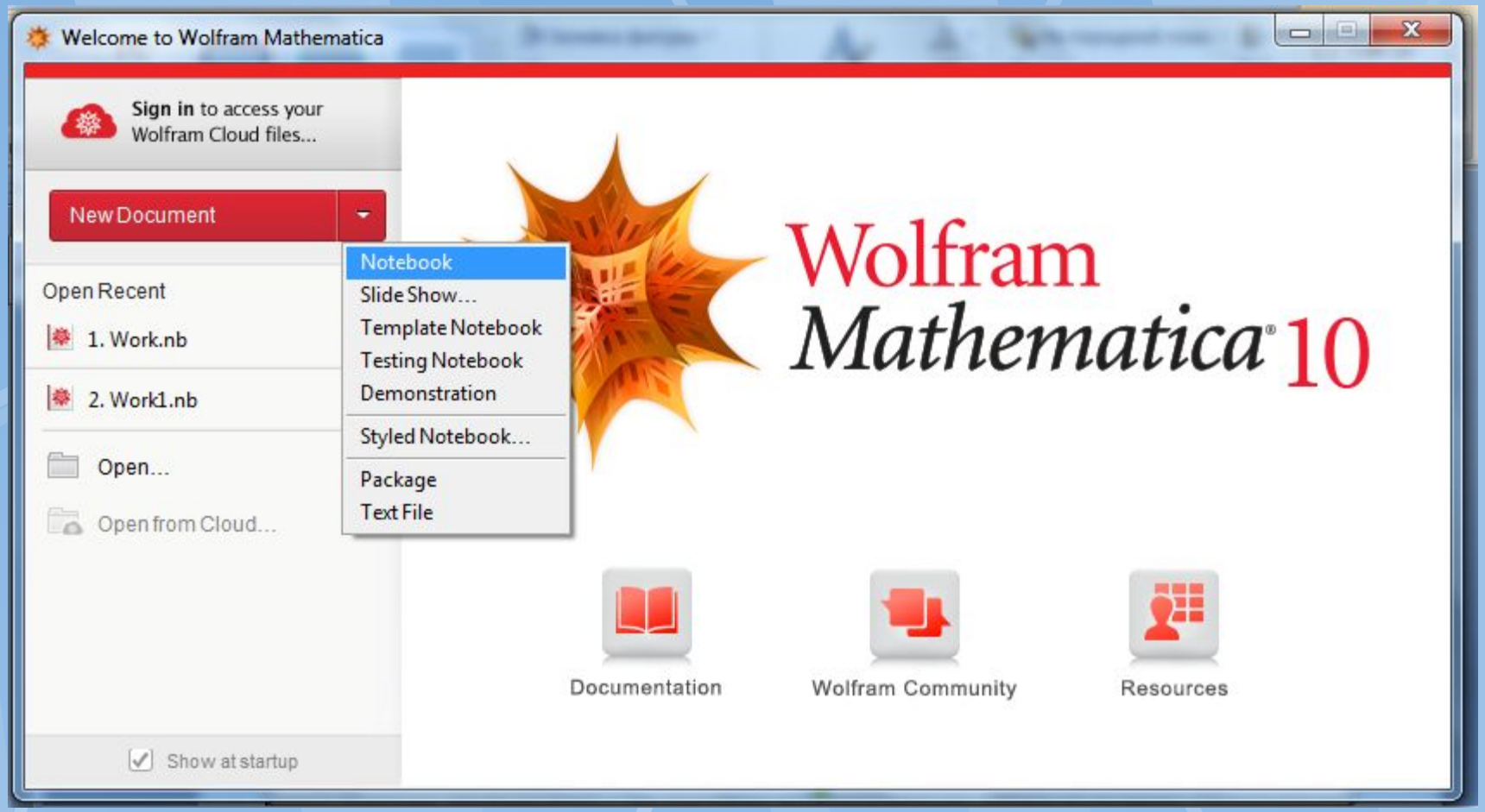


Рис.1.

Інтерфейс системи Mathematica реалізує відображення вікон, палітр, панелей інструментів, знаків і розташування їх у різному вигляді і в різних місцях екрану монітора. Головне вікно програми показано на рис.2.

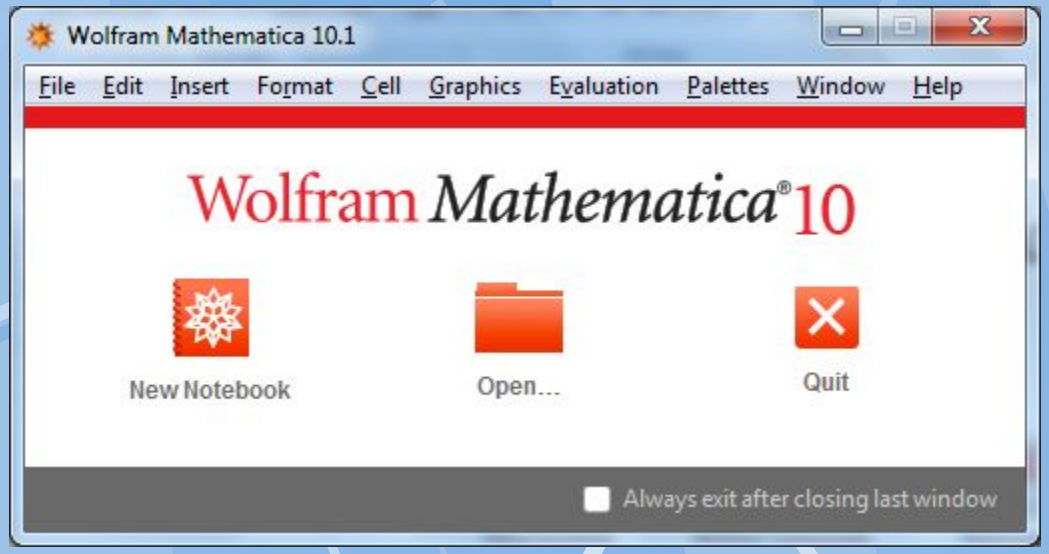


Рис. 2. Головне вікно

Вікно складається з основного меню програми (у верхній частині екрана), вікна робочого документа або «блокноту» (notebook) і панелі (палітри) для введення спец символів і знаків найбільш вживаних математичних операцій.

Основне меню програми містить кілька сотень найменувань пунктів меню, підменю, команд, функцій. Вивчити їх відразу неможливо: з короткого опису не можна зрозуміти зміст.

Зміст пунктів меню, підменю, команд можна зрозуміти тільки в процесі роботи з системою.

Після виконання команди **New Notebook,** з’являється вікно робочого документу рис.3.

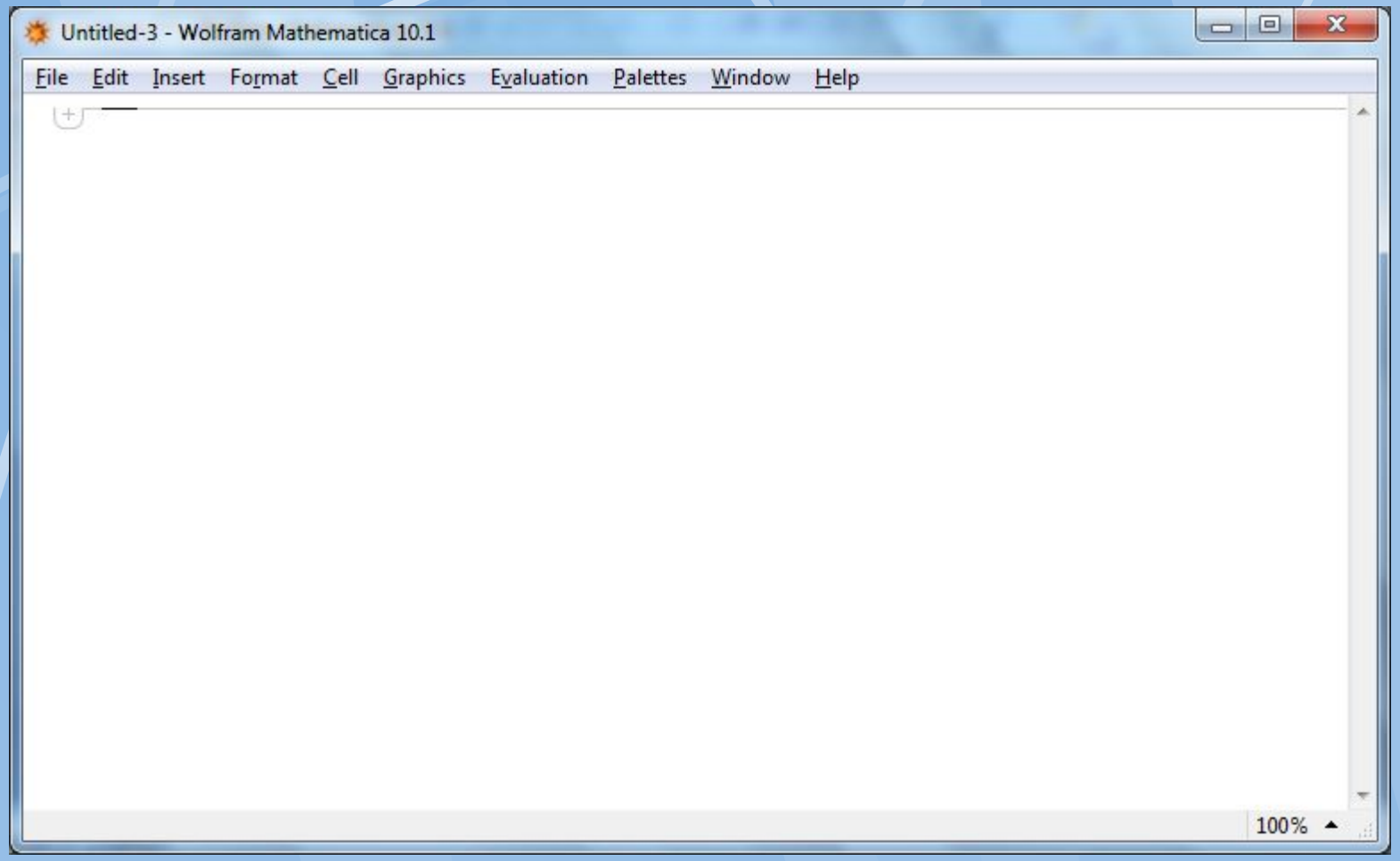


Рис.3.

Вікно робочого документа або блокнот складається з комірок. Комірку можна порівняти з параграфом у текстовому редакторі. Вся інформація, яка є в блокноті, зберігатися в його комірках. Як тільки в порожньому новому файлі набирається хоча б один символ, Mathematica створить для нього комірку. Комірка також є мінімальною одиницею, яку можна обчислити.

Усі комірки можна розділити на три типи:

- **комірки введення** – в них задаються команди (формули), які будуть обчислені;
- **комірки результату** – у них Mathematica виводить результат обчислень;
- **не обчислювані комірки** – комірки з текстом, заголовки і все інше, що вводить користувач і обчислювати не порібно.

Будь-які клітинки можна об’єднувати і розбивати за допомогою команд меню **Cell: Divide Cell** (розбити клітинку) і **Merge Cells** (об’єднати комірки).

Введення даних здійснюється в комірки. Пакет підтримує кирилицю і грецькі літери нарівні з англійським алфавітом. Можна називати змінні російськими літерами, також як і грецькими. У той же час, ідентифікатори розрізняються по регістру (тобто змінна А не те саме, що змінна а).

Для швидкого доступу до функцій, розробники Mathematica ввели спеціальні типи вікон, які називаються палітрами. Палітри містять вікна з кнопками, які виконують дії.

Дії можуть бути абсолютно різними: від додавання грецької букви, до розкриття дужок у алгебраїчному виразі.

Різні палітри доступні через меню **Palettes** (рис.4).

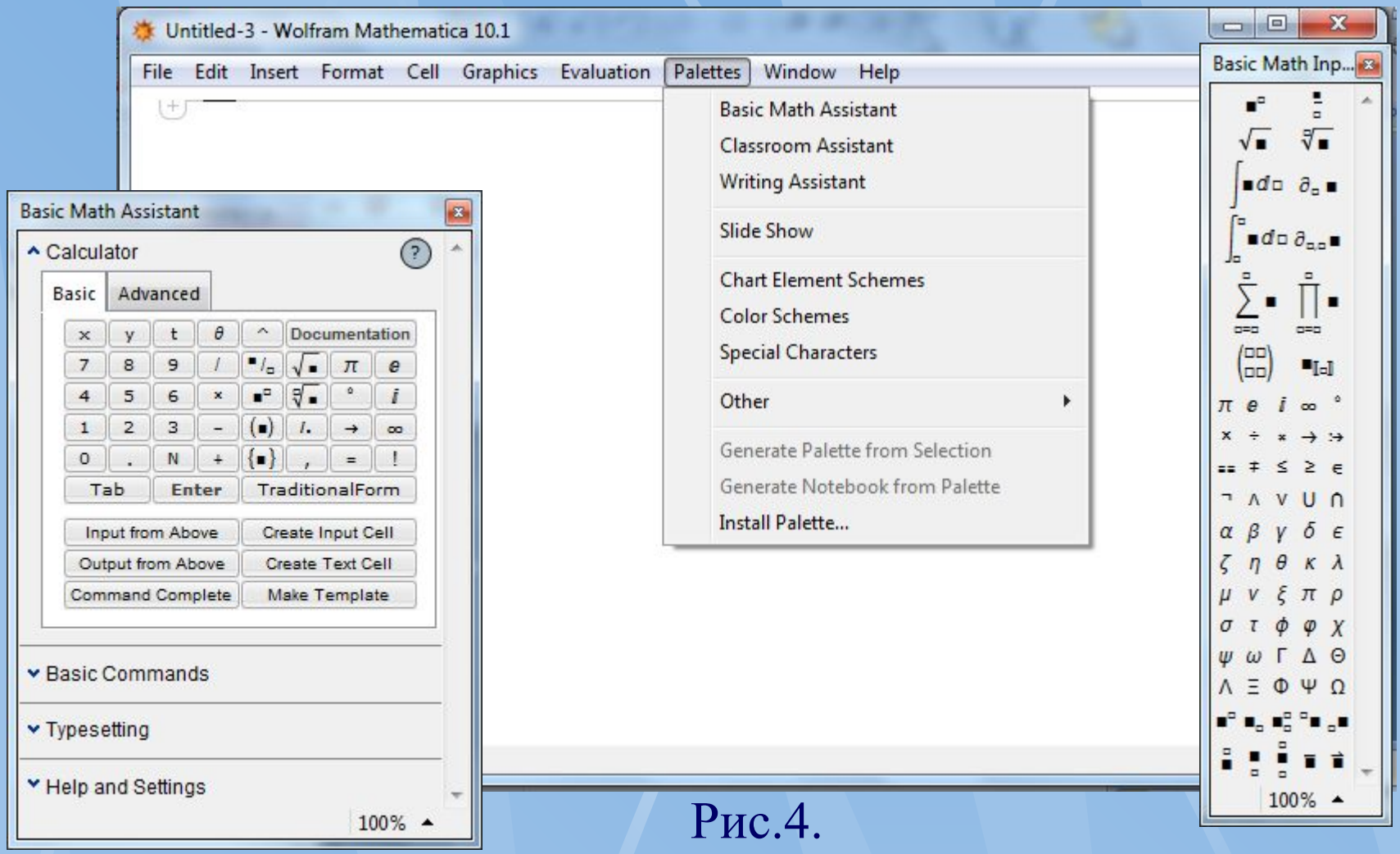


Рис.4.

Wolfram Mathematica має розвинені засоби форматування тексту. За допомогою їх можна розбивати блокнот на розділи, вводити пояснювальний текст і т.д. Стилі можна задати як всьому блокноту, так і окремій комірці цілком, або частково. Також можна змінити відображення всіх стандартних стилів і додати нові.

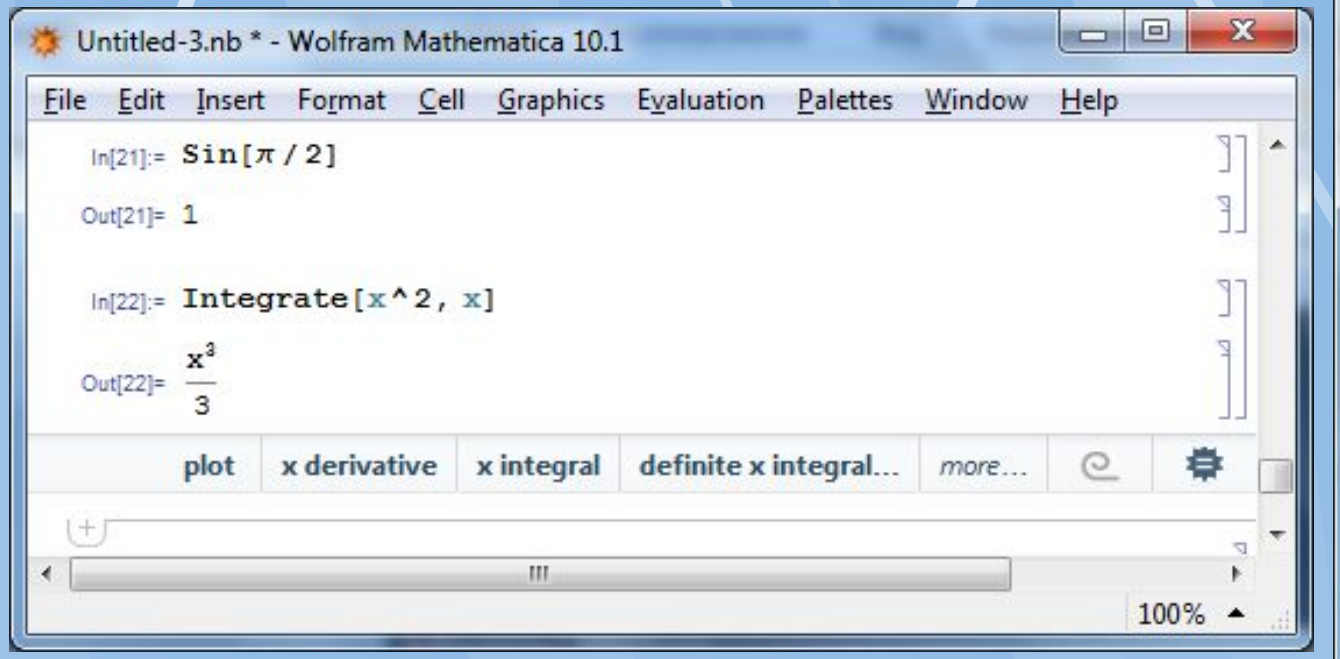
2. Основи роботи в пакеті.

Інтерфейс Mathematica =

= панель меню + робоча область + палітри

В робочій області вводяться команди.

Для їх виконання натискають <Shift+Enter>



Можливості пакету Mathematica

Пакет Mathematica дозволяє розв'язувати наступні задачі:

1. Виконувати алгебраїчні розрахунки та операції з виразами
2. Розв'язувати алгебраїчні рівняння та системи
3. Виконувати інтегрування та диференціювання
4. Розв'язувати диференціальні рівняння
5. Створювати програмні коди
6. Виконувати числові розрахунки
7. Створювати графіку та анімацію

Основні арифметичні оператори:

1. **+ додавання**
2. **- віднімання**
3. *** множення**
4. **/ ділення**
5. **^ піднесення в степінь**

Додаткові оператори:

6. **% результат виконання останньої команди**
7. **%% результат виконання передостанньої команди**

Функції

Основні правила:

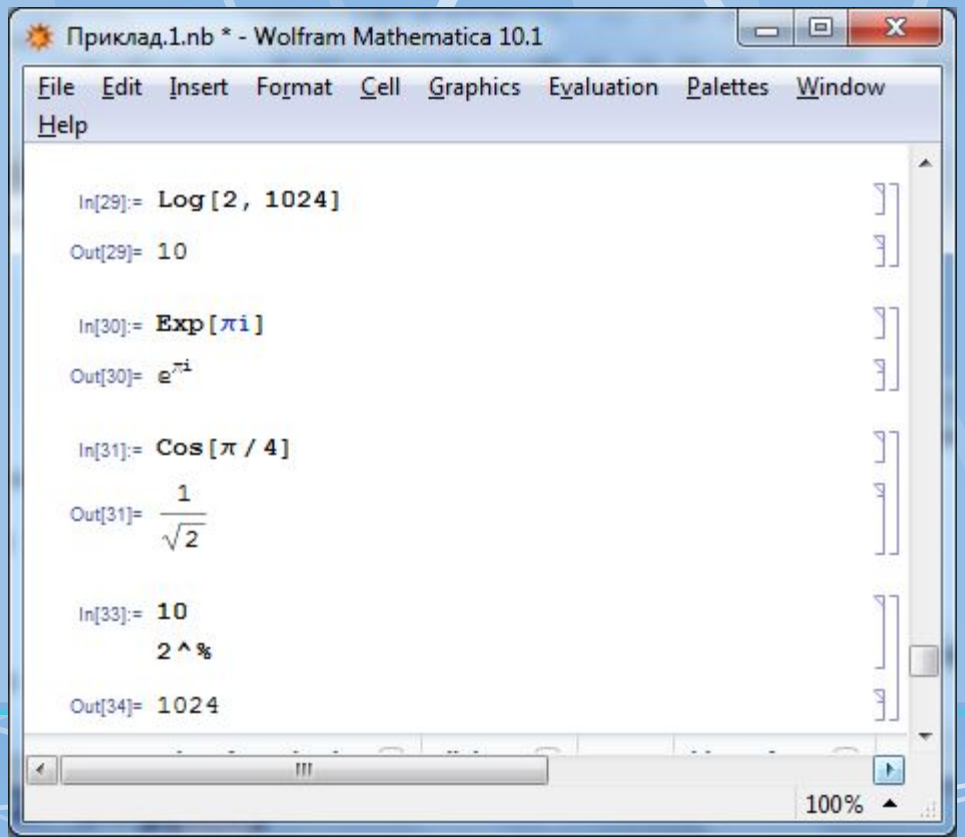
1. Назва функції - з великої літери

2. Аргумент - в квадратних дужках

3. Аргументи розділяються комами

Наприклад:

1. **Sin[x]**
2. **Log[a,b]**
3. **Exp[πi]**



Математичні функції

Функція	Опис	Функція	Опис
Abs[x]	Модуль комплексного числа x	**Cot[z]**	Котангенс від z
ArcCos[z]	Арккосинус від z	**Coth[z]**	Котангенс гіперболічний від z
ArcCosh[z]	Арккосинус гіперболічний від z	**Csc[z]**	Косеканс від z
ArcCot[z]	Арккотангенс від z	**Csch[z]**	Косеканс гіперболічний від z
ArcCoth[z]	Арккотангенс гіперболічний від z	**Exp[z]**	Експонента
ArcCsc[z]	Арккосеканс від z	**Im[z]**	Уявна частина z
ArcCsch[z]	Арккосеканс гіперболічний від z	**Log[a,z]**	Логарифм числа z за основою a
ArcSec[z]	Арксеканс від z	**Re[z]**	Дійсна частина z
ArcSech[z]	Арксеканс гіперболічний від z	**Sec[z]**	Секанс від z
ArcSin[z]	Арксинус від z	**Sech[z]**	Секанс гіперболічний від z
ArcSinh[z]	Арксинус гіперболічний від z	**Sign[x]**	Знак числа x
ArcTan[z]	Арктангенс від z	**Sin[z]**	Синус від z
ArcTanh[z]	Арктангенс гіперболічний від z	**Sinh[z]**	Синус гіперболічний від z
Arg[z]	Аргумент числа z	**Sqrt[z]**	Корінь квадратний від z
Conjugate[z]	Комплексно спряжене до z	**Tan[z]**	Тангенс від z
Cos[z]	Косинус від z	**Tanh[z]**	Тангенс гіперболічний від z
Cosh[z]	Косинус гіперболічний від z		

Приклади розрахунків - числа

Приклад 2. Числа

Приклад 2.nb

123!

12 146 304 367 025 329 675 766 243 241 881 295 855 454 217 088 483 382 315 328 918 161 829 235 892 362 167 668 831 156 960 612 640 202 170 ↓
735 835 221 294 047 782 591 091 570 411 651 472 186 029 519 906 261 646 730 733 907 419 814 952 960 000 000 000 000 000 000 000 000 000

N[%]

1.21463×10^{205}

N[π]

3.14159

N[π, 100]

3.14159265358979323846264338327950288419716939937510582097494459230781640628620899862803482534211706 8

N[e]

2.71828

N[E, 100]

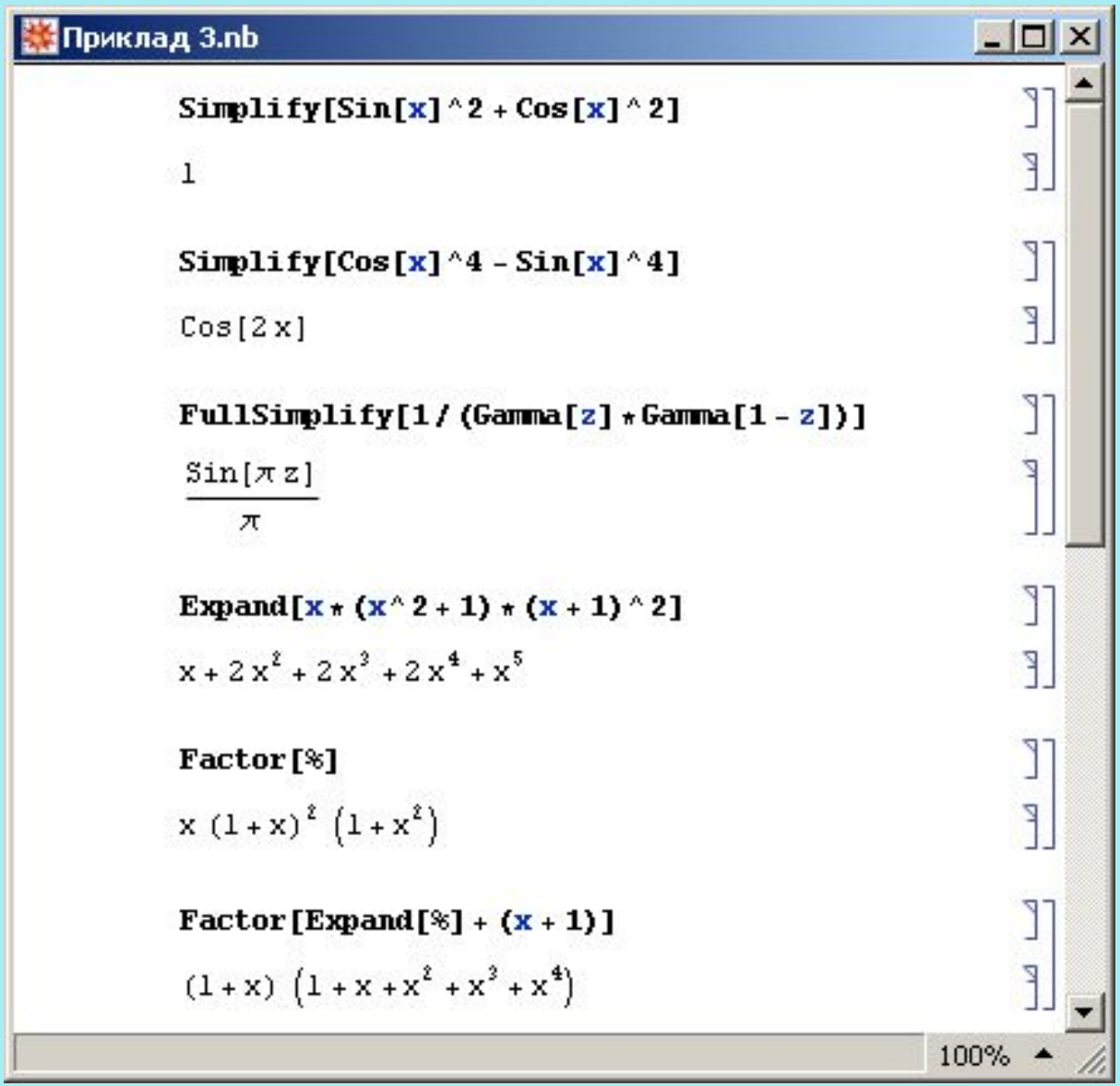
2.71828182845904523536028747135266249775724709369995957496696762772407663035354759457138217852516642 7

100%

N[*expr*]
gives the numerical value of *expr*.
N[*expr*,*n*]
attempts to give a result with *n*-digit precision.

Приклади розрахунків - вирази

Приклад 3. Операції з виразами



Команди для спрощення виразів:

Simplify[ ]

FullSimplify[ ]

Команда для розкладу виразу:

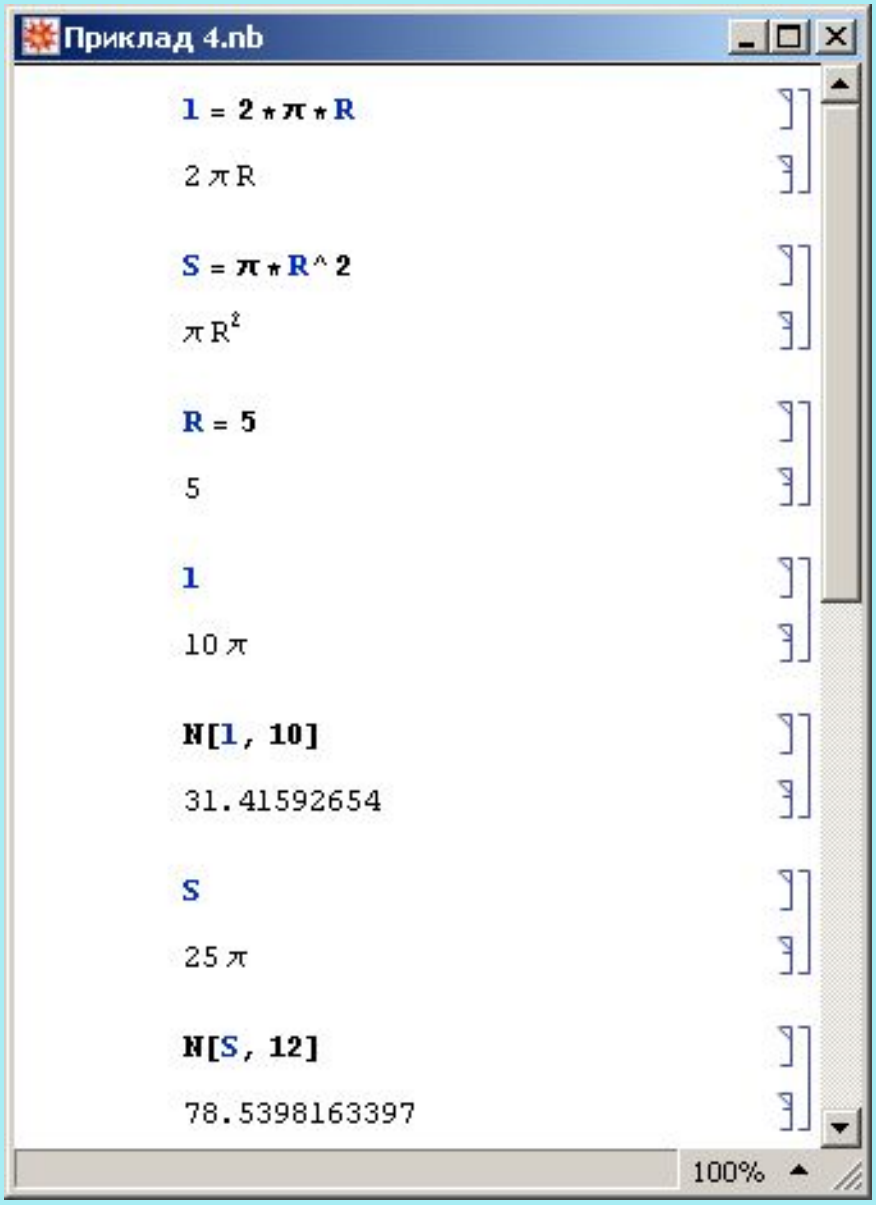
Expand[ ]

Команда для згортки виразу:

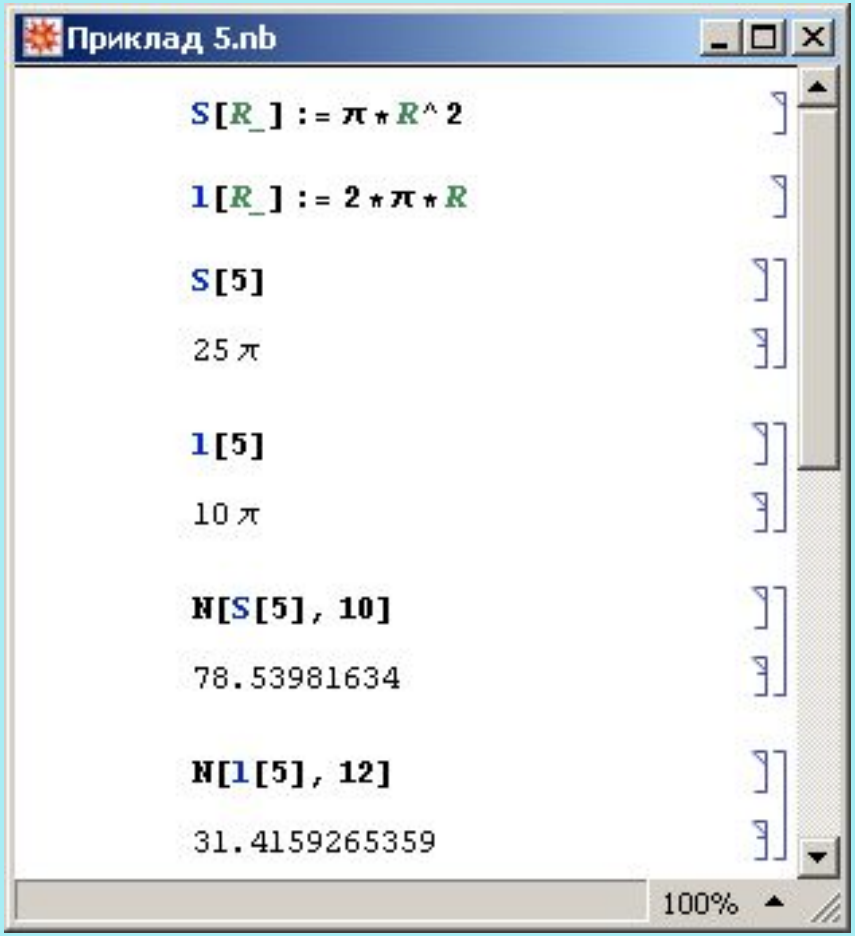
Factor[ ]

Приклади розрахунків - змінні та функції

Приклад Приклад 4Приклад



Приклад 5. Функції

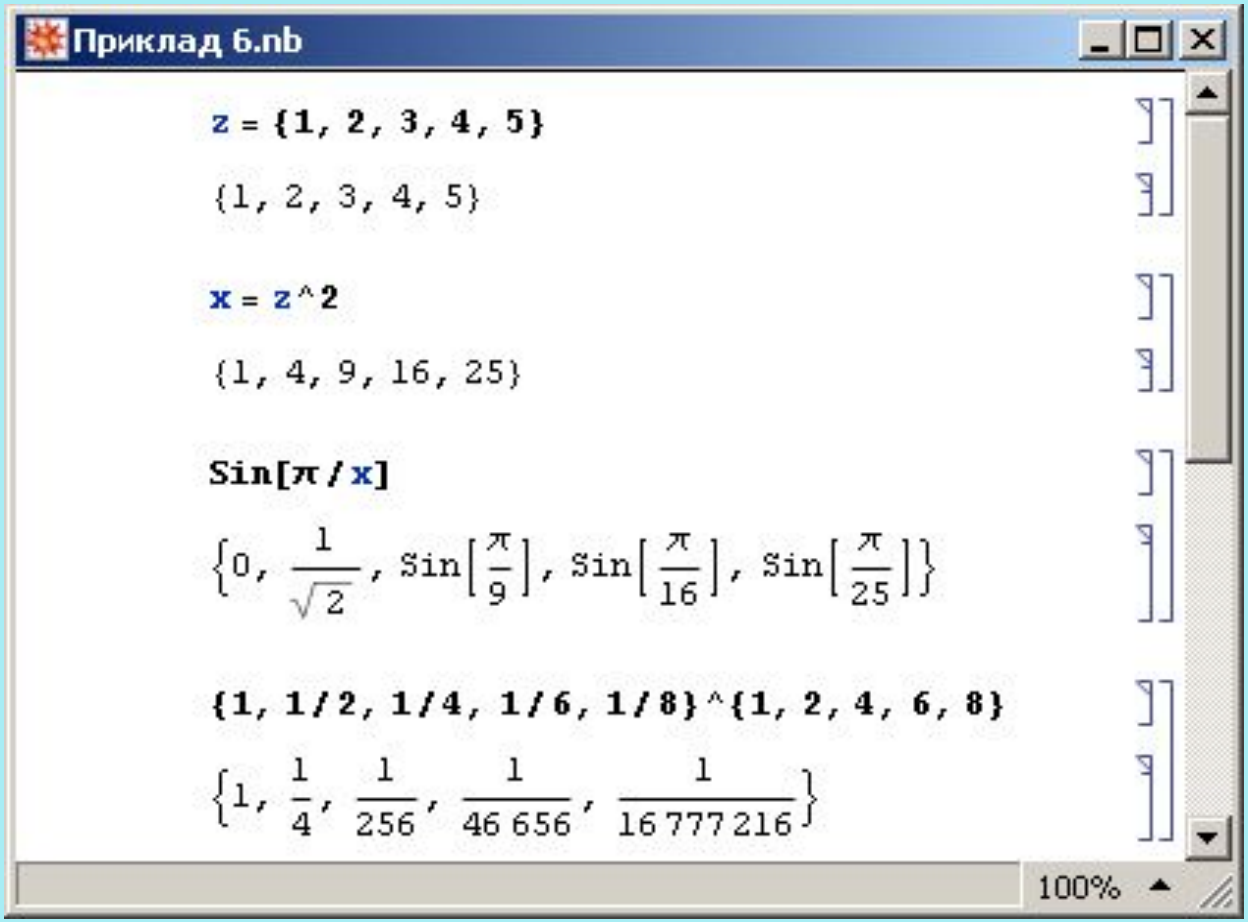


Зверніть увагу на різницю у використанні операторів = та :=

Списки

Список реалізується у вигляді набору елементів (взагалі різного типу), розділених комами і взятих у фігурні дужки. Порядок слідування елементів має значення.

Приклад 6. Списки



Побудова графіків на площині

У відношенні графіки система Mathematica є лідером серед систем комп'ютерної алгебри. Велика кількість опцій дозволяє оформляти графічні образи практично в будь-якому бажаному вигляді.

Графіки в системі Mathematica є об'єктами і тому вони можуть бути значеннями змінних.

Почнемо розгляд графічних можливостей системи з побудови найпростіших графіків функцій однієї змінної виду $y = f(x)$ або просто $f(x)$.

Графік таких функцій будується на площині, тобто в двовимірному просторі. При цьому використовується прямокутна (декартова) система координат. За замовчуванням будуються і лінії координатної системи.

Для побудови двовимірних графіків функцій виду f(x) використовується вбудована в ядро функція **Plot:**

Plot [f, {x, xmin, xmax}] – повертає об'єкт, що представляє собою графік функції *f* аргументу *x* в інтервалі від *xmin* до *xmax*;

Plot [{f1, f2 ,...}, {x, xmin, xmax}] – повертає об'єкт у вигляді графіків ряду функцій *fi*.

Функція **Plot** використовується для побудови однієї або кількох ліній, що дають графічне представлення для зазначених функцій *f, f1, f2* і т. д.

Приклади застосування функції **Plot** показані на рис.5,6.

Зауважимо, що графіки побудовані без використання будь-яких опцій (точніше, з набором опцій за замовчуванням).

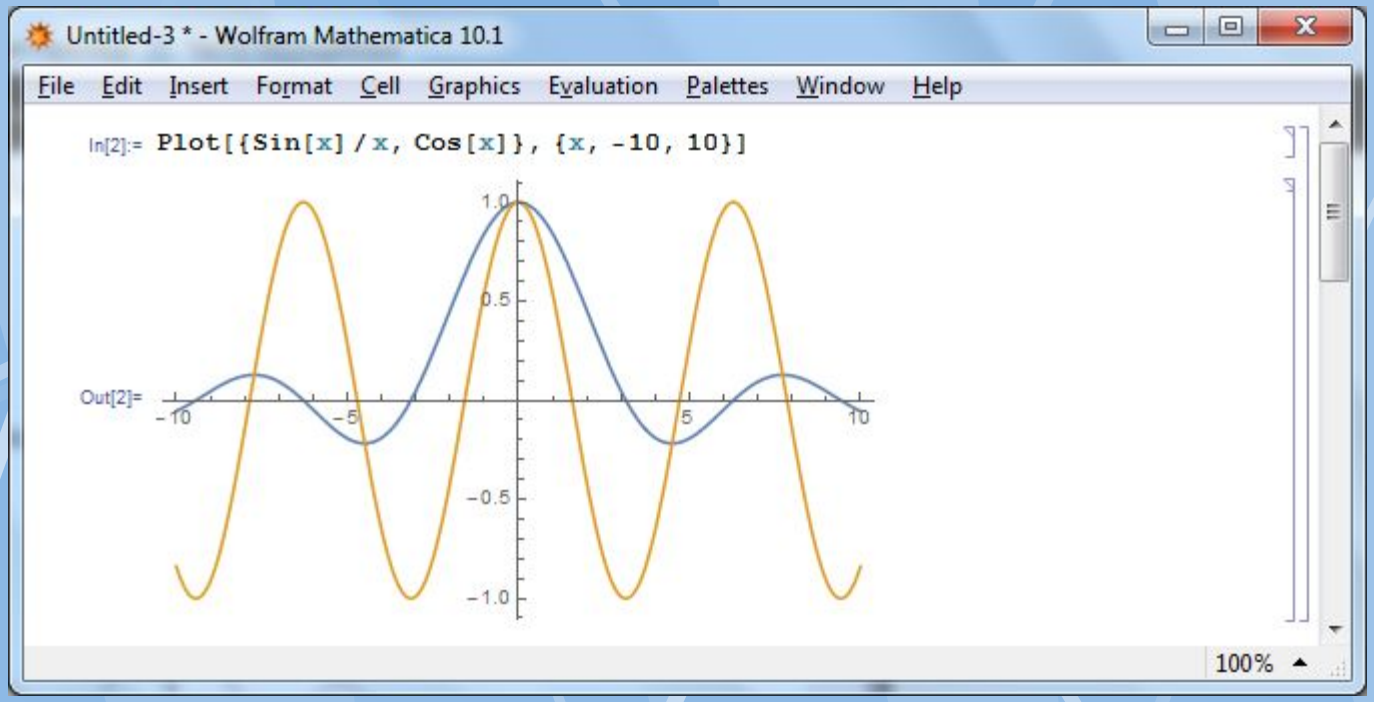

Untitled-3 * - Wolfram Mathematica 10.1
File Edit Insert Format Cell Graphics Evaluation Palettes Window Help
In[2]:= Plot[{Sin[x] / x, Cos[x]}, {x, -10, 10}]
Out[2]=
1.0
0.5
-0.5
-1.0
-10
-5
5
10
100%

Рис.5.

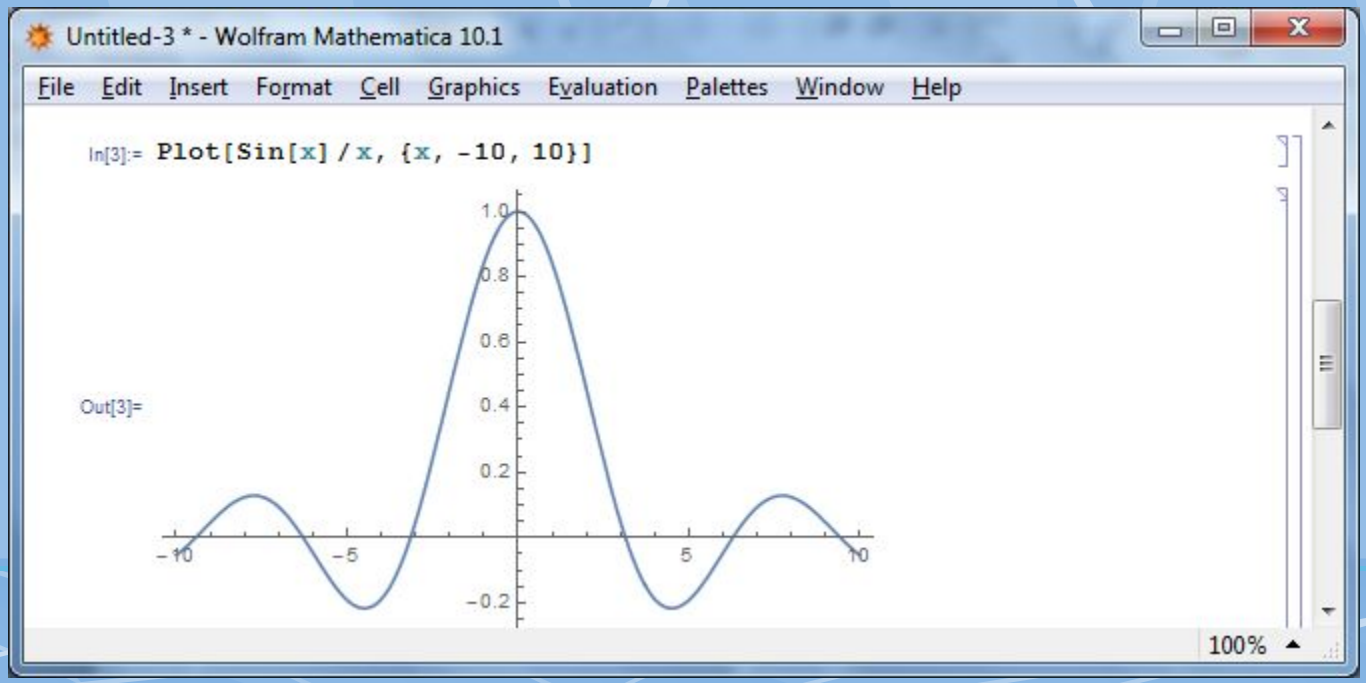

Untitled-3 * - Wolfram Mathematica 10.1
File Edit Insert Format Cell Graphics Evaluation Palettes Window Help
In[3]:= Plot[Sin[x] / x, {x, -10, 10}]
Out[3]=
1.0
0.8
0.6
0.4
0.2
-0.2
-10
-5
5
10
100%

Рис.6.

В міру ускладнення задач користувачеві рано чи пізно перестануть влаштовувати графіки, одержувані при автоматичному виборі їх стилю та інших параметрів.

Для точного налаштування графіків Mathematica використовує спеціальні опції графічних функцій. Для виведення їх списку треба використовувати команду : **Options [Plot].**

Ще одним важливим засобом настроювання графіків є графічні директиви. Синтаксис їх подібний синтаксису функцій. Однак директиви не повертають об'єктів, а лише впливають на їх характеристики.

Застосування графічних директив спільно з опціями дозволяє створювати графіки самого різного виду.

Опції задаються своїм іменем name і значенням value
name -> value

Символьні значення опцій:
Automatic — використовується автоматичний вибір;
None — опція не використовується;
All — використовується в будь-якому випадку;
True — використовується;
False — не використовується.
Багато опцій можуть мати числові значення.

```
Options[Plot]
```

$$\left\{\text{AlignmentPoint} \to \text{Center}, \text{AspectRatio} \to \frac{1}{\text{GoldenRatio}},\right.$$

```
Axes → True, AxesLabel → None, AxesOrigin → Automatic,
AxesStyle → {}, Background → None, BaselinePosition → Automatic,
BaseStyle → {}, ClippingStyle → None, ColorFunction → Automatic,
ColorFunctionScaling → True, ColorOutput → Automatic,
ContentSelectable → Automatic, CoordinatesToolOptions → Automatic,
DisplayFunction :→ $DisplayFunction, Epilog → {}, Evaluated → Automatic,
EvaluationMonitor → None, Exclusions → Automatic,
ExclusionsStyle → None, Filling → None, FillingStyle → Automatic,
FormatType :→ TraditionalForm, Frame → False, FrameLabel → None,
FrameStyle → {}, FrameTicks → Automatic, FrameTicksStyle → {},
GridLines → None, GridLinesStyle → {}, ImageMargins → 0.,
ImagePadding → All, ImageSize → Automatic, ImageSizeRaw → Automatic,
LabelStyle → {}, MaxRecursion → Automatic, Mesh → None,
MeshFunctions → {#1 &}, MeshShading → None, MeshStyle → Automatic,
Method → Automatic, PerformanceGoal :→ $PerformanceGoal,
PlotLabel → None, PlotLegends → None, PlotPoints → Automatic,
PlotRange → {Full, Automatic}, PlotRangeClipping → True,
PlotRangePadding → Automatic, PlotRegion → Automatic,
PlotStyle → Automatic, PlotTheme :→ $PlotTheme,
PreserveImageOptions → Automatic, Prolog → {}, RegionFunction → (True &),
RotateLabel → True, TargetUnits → Automatic, Ticks → Automatic,
TicksStyle → {}, WorkingPrecision → MachinePrecision}
```

```
{СоотношениеСторон = AspectRatio, ЗолотоеСечение = GoldenRatio,
  Оси = Axes, Да = True, ПодписиОсей = AxesLabel, Никакой = None,
  ПересечениеОсей = AxesOrigin, Авто = Automatic,
  СтильОсей = AxesStyle, Фон = Background,
  СтильОбрезки = ClippingStyle, ЦветоваяФункция = ColorFunction,
  МасштабированиеЦветовойФункции = ColorFunctionScaling,
  Эпилог = Epilog, Исключения = Exclusions,
  СтильИсключений = ExclusionsStyle, Заливка = Filling,
  СтильЗаливки = FillingStyle, Рамка = Frame, Нет = False,
  ПодписиРамки = FrameLabel, СтильРамки = FrameStyle,
  ЗасечкиРамки = FrameTicks, СтильЗасечекРамки = FrameTicksStyle,
  ЛинииСетки = GridLines, СтильЛинийСетки = GridLinesStyle,
  РазмерИзображения = ImageSize, СтильПодписи = LabelStyle,
  Сетка = Mesh, ФункцииСетки = MeshFunctions,
  ЗатенениеСетки = MeshShading, СтильСетки = MeshStyle,
  ПодписьГрафика = PlotLabel, ЛегендыГрафика = PlotLegends,
  КоличествоТочек = PlotPoints, ДиапазонОтображения = PlotRange,
  Полностью = Full, СтильГрафика = PlotStyle, Пролог = Prolog,
  ФункцияОбласти = RegionFunction, ВращатьПодпись = RotateLabel,
  Засечки = Ticks, СтильЗасечек = TicksStyle,
  РабочаяОтносительнаяТочность = WorkingPrecision,
  МашиннаяТочность = MachinePrecision};
```

Опції функції Plot[]

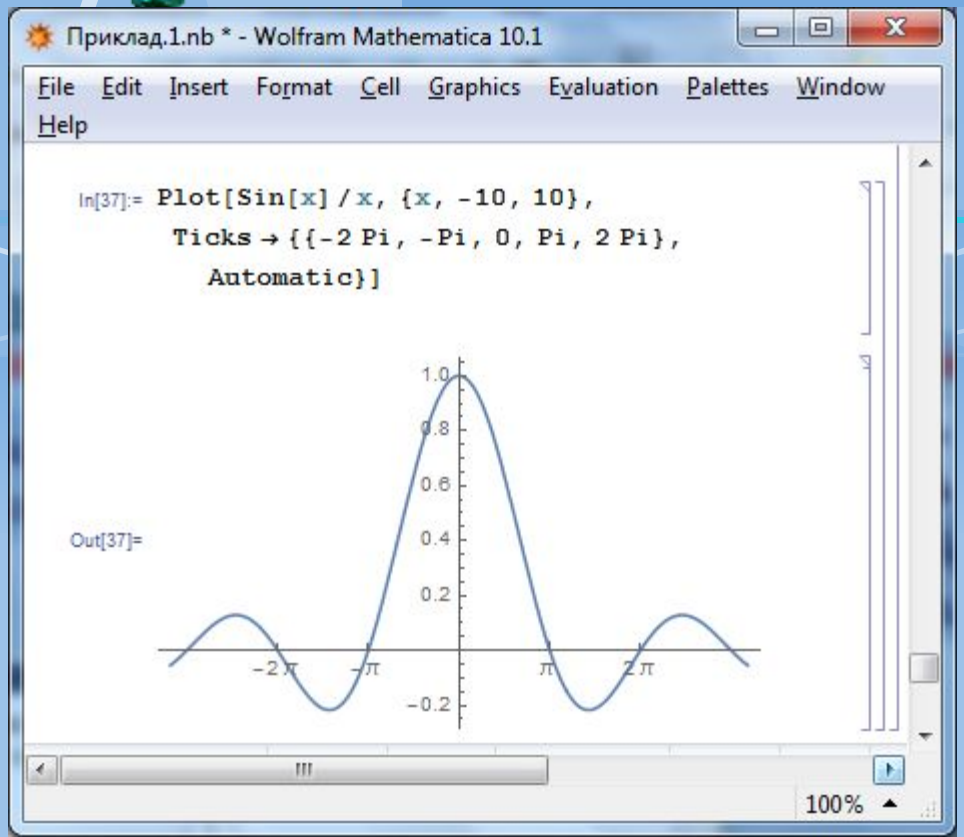
♦ Приклад♦ Приклад 1♦ Приклад 1:

```
Plot[Sin[x]/x,{x,-10,10},Ticks->{{-2 Pi,-Pi,0,Pi,2 Pi},Automatic}]
```

мітки шкали ("рисочки")

координати "рисочок"

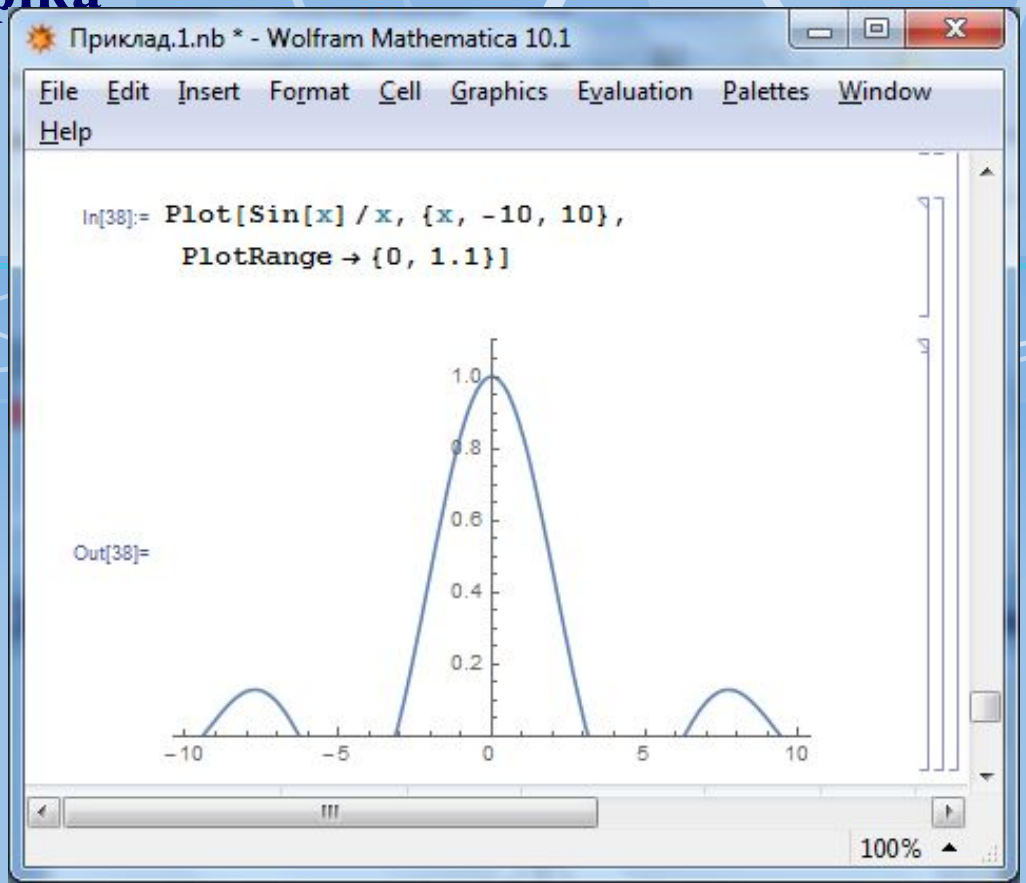
режим відображення "рисочок" для вісі Y



♦ Приклад♦ Приклад 2♦ Приклад 2:

Plot[Sin[x]/x,{x,-10,10}, PlotRange->{0,1.1}]

Діапазон (по вісі Y) для відображення графіка



♦ Приклад♦ Приклад 3♦ Приклад 3:

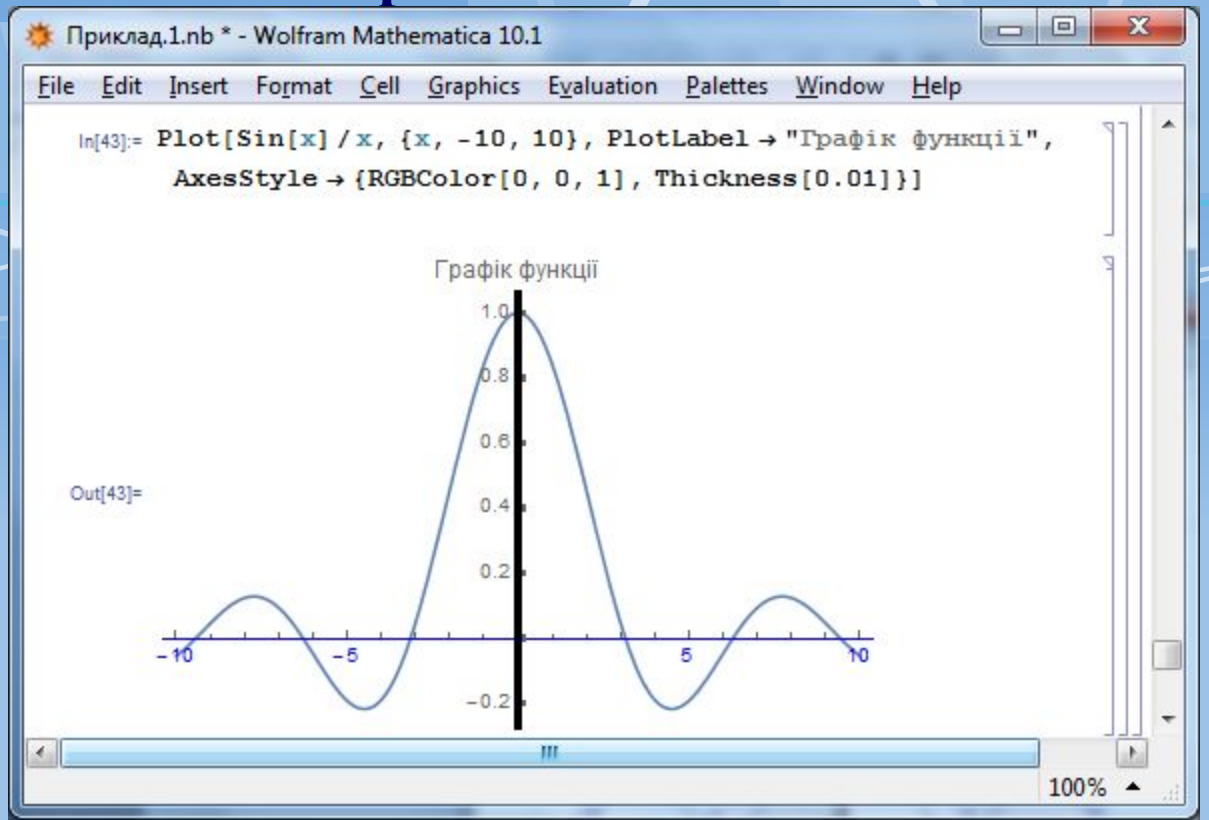
```
Plot[Sin[x]/x,{x,-10,10}, PlotLabel -> "Графік функції",
 AxesStyle -> {RGBColor[0, 0, 1],Thickness[0.01]}]
```

Стиль координатних осей:

вісь X - колір

товщина

Заголовок для графіка

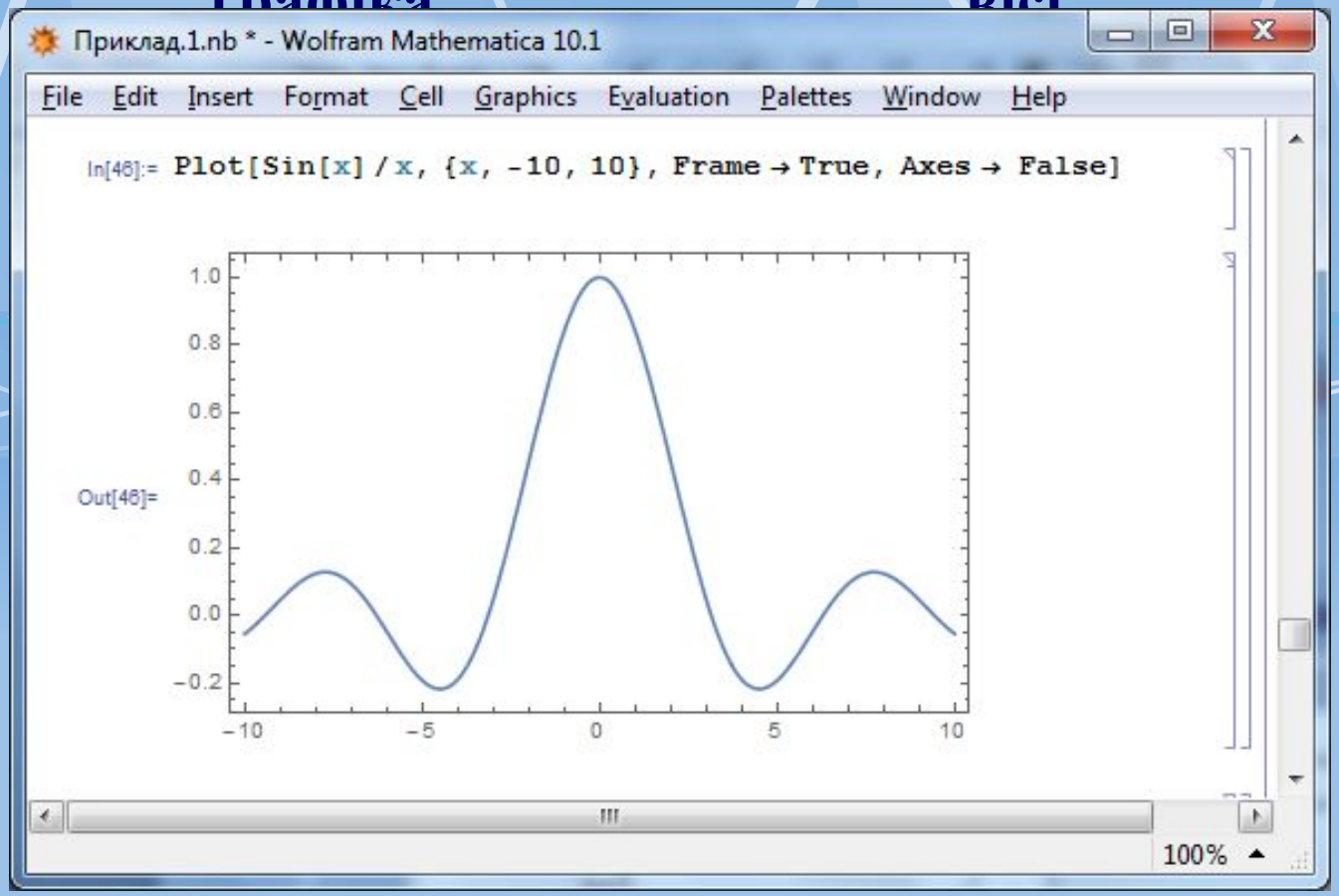


♦ Приклад♦ Приклад 4♦ Приклад 4:

Plot[Sin[x]/x,{x,-10,10}, Frame -> True, Axes -> False]

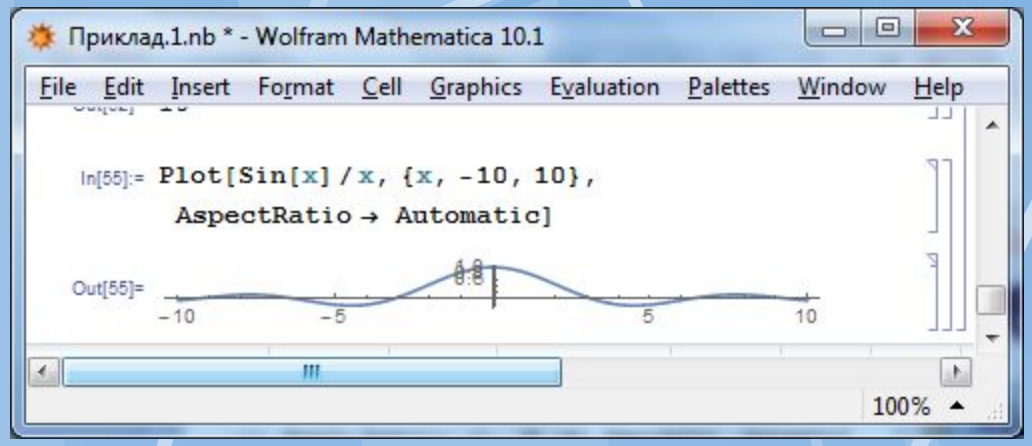
Рамка навколо графіка

Координатні вісі

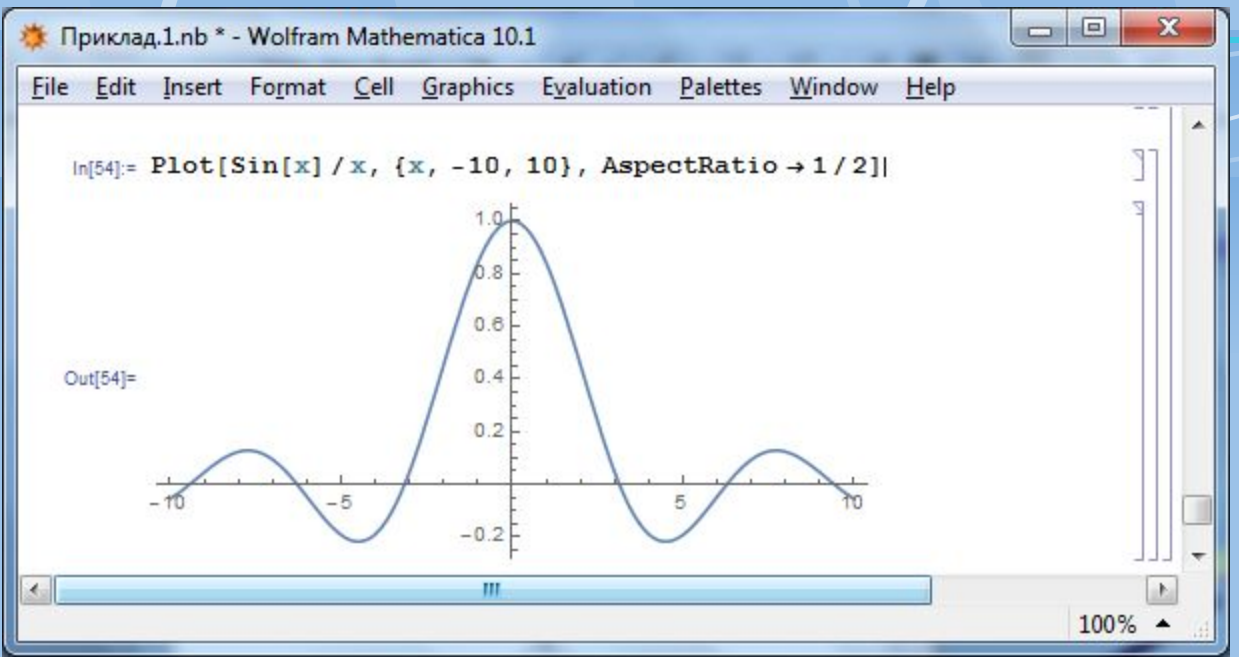


♦ **Приклад 5:**

Plot[Sin[x]/x,{x,-10,10}, AspectRatio -> Automatic]



Масштаб (співвідношення висота/ширина графіка)



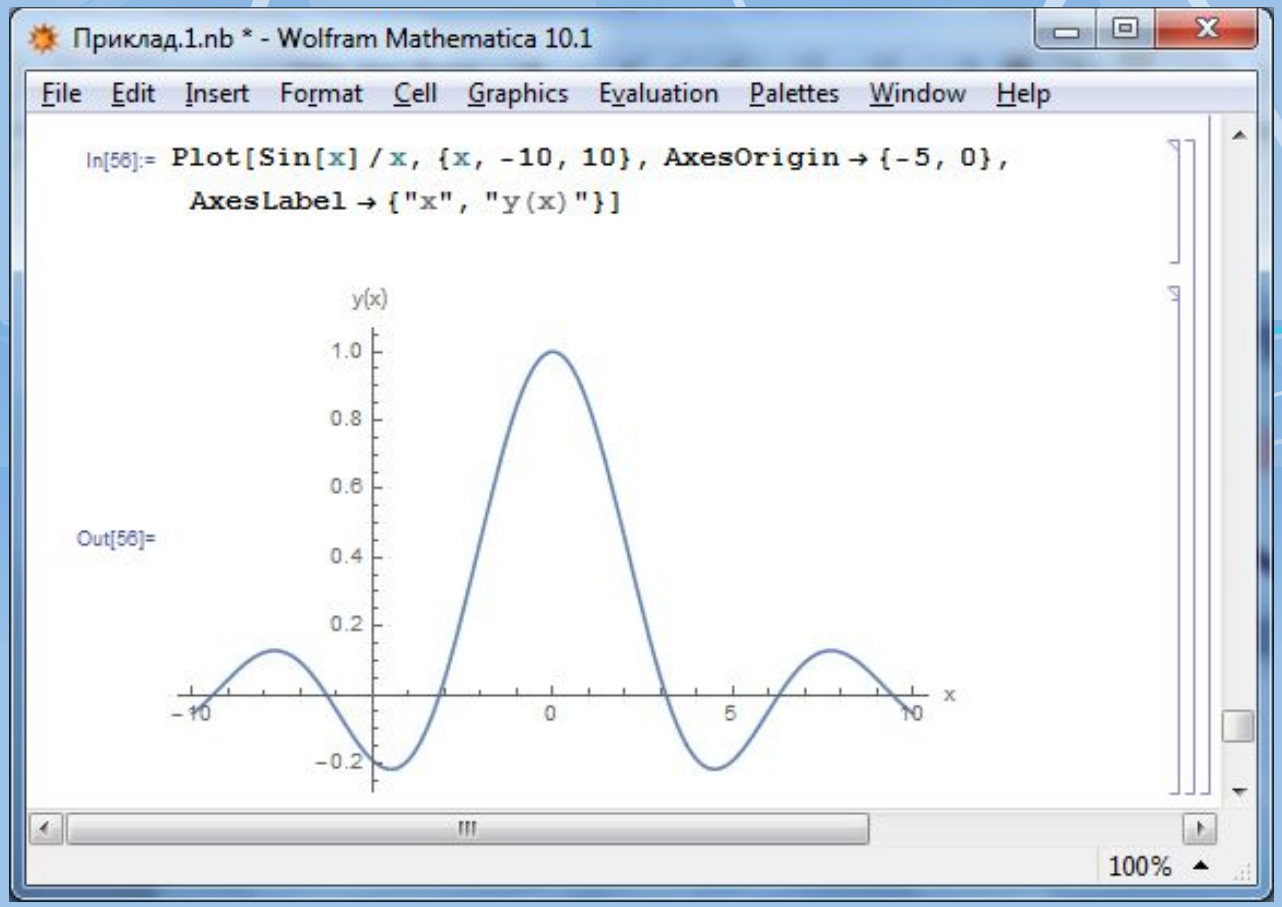
AspectRatio -> 1/2

♦ Приклад♦ Приклад 6♦ Приклад 6

```
Plot[Sin[x]/x,{x,-10,10},
 AxesOrigin -> {-5, 0}, AxesLabel -> {"x", "y(x)"}]
```

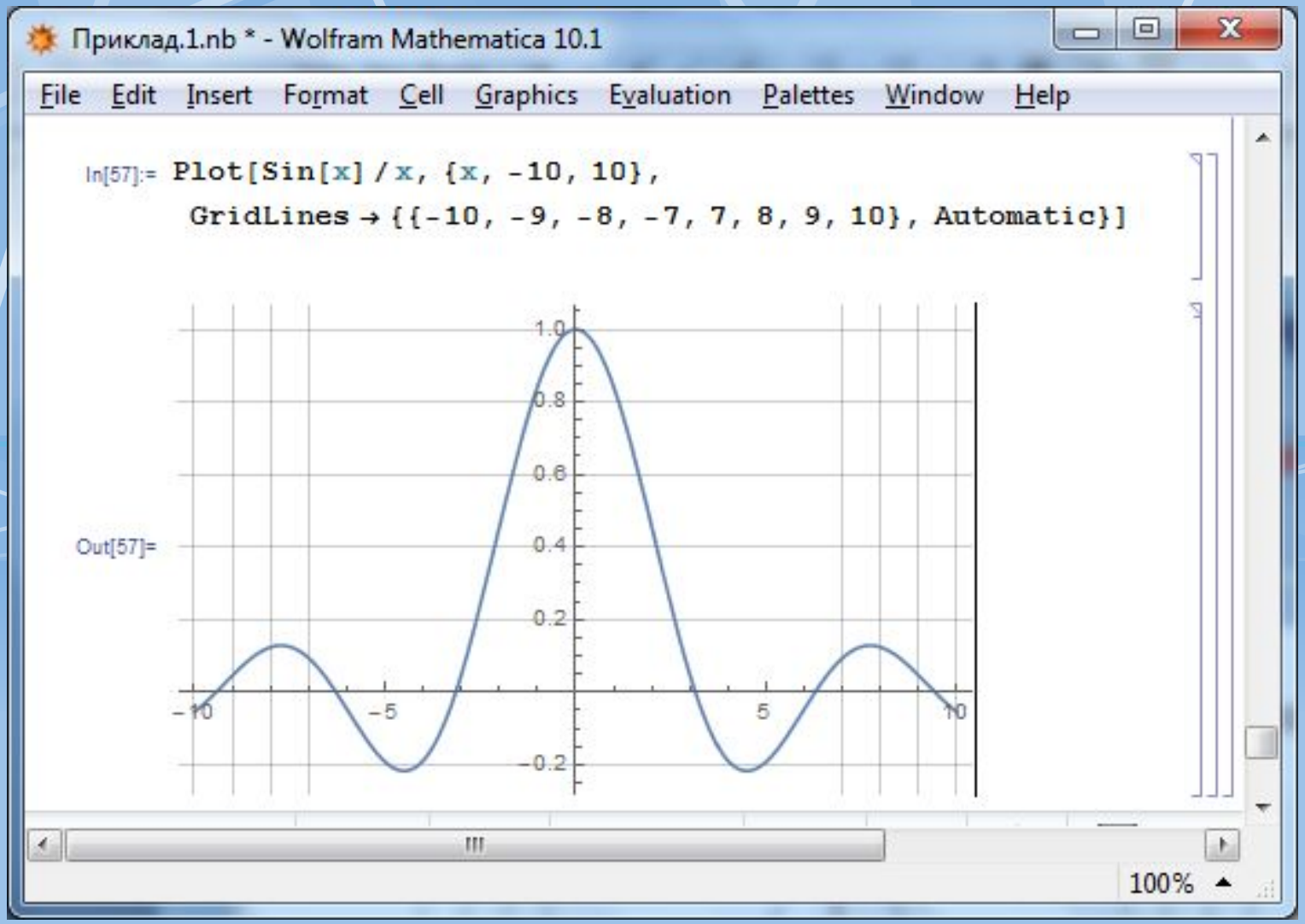
Точка перетину осей

Підписи для осей



♦ Приклад♦ Приклад 7♦ Приклад 7 **Лінії координатної сітки**

Plot[Sin[x]/x,{x,-10,10},
GridLines -> {{-10, -9, -8, -7, 7, 8, 9, 10}, Automatic}]



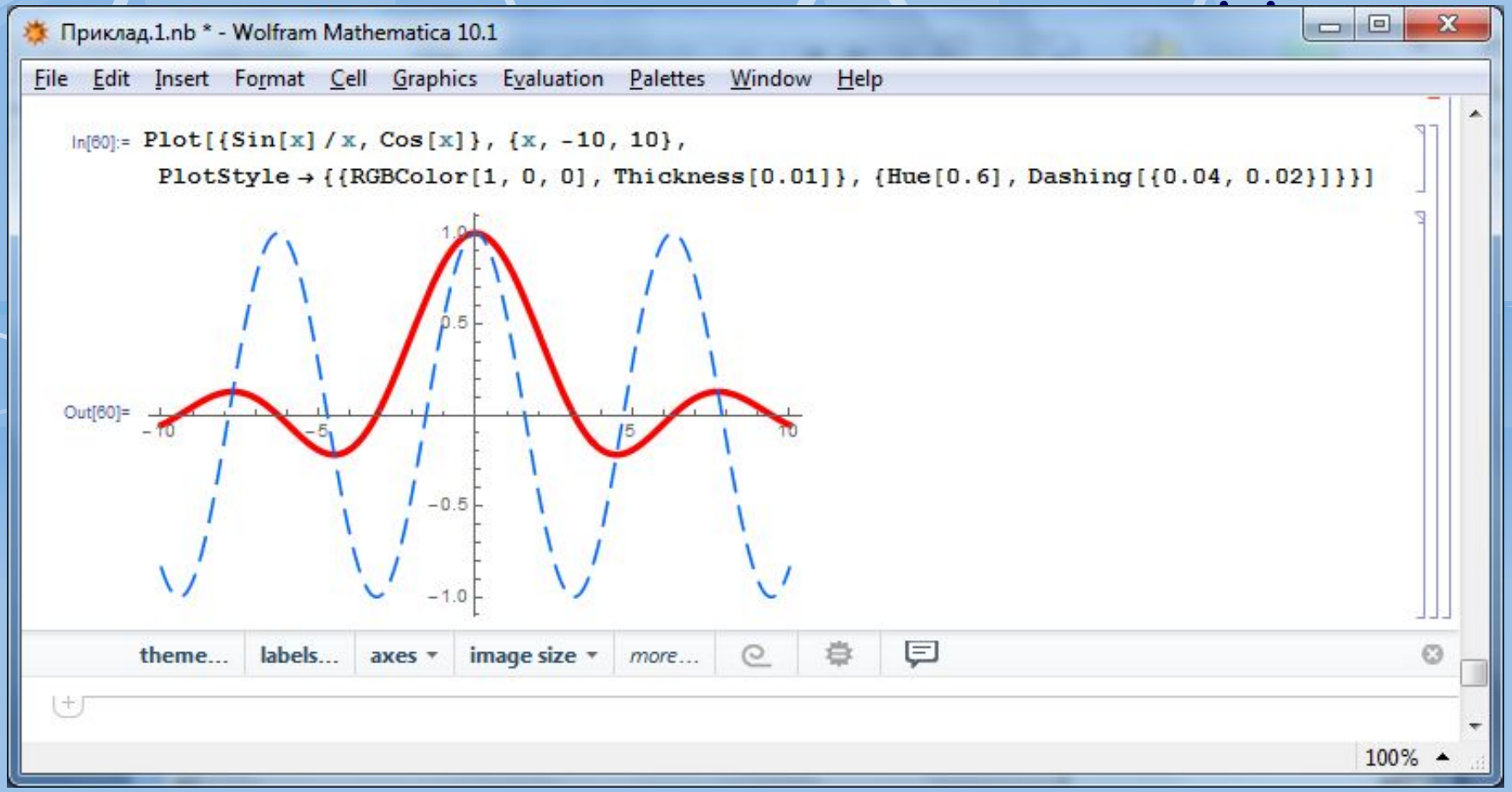
♦ Приклад♦ Приклад 8♦ Приклад 8:

```
Plot[{Sin[x]/x, Cos[x]},{x,-10,10},
 PlotStyle -> {{RGBColor[1, 0, 0], Thickness[0.01]},
{Hue[0.6],    Dashing[{0.04, 0.02}]}}]
```

Стиль ліній

Перша лінія

Друга

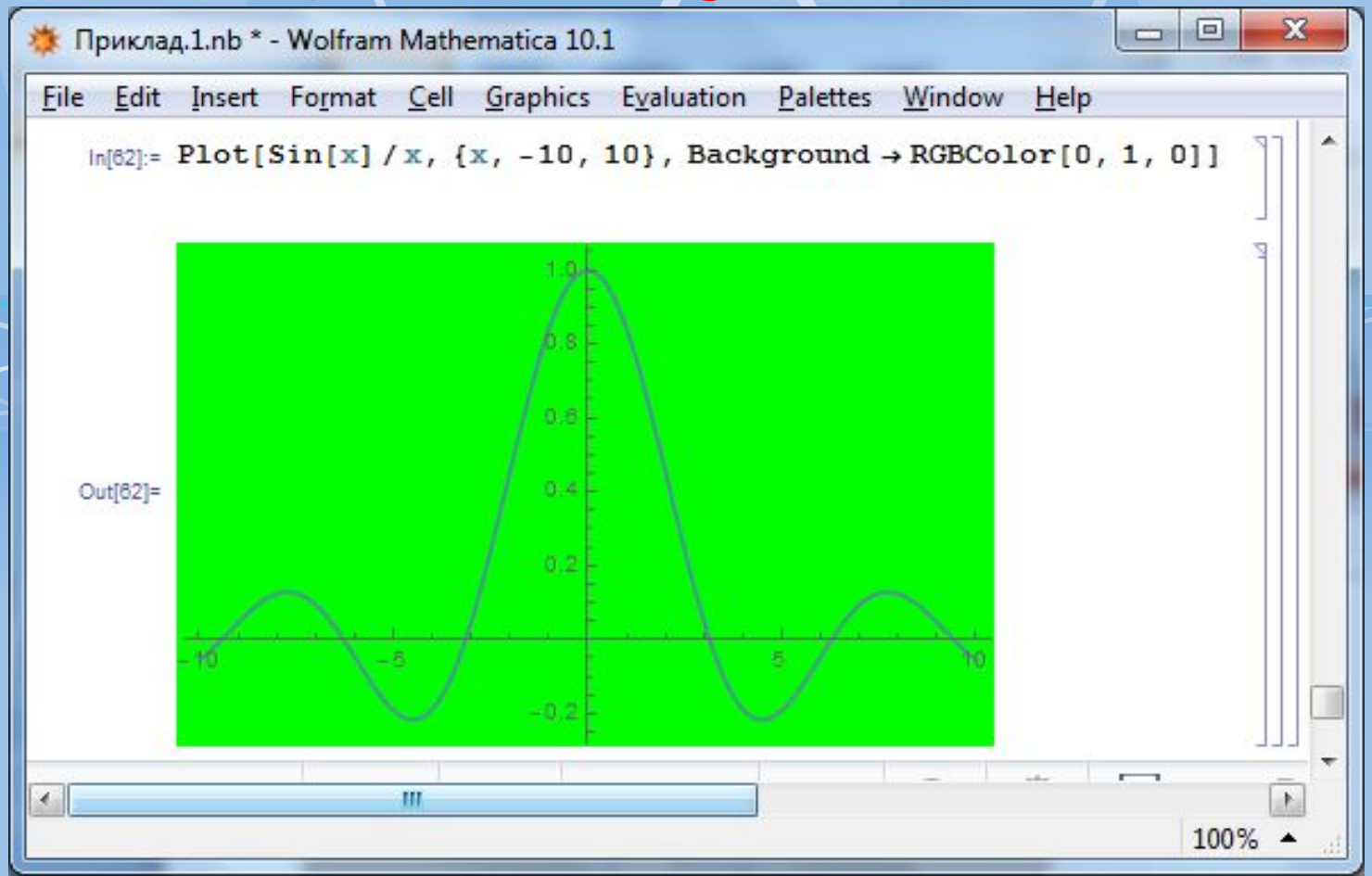


♦ **Приклади:**

Plot[Sin[x]/x,{x,-10,10}, Background -> GrayLevel[0.7]]

Колір фона

Plot[Sin[x]/x,{x,-10,10}, Background -> RGBColor[0, 1, 0]]



Також часто виникає необхідність побудови графіка по точках. Це забезпечує вбудована в ядро графічна функція **ListPlot:**

• **ListPlot [{yl, y2 ,...}]** – виводить графік списку величин.
Координати *x* приймають значення 1, 2, ...;

• **ListPlot [{{x1, y1}, {x2, y2 },...}]**– виводить графік списку величин з зазначеними x і y координатами.

У найпростішому випадку (рис. 7) ця функція сама задає значення координати x = 0, 1, 2, 3, ...і будує на графіку точки з координатами *(x,y)*, вибираючи у послідовно зі списку координат.

Функція **ListPlot**, особливо в її другій формі (із заданими координатами *x* і *y*), зручна для виведення на графік експериментальних точок.

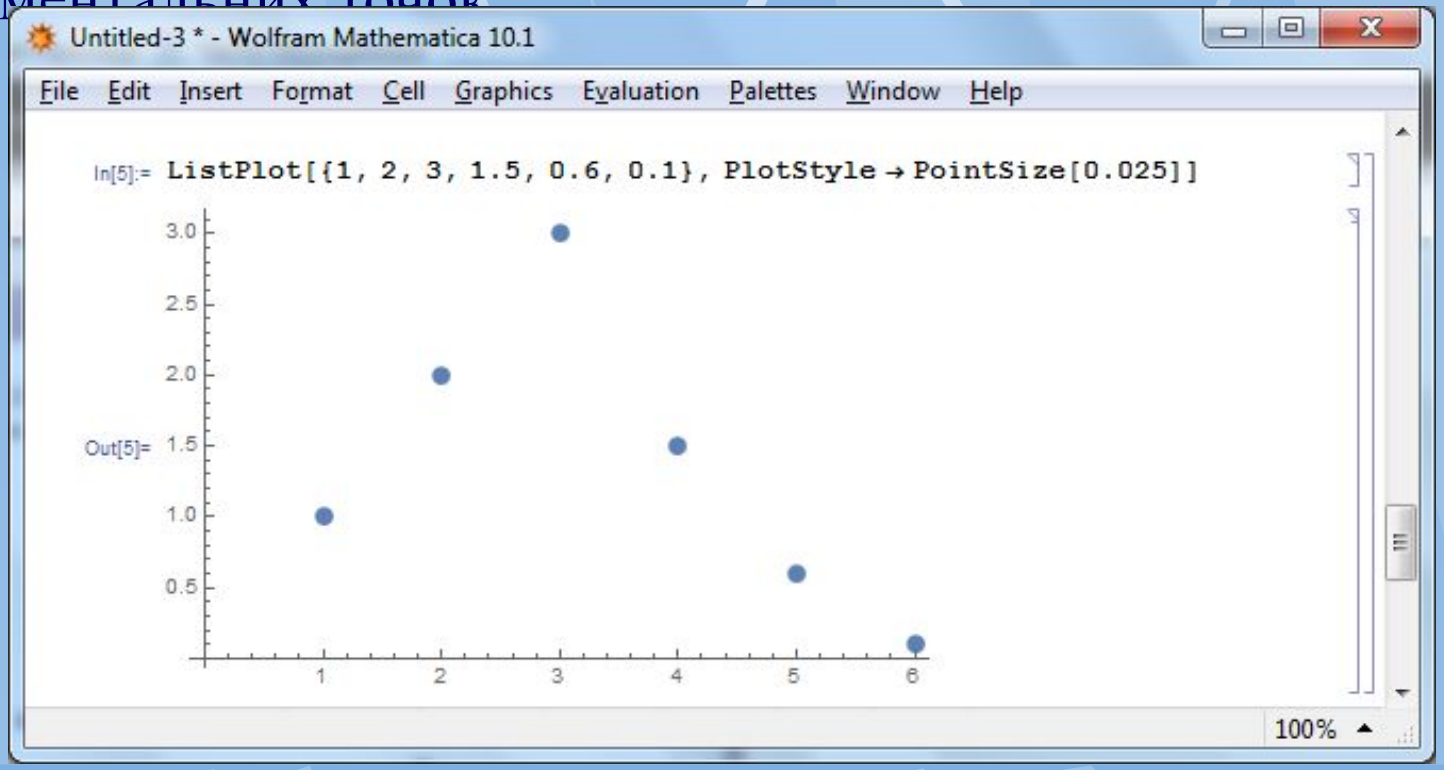


Рис.7. Приклад Побудови графіка по точках

Для побудови параметрично заданих функцій використовуються наступні графічні засоби:

- **ParametricPlot [{fx, fy}, {t, tmin, tmax}]** – будує параметричний графік з координатами *fx* і *fy* (відповідними *x* і *y*), одержуваними як функції від *t*;
- **ParametricPlot [{{fx, fy}, {gx, gy },...}, {t, tmin, tmax}]** – будує графіки декількох параметричних кривих.

Функції *fx, fy* можуть бути як безпосередньо вписані в список параметрів, так і визначені як функції користувача.

На рис. 8 показано побудову параметрично заданої фігури Ліссажу.

Вона задається функціями синуса і косинуса з постійним параметром R і аргументами, кратними t.

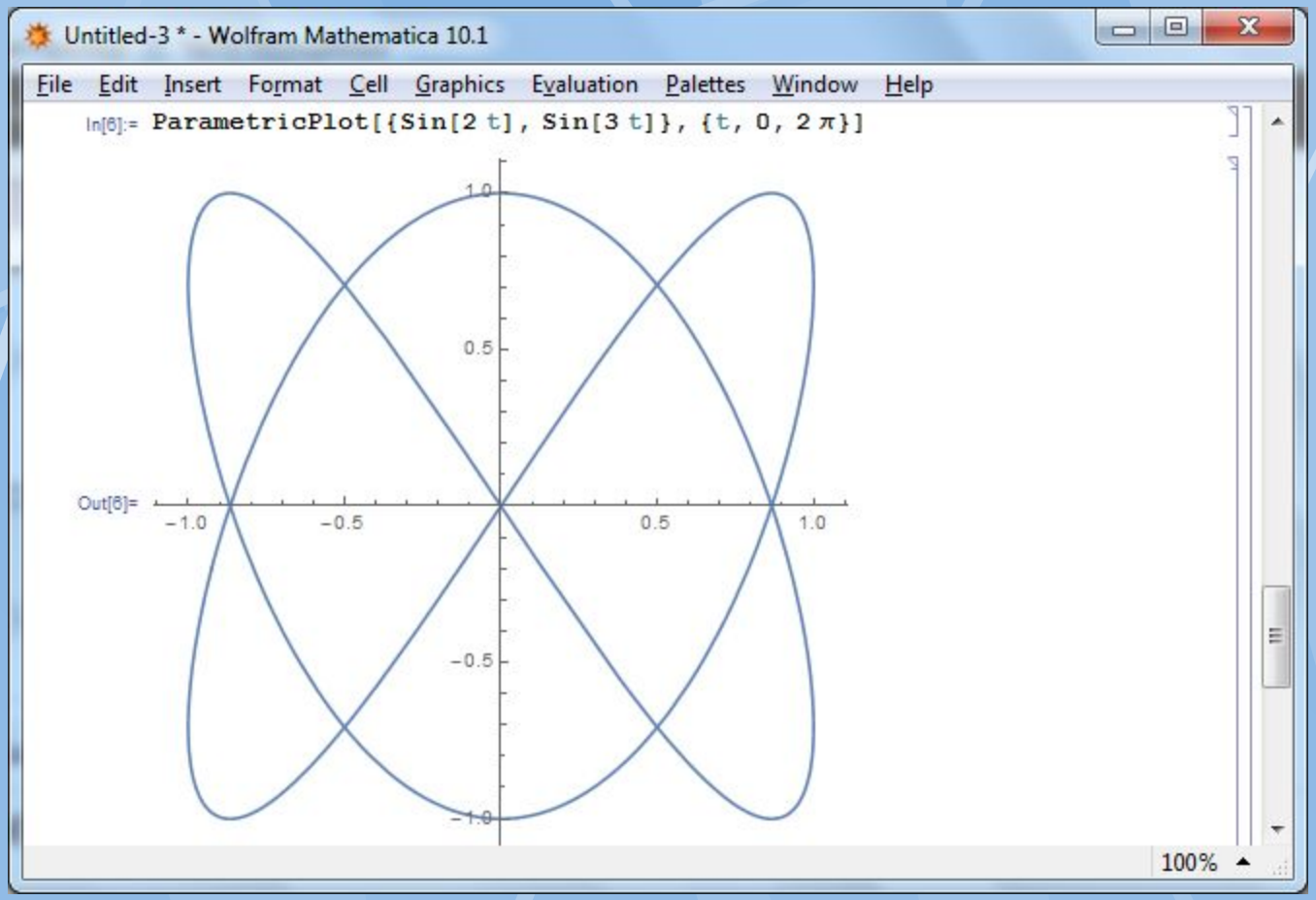


Рис.8. Побудова фігури Ліссажу

Тепер розглянемо спосіб побудови графіків в полярній системі координат (рис. 9).

Для цього використовується функція **PolarPlot:**

- **PolarPlot [f, {t, tmin, tmax}]** – будує графік в полярній системі координат.

- **PolarPlot [{f1, f2, f3, ...}, {t, tmin, tmax}]** – будує графіки функцій в полярній системі координат.

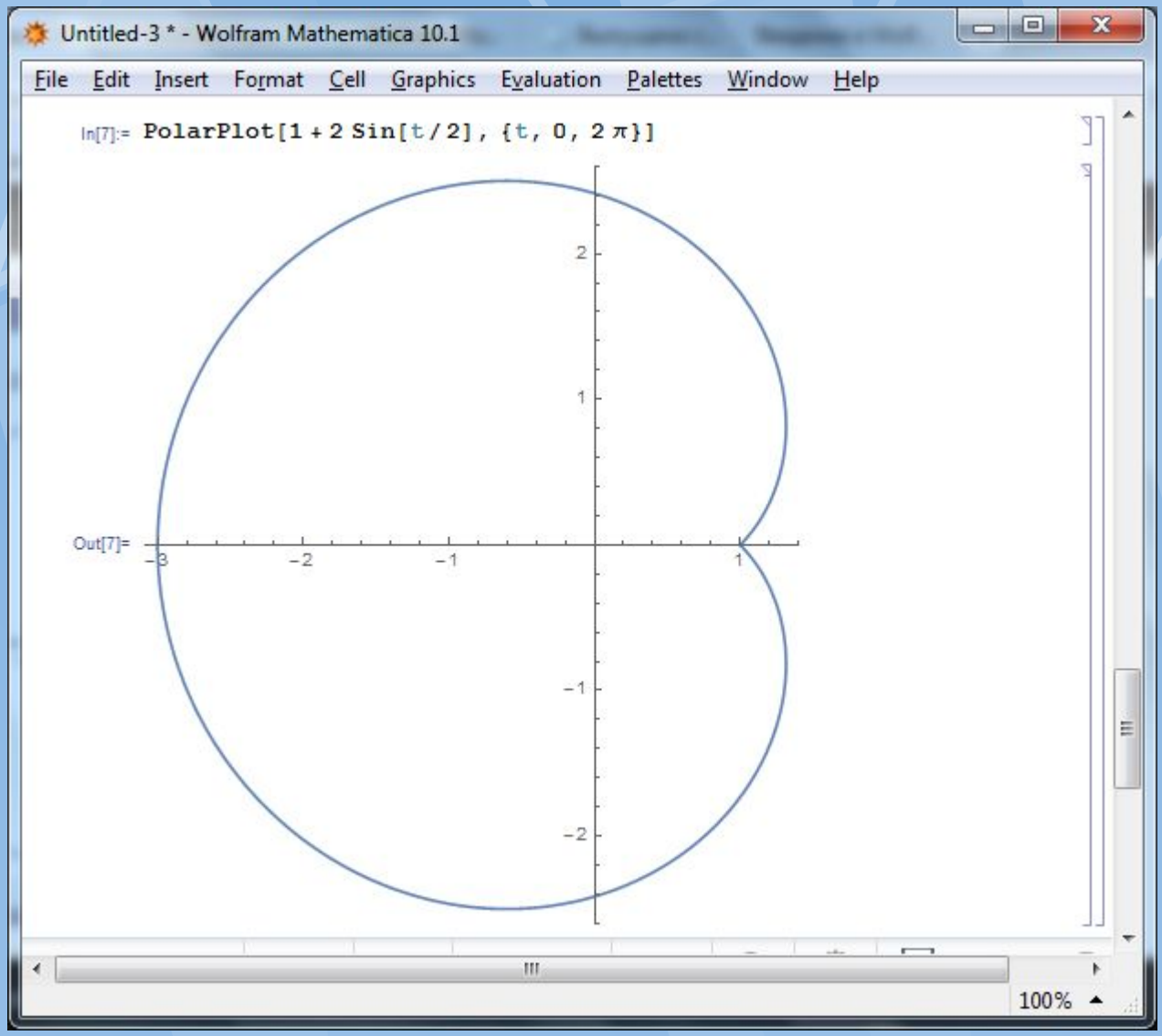


Рис.9.Приклад побудови графіка функції в полярній системі координат

Побудова графіків поверхонь

Функція двох змінних $z = f(x, y)$ утворює в просторі деяку тривимірну поверхню або фігуру. Для їх побудови доводиться використовувати координатну систему з трьома осями координат: *x*, *y* і *z*. Оскільки екран дисплея плоский, то насправді об'ємність фігур лише імітується.

Для побудови графіків тривимірних поверхонь в системі Mathematica використовується основна графічна функція **Plot3D:**

- **Plot3D [f, {x, xmin, xmax}, {y, ymin, ymax}]** – будує тривимірний графік функції f (x, y);
- **Plot3D [{f, s}, {x, xmin, xmax}, {y, ymin, ymax}]** – будує тривимірний графік, в якому висоту поверхні визначає параметр *f*, а затінення - параметр *s*.

На рис.11 показаний приклад побудови поверхні, що описується функцією двох змінних *cos (xy)* при *x* і *y*, що міняються від -3 до 3.

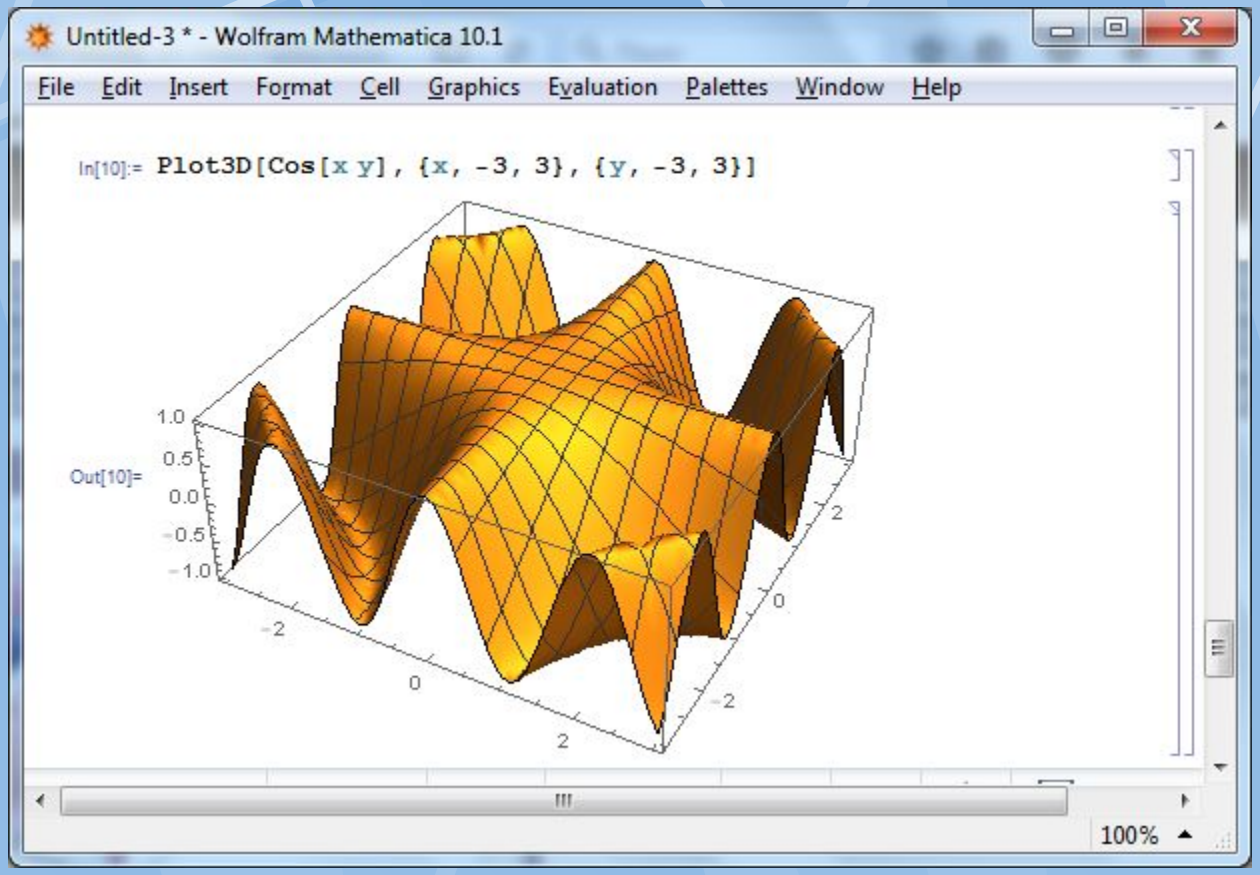


Рис.11. Приклад побудови поверхні

Поверхні також можна будувати по точкам та в параметричній формі використовуючи при цьому відповідні функції **ListPointPlot3D** і **ParametricPlot3D.**

3. Елементи програмування

3.1. Основні арифметичні оператори

2.3. Логічні оператори

3. 3. Оператори порівняння

3.4. Умовні оператори

3.5. Оператори циклу

3.6. Створення модулів

3.7. Приклад

Арифметичні оператори (1)

= Присвоєння

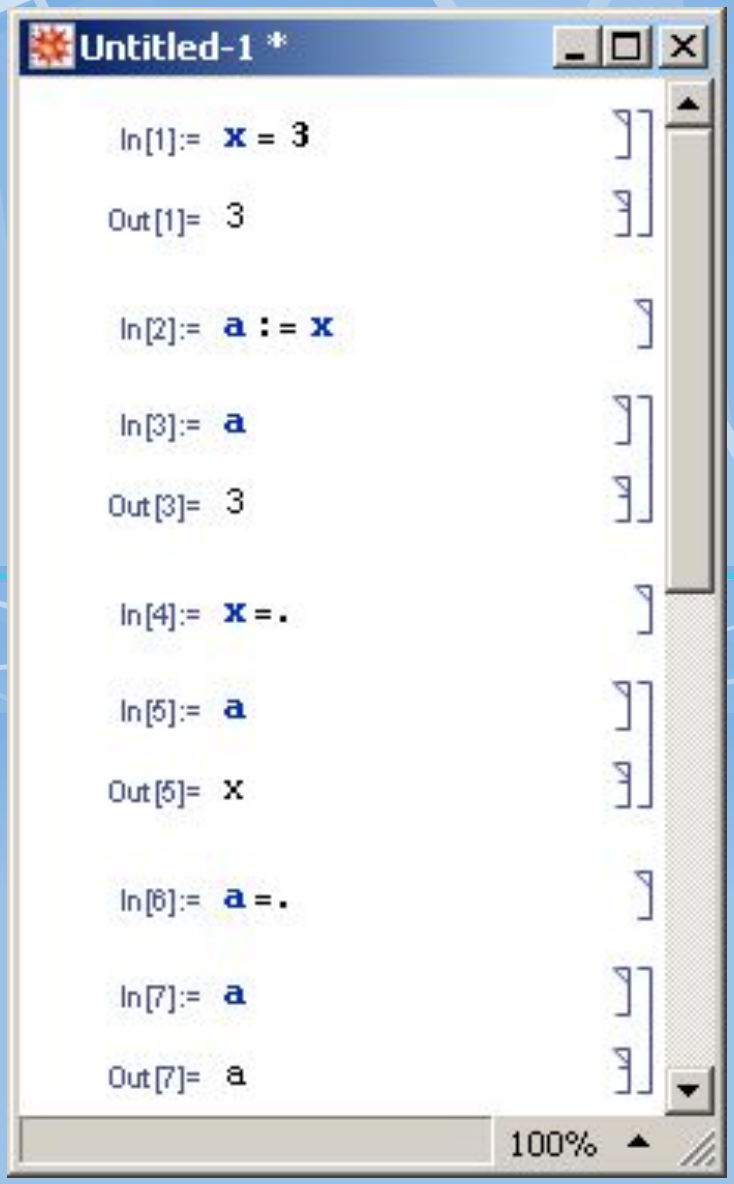
```
In[8]:= x = 3
Out[8]= 3

In[9]:= b = x
Out[9]= 3

In[10]:= b
Out[10]= 3

In[11]:= x = 5
Out[11]= 5

In[12]:= b
Out[12]= 3
```

:= Функціональне присвоєння

```
In[13]:= x = 3
Out[13]= 3

In[14]:= b := x

In[15]:= b
Out[15]= 3

In[16]:= x = 5
Out[16]= 5

In[17]:= b
Out[17]= 5
```

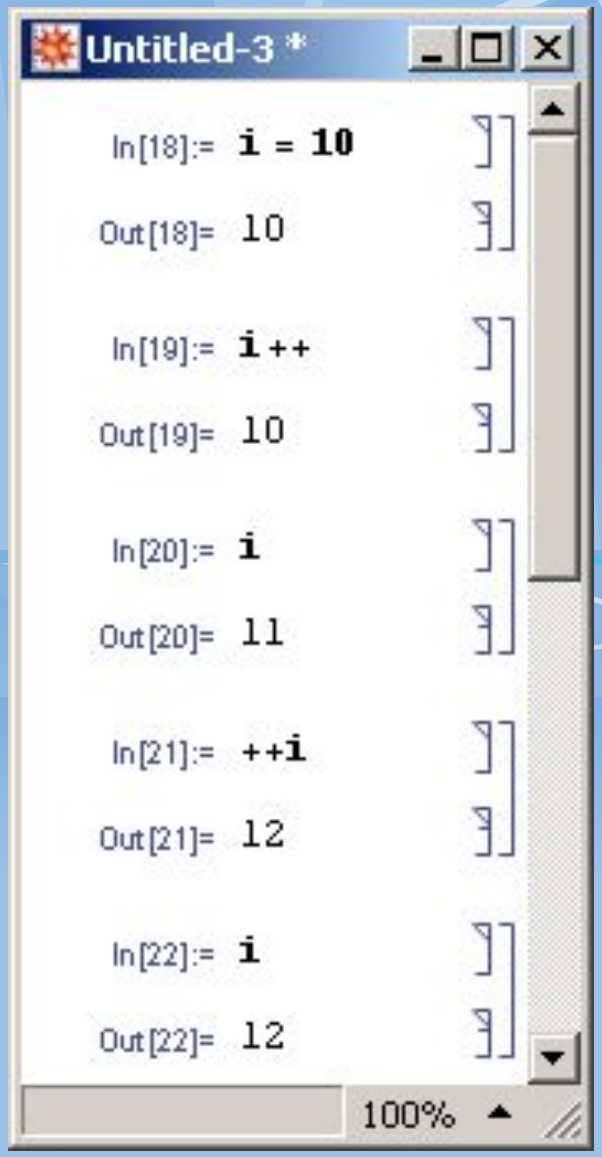
Арифметичні оператори (2)

=. Оператор відмови від значення

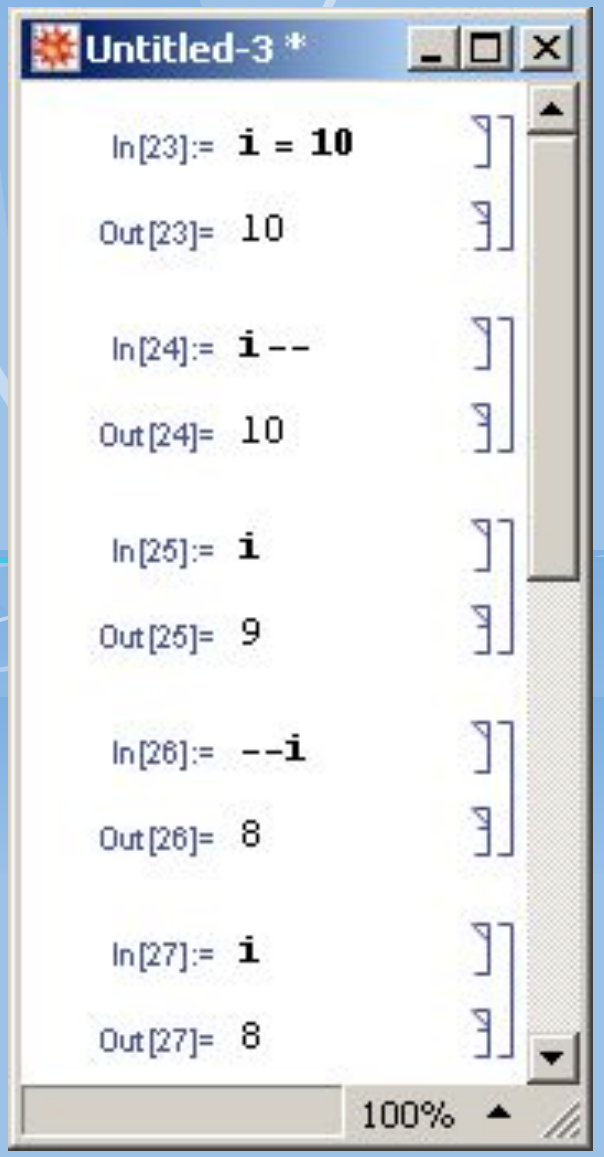


Арифметичні оператори (3)

++ Оператор інкременту



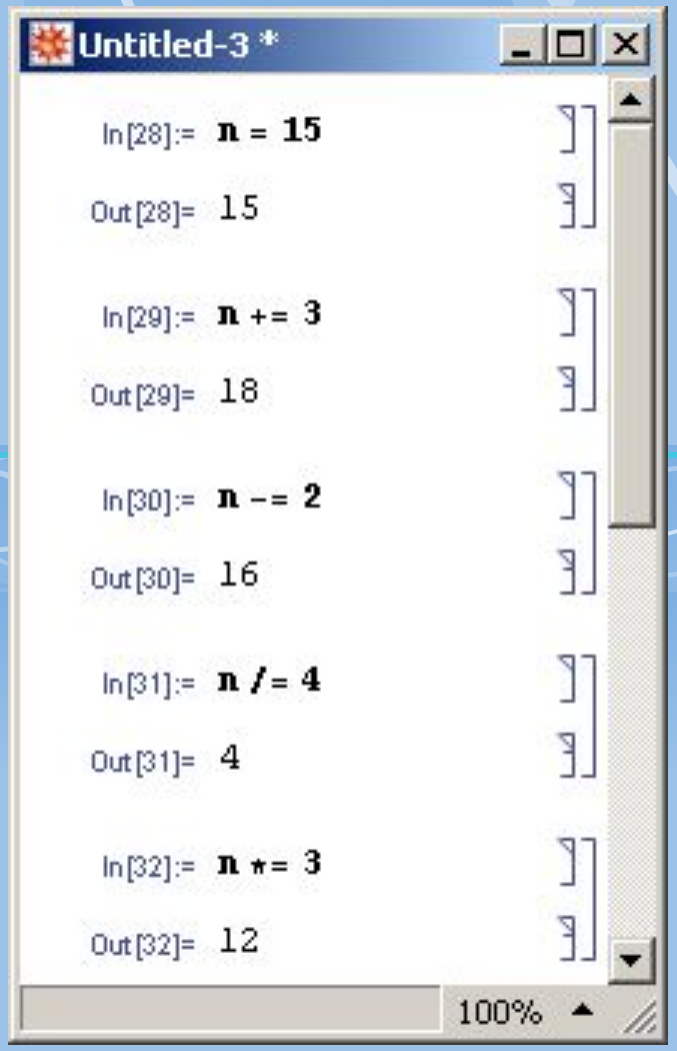
-- Оператор декременту



Арифметичні оператори (4)

Скорочені форми операторів:

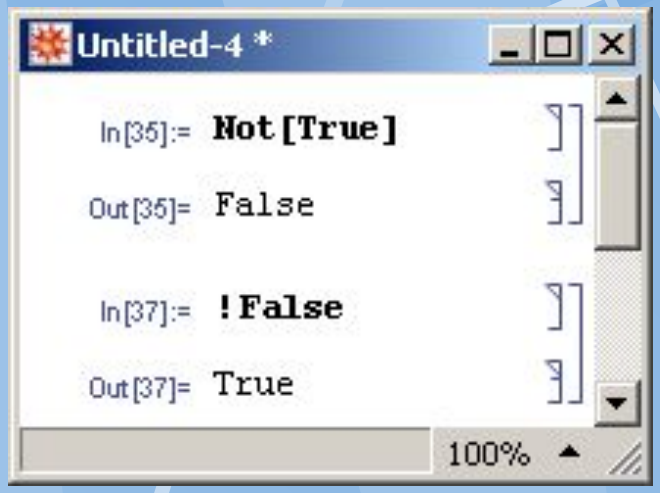
+= додавання **-=** віднімання ***=** множення **/=** ділення



Логічні оператори (1)

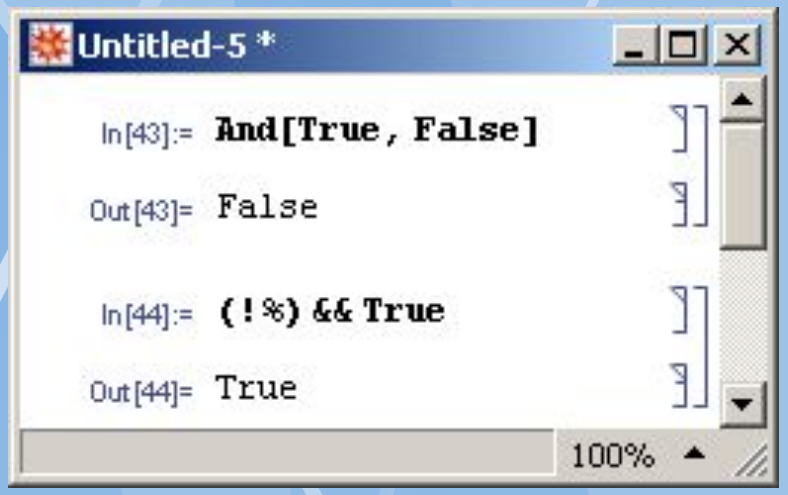
Логічне заперечення:

Not[] або !



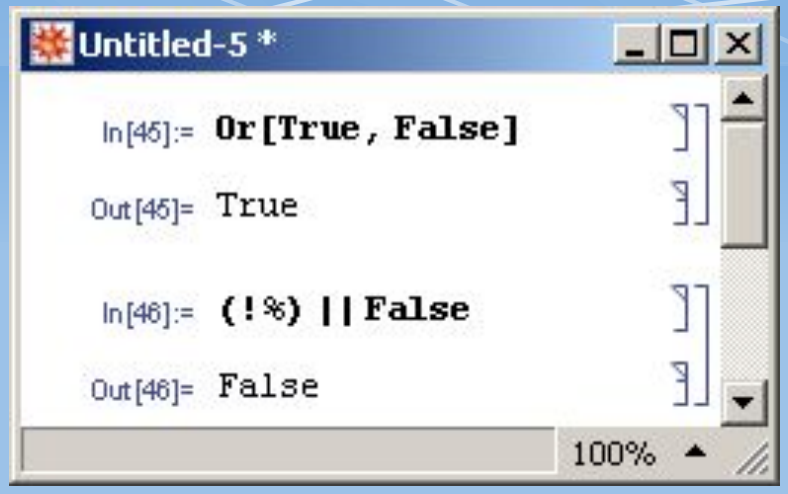
Логічне AND:

And[] або &&



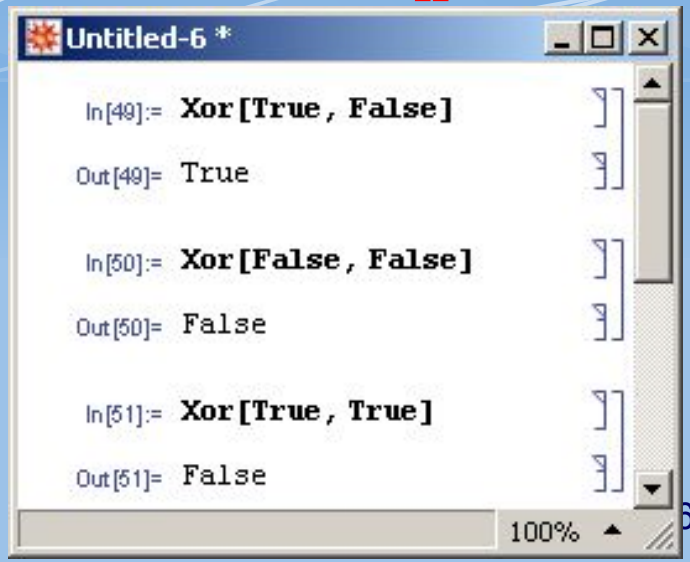
Логічне OR:

Or[] або ||



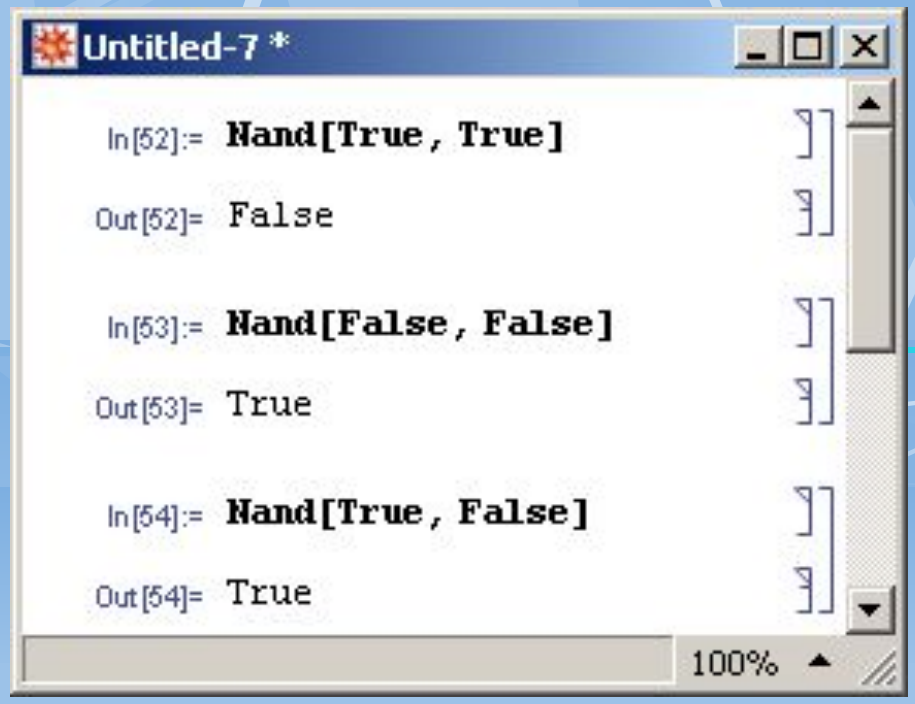
Логічне XOR:

Xor[]

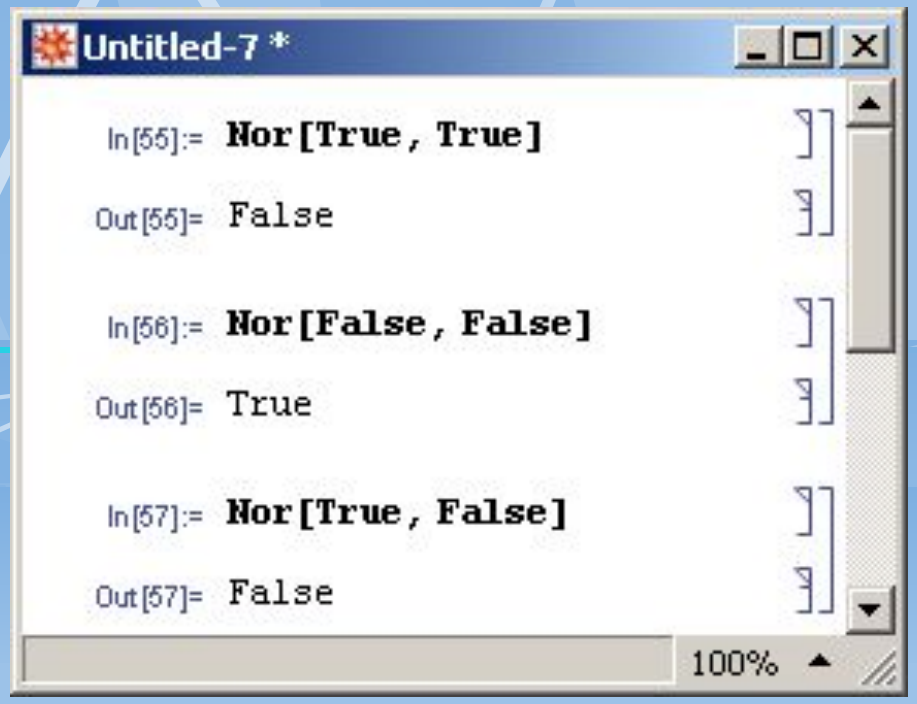


Логічні оператори (2)

Логічне заперечуюче AND: Nand[]



Логічне заперечуюче OR: Nor[]



Оператори порівняння

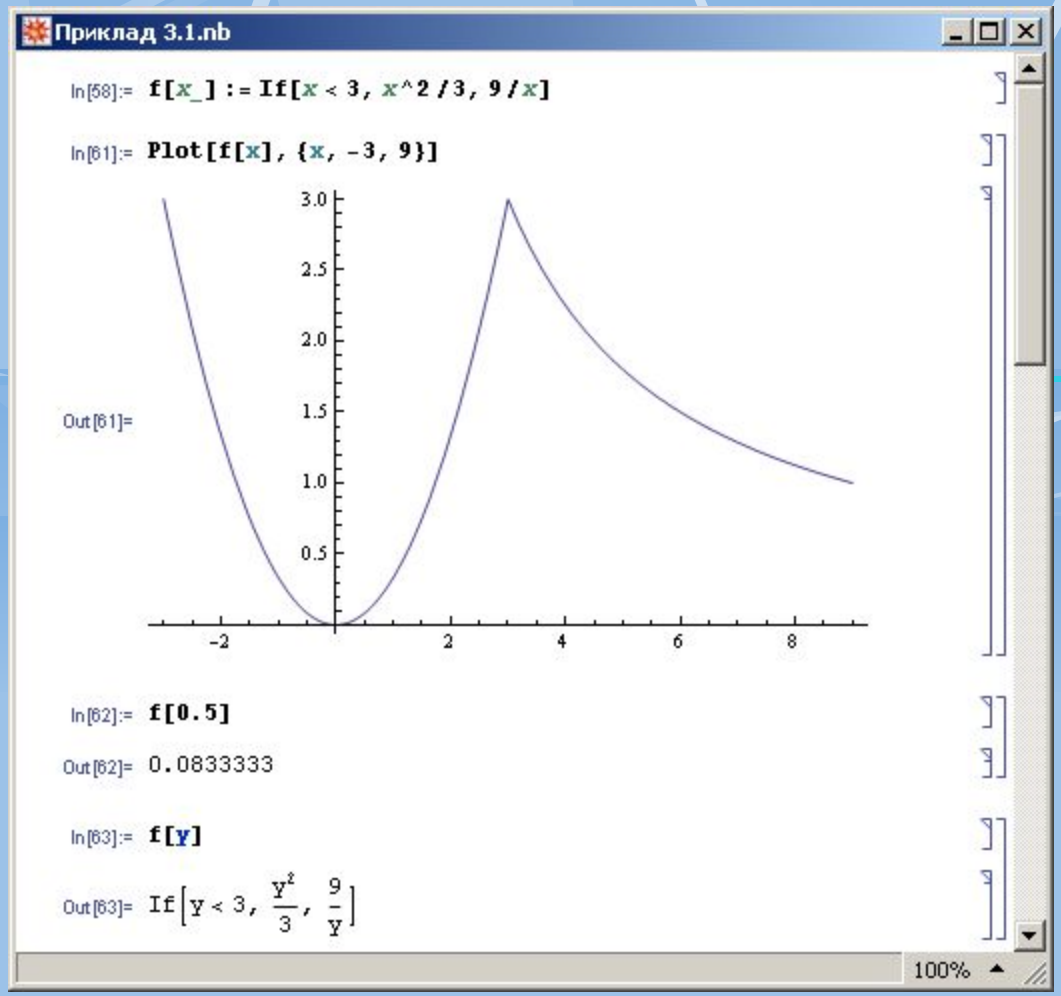
Оператор	Опис
==	Дорівнює
!=	Не дорівнює
===	Тотожно дорівнює
=!=	Тотожно не дорівнює
<	Менше
<=	Менше або дорівнює
>	Більше
>=	Більше або дорівнює

Умовний оператор If[]

Синтаксис: **If [умова,оператор1,оператор2]**

або: **If[умова,оператор1,оператор2,оператор3]**

Приклад 1:



Якщо не може бути розрахована **УМОВА**

Умовний оператор Which[]

Синтаксис:

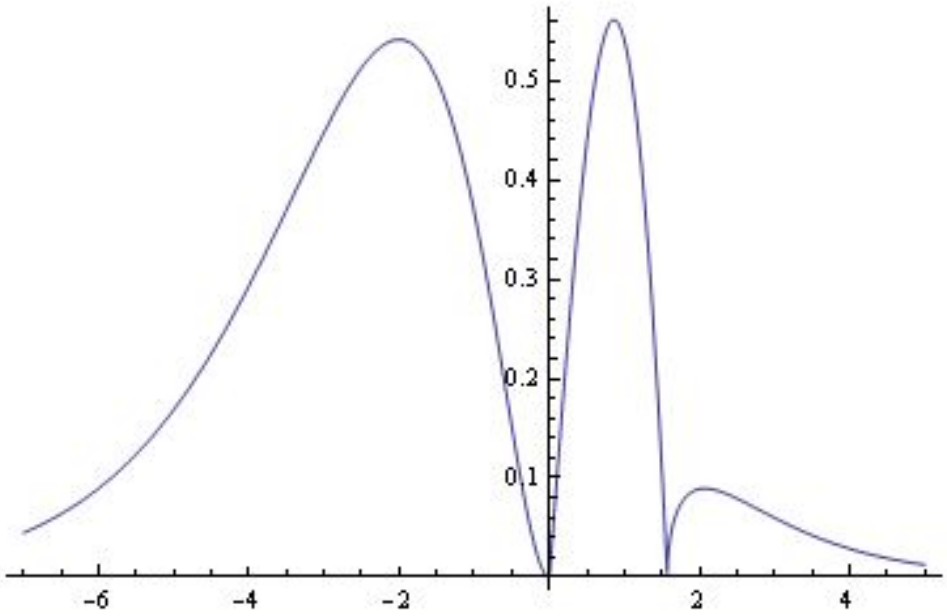
Which[умова1,оператор1, умова2,оператор2,…]

Приклад Приклад 2:

Приклад 3.2.nb

```
In[69]:= f[x_] := Which[x < 0, x^2 * Exp[x], x < π/2, x * Cos[x], x ≥ π/2, Exp[-x] * Sqrt[x - π/2]]

In[71]:= Plot[f[x], {x, -7, 5}]
```

Out[71]=



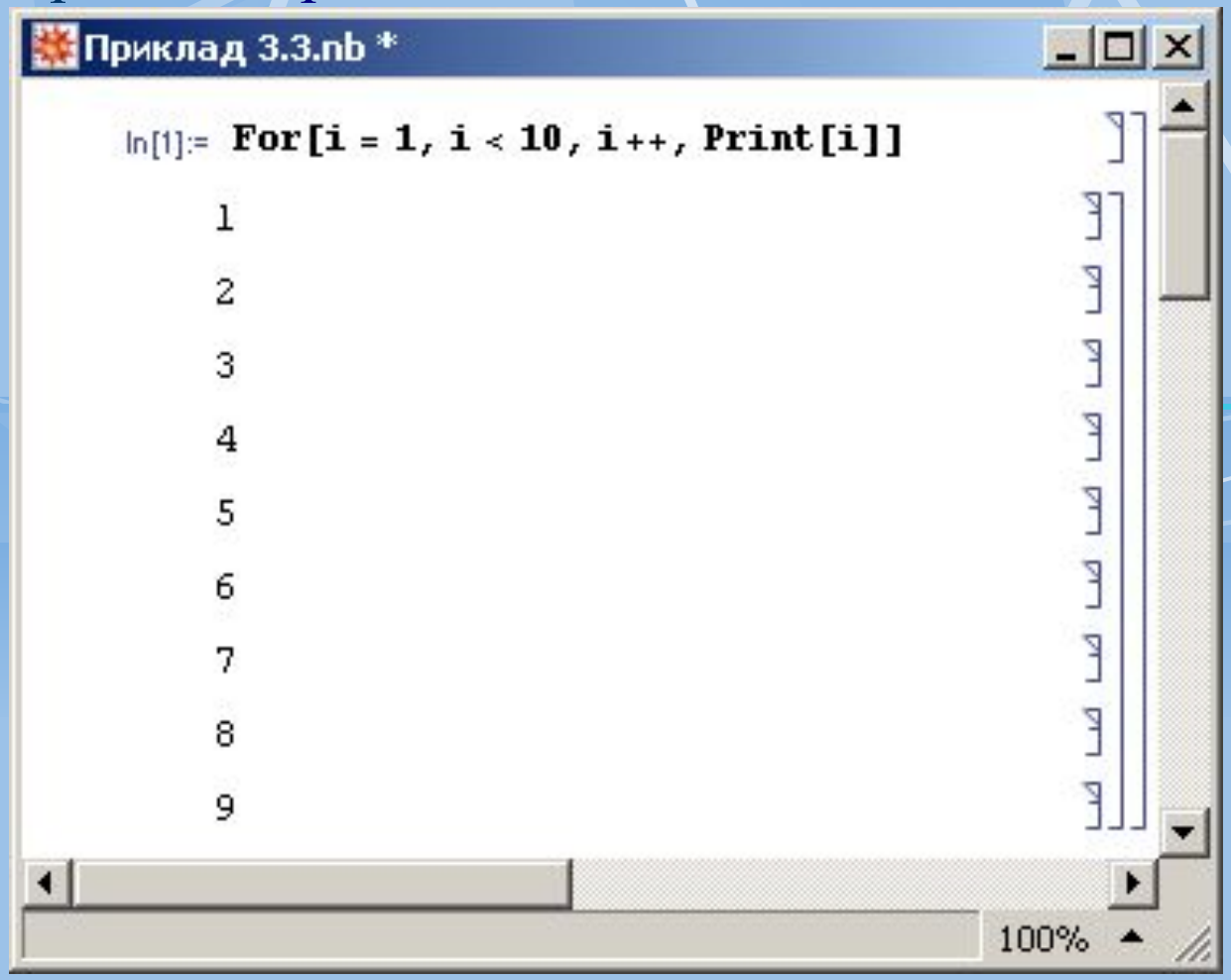
100%

Оператор циклу For[]

Синтаксис:

For[ініціалізація,умова,зміна_параметра_циклу,оператори]

Приклад Приклад 3:

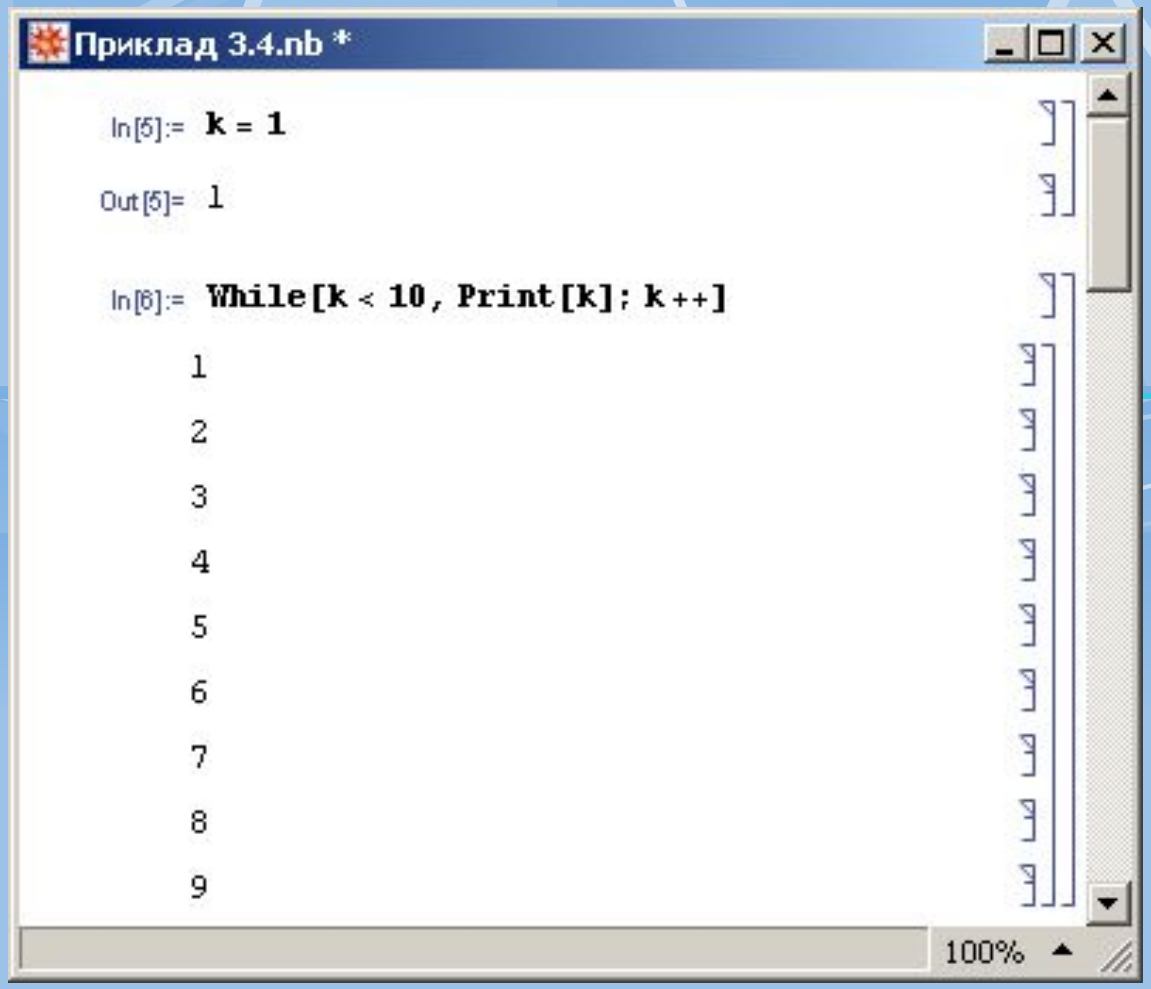


Оператор циклу While[]

Синтаксис:

While[умова,оператори]

Приклад Приклад 4:

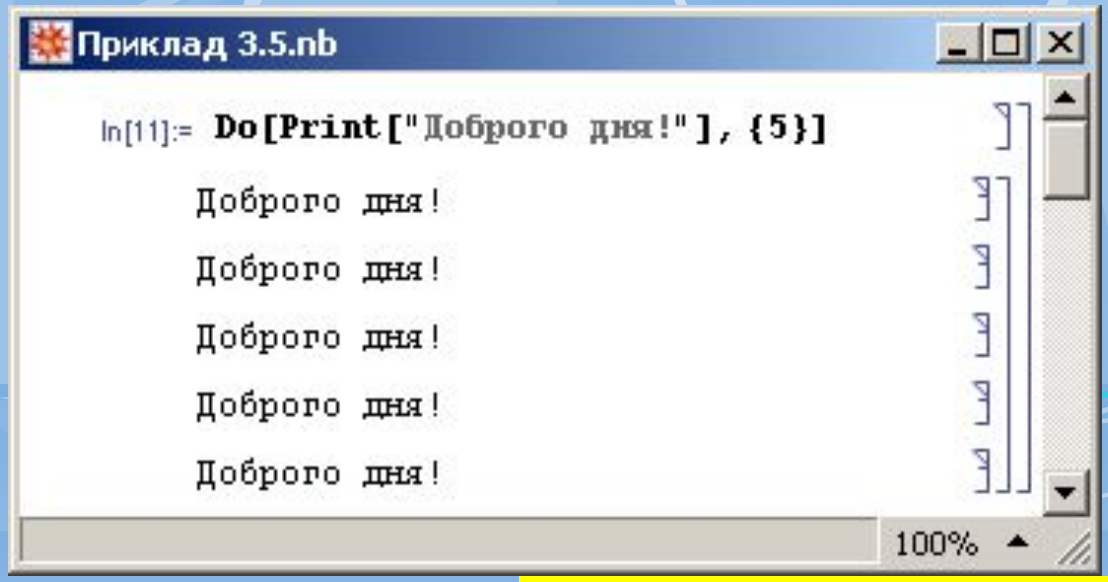


Оператор циклу Do[]

Синтаксис:

Do[оператори,список_кількість_циклів]

Приклад Приклад 5:



1. **Список з кількістю циклів.**
2. **Список з індексною змінною і границею натурального ряду**
3. **Список з індексною змінною, границями діапазону зміни та кроком дискретності**

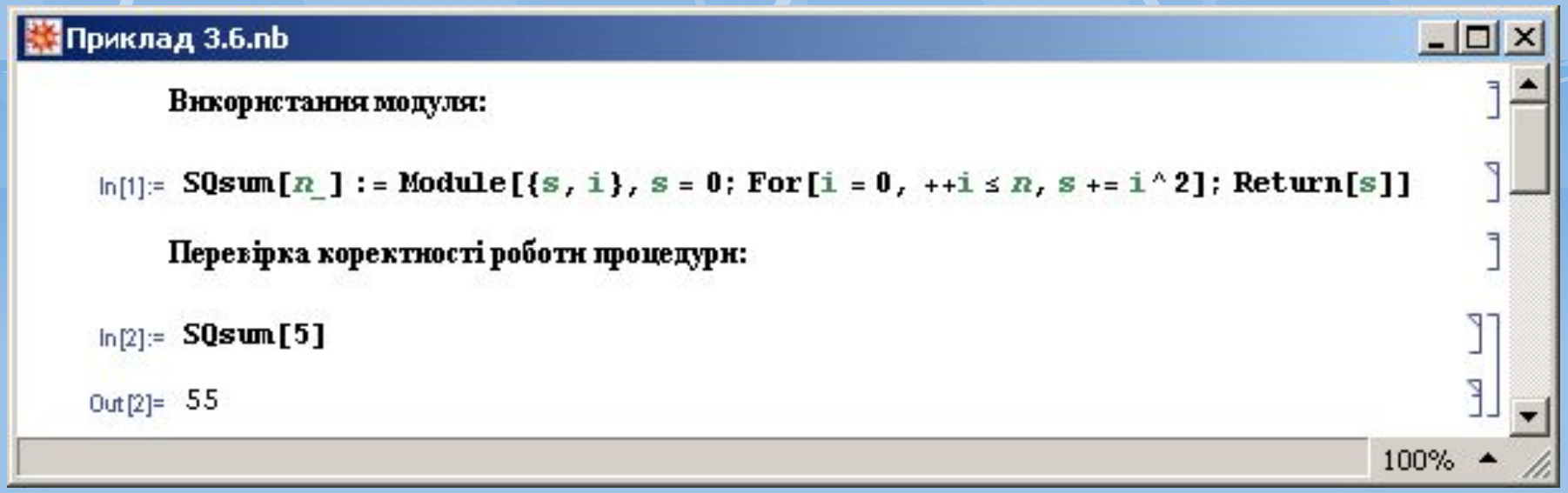
Використання модулів

Функція **Module[]**

Аргументи:

1. Список з локальними змінними модуля
2. Блок операторів модуля
3. Для повернення значення використовують функцію **Return[]**

Приклад 6:



Приклад: числа Фібоначчі

Числа: 1, 1, 2, 3, 5, 8, 13 і т.д.

Приклад 7: