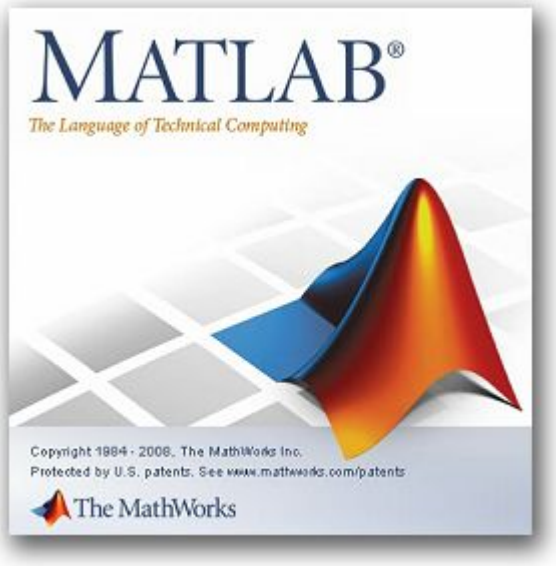
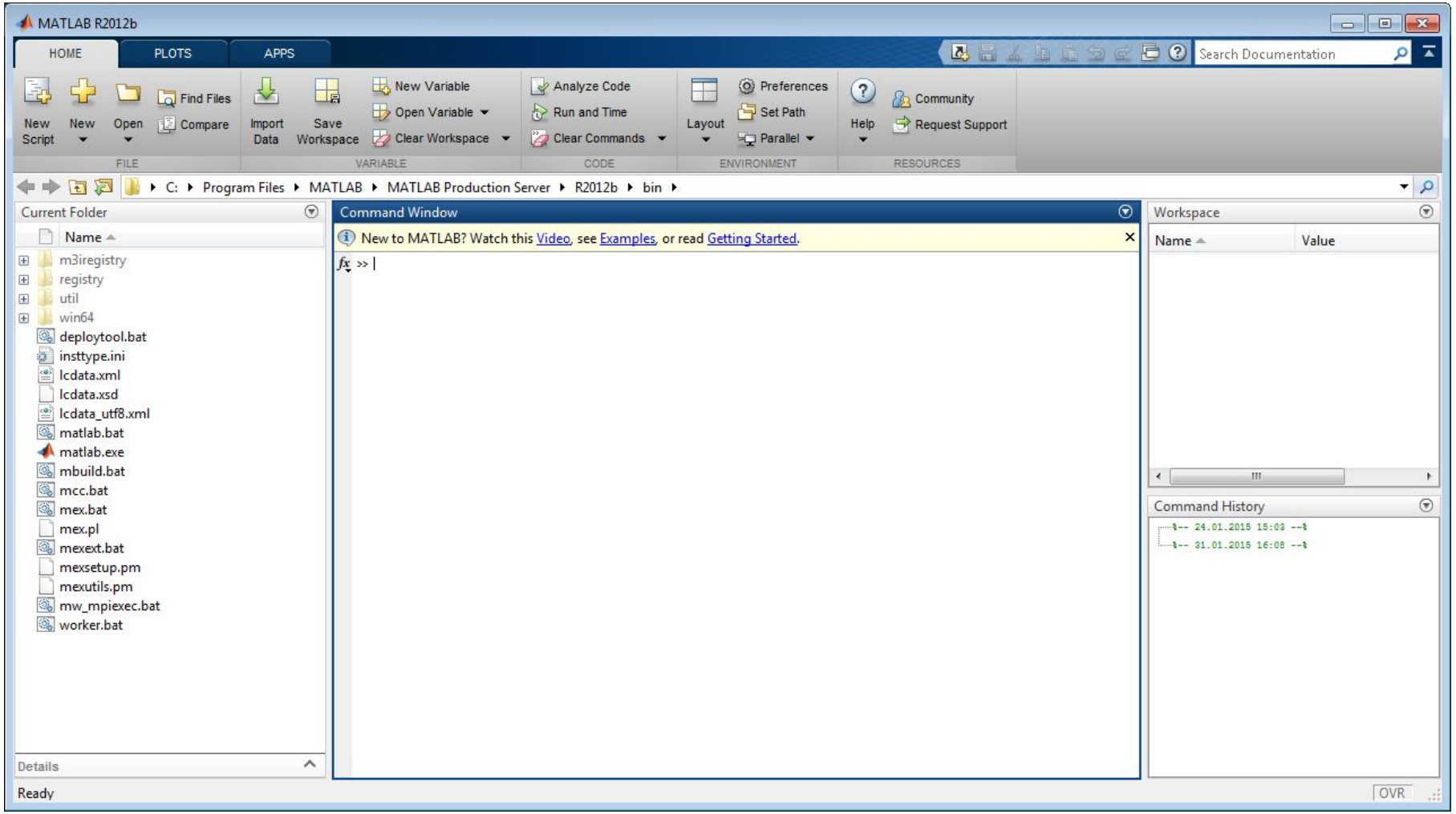
Основи роботи в середовищі MATLAB

MATLAB – це математичний пакет прикладних програм, заснований на використанні матриць. Назва пакету є абревіатурою двох слів MATrix LABoratory (МАТрична ЛАБораторія). MATLAB містить велику кількість спеціалізованих програм (функцій), має власну мову програмування високого рівня, а також надає потужні можливості візуального подання двовимірних та тривимірних даних. Важливу роль у MATLAB відіграють спеціалізовані групи програм (пакети) – так звані Toolbox, в яких зібрані функції для розв'язування окремих класів задач, наприклад, PDE Toolbox, Spline Toolbox та інші.



Робоче середовище MATLAB



Змінні та константи

В середовищі MATLAB визначені декілька спеціальних *констант* – сталих чисельних значень, що позначаються унікальним ідентифікатором та мають математичний смисл:

`pi` – число $\pi \approx 3{,}14159265358979$;

`inf` – нескінченність ∞;

`NaN` – дані нечислового типу (Not a Number);

`eps` – точність, з якою по замовчуванню виконуються обчислення при

Змінні в MATLAB характеризуються іменем, яке також називають ідентифікатором, та значенням. В MATLAB кількість символів в імені змінної обмежена і дорівнює 31. Ідентифікатор повинен починатися з букви та може містити букви, цифри й деякі інші допустимі символи (окрім пробілу). Великі та малі літери в іменах змінних, функцій та команд розрізняються, наприклад, `y` і `Y` є двома різними змінними.

Змінні та константи

Щоб присвоїти змінній деяке значення, використовується знак =

```
>> a=4.75
a =
    4.7500
>> b=-1.53;
```

Змінній окрім сталого числового значення можна присвоїти значення виразу, записавши в командному рядку

```
>> ім'я змінної = вираз
```

Зауважимо, що при відсутності імені змінної в лівій частині та знаку = автоматично створюється змінна *ans* (скорочено від англ. "answer"), якій присвоюється результат обчислення виразу.

Приклад 1. Обчислити в MATLAB значення виразу $4^3 + 7x \cdot (5{,}2 - \frac{1}{3})$ при $x = 1{,}03$:

```
>> x=1.03;
>> 4^3+7*x*(5.2-1/3)
ans =
   99.0887
```

Вбудовані елементарні математичні функції

Формати подання даних в пакеті MATLAB

Команда	Опис формату подання числа
`format short`	з фіксованою крапкою та 4 знаками після крапки (встановлений по замовчуванню)
`format long`	з фіксованою крапкою та 14 знаками після крапки
`format short e`	з плаваючою крапкою та 4 десятковими знаками
`format long e`	з плаваючою крапкою та 15 десятковими знаками
`format short g`	з фіксованою крапкою та 3 знаками після крапки
`format long g`	з фіксованою крапкою та 13 знаками після крапки
`format rat`	у вигляді звичайних дробів
`format hex`	в шістнадцятковій системі числення

– *алгебраїчні та арифметичні функції*

`abs(x)` – абсолютне значення (модуль) x;

`exp(x)` – експоненціальна функція e^x;

`log(x)`, `log10(x)`, `log2(x)` – натуральний, десятковий та двійковий логарифми;

`pow2(x)` – піднесення числа 2 до степеня x;

`sqrt(x)` – квадратний корінь з x;

– *тригонометричні та обернені до них функції*

`sin(x)`, `cos(x)`, `tan(x)`, `cot(x)` – синус, косинус, тангенс, котангенс;

`sign(x)` – знак числа x;

`mod(x,y)` – залишок від цілочисельного ділення x на y (зі знаком);

`rem(x,y)` – залишок від цілочисельного ділення x на y.

Матриці

Всі змінні в середовищі MATLAB являють собою матриці (або масиви). Так, скаляр (число) в MATLAB є масивом розмірності 1x1, вектор – матрицею з одним рядком або одним стовпчиком.

Приклад 3.

```
>> A=[1 2 3; 4 5 6; 7 8 9]
A =
     1     2     3
     4     5     6
     7     8     9
```

Приклад 4.

```
>> x=[3 1 2 9]
x =
     3     1     2     9
>> y=[15;10]
y =
    15
    10
```

Команду для створення вектор-рядка, всі елементи якого відрізняються один від одного на постійний крок, можна умовно записати:

```
>> ім'я вектора=[початкове значення:крок:кінцеве значення]
```

Приклад 5. Створимо вектор-рядок $z = (0 \quad 0{,}2 \quad 0{,}4 \quad 0{,}6 \quad 0{,}8 \quad 1)$ з використанням оператору двокрапки:

```
>> z=0:0.2:1
z =
     0    0.2000    0.4000    0.6000    0.8000    1.0000
```

Матриці

Відмітимо, що з використанням оператора : можна автоматично створювати не лише вектори, а й матриці. Так, команда

```
>> B=[1:2:7; 8, 2:4; 12:-1:9; 14, 13, 16, 15]
```

призводить до створення наступної матриці

```
B =
     1     3     5     7
     8     2     3     4
    12    11    10     9
    14    13    16    15
```

Функції формування спеціальних матриць

Тип матриці	Функція для створення	
	квадратна матриця розміру $n \times n$	прямокутна матриця розміру $m \times n$
Матриця з нульовими елементами	`zeros(n)`	`zeros(m,n)`
Матриця з одиничними елементами	`ones(n)`	`ones(m,n)`
Одинична матриця	`eye(n)`	`eye(m,n)`
Матриця випадкових чисел, рівномірно розподілених між нулем та одиницею	`rand(n)`	`rand(m,n)`
Матриця випадкових чисел, розподілених за нормальним законом	`randn(n)`	`randn(m,n)`

Математичні операції з матрицями

Звернення до елементів векторів і матриць здійснюється за допомогою *індексів*. Індекси вказуються в круглих дужках після імені матриці або вектора і являють собою порядковий номер елемента (у випадку векторів) або номер рядка та стовпчика, в яких розташований елемент (у випадку матриць),

Над матрицями можна виконувати ті ж самі арифметичні операції, що й над числами – додавання (+) і віднімання (–), множення (*) і ділення (/), піднесення до степеня (^), – а також операцію транспонування ('). Всі ці операції виконуються одразу над цілими матрицями, тому вони називаються *матричними*. При застосуванні матричних операцій слід пам'ятати наступне:

- для додавання або віднімання матриці повинні бути однакового розміру;
- при множенні кількість стовпчиків першої матриці (та, що стоїть зліва від оператора *) обов'язково повинна дорівнювати кількості рядків другої матриці;
- операція піднесення до степеня застосовується лише для квадратних матриць.

Матриці

операції також можна виконувати поелементно, вказавши перед ними крапку: `.*`, `./` та `.^`

Приклад 7. Виконаємо в MATLAB всі наведені вище операції для наступних матриць:

```
>> A=[1 2; 3 4];
>> B=[0 -1; 5 8];
```

Знайдемо результати додавання, віднімання, множення та ділення матриць `A` і `B`:

```
>> A+B
ans =
     1     1
     8    12
```

```
>> A-B
ans =
     1     3
    -2    -4
```

```
>> A*B
ans =
    10    15
    20    29
```

```
>> A/B
ans =
  -0.4000   0.2000
   0.8000   0.6000
```

Операції піднесення до степеня та транспонування застосовуються до однієї матриці:

```
>> A^2
ans =
     7    10
    15    22
```

```
>> A'
ans =
     1     3
     2     4
```

Виконаємо над матрицями `A` і `B` поелементні операції множення та ділення:

```
>> A.*B
ans =
     0    -2
    15    32
```

```
>> B./A
ans =
         0   -0.5000
    1.6667    2.0000
```

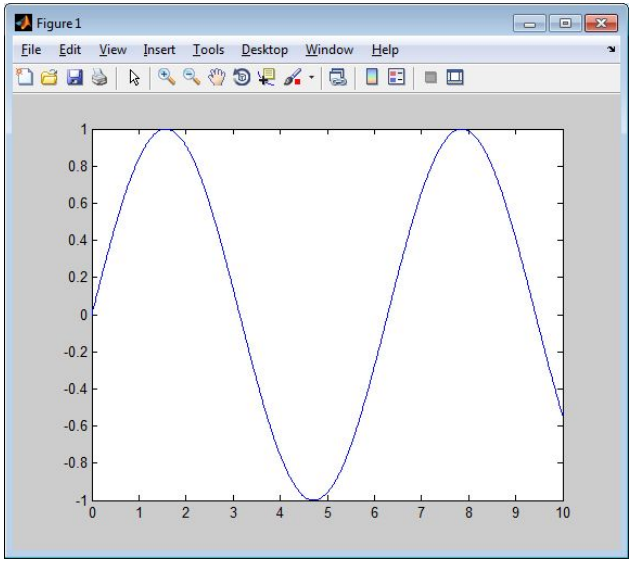
Графіки функцій

Можливими формами виклику функції `plot` є:

- `plot(y)` – побудова графіку залежності елементів вектора `y` від їх індексів;
- `plot(x,y)` – побудова графіку залежності `y` від `x`;
- `plot(x,y,s)` – побудова графіку залежності `y` від `x` зі стилем лінії (колір і тип лінії, тип маркеру), заданим аргументом `s`;
- `plot(x1,y1,x2,y2,…,xn,yn)` – побудова графіків декількох функцій на одних координатних осях;
- `plot(x1,y1,s1,x2,y2,s2,…,xn,yn,sn)` – побудова на одних координатних осях графіків декількох функцій з відповідними стилями ліній.

Приклад 8. Побудуємо графік функції $y(x) = \sin x$ на відрізку $[0, 10]$:

```
>> x=0:0.01:10;
>> y=sin(x);
>> plot(x,y)
```



Графіки функцій

Приклад 9. Побудуємо на одному графіку функції $y(x) = \sin x$ і $f(x) = \cos x$ на відрізку $[0, 10]$.

Сформуємо вектори `x`, `y` та `z`:

```
>> x=0:0.01:10;
>> y=sin(x);
>> z=cos(x);
>> hold on
>> plot(x,y)
>> plot(x,z)
```

Альтернативний спосіб побудови двох функцій на одних координатних осях полягає у використанні команди `plot` з двома парами аргументів:

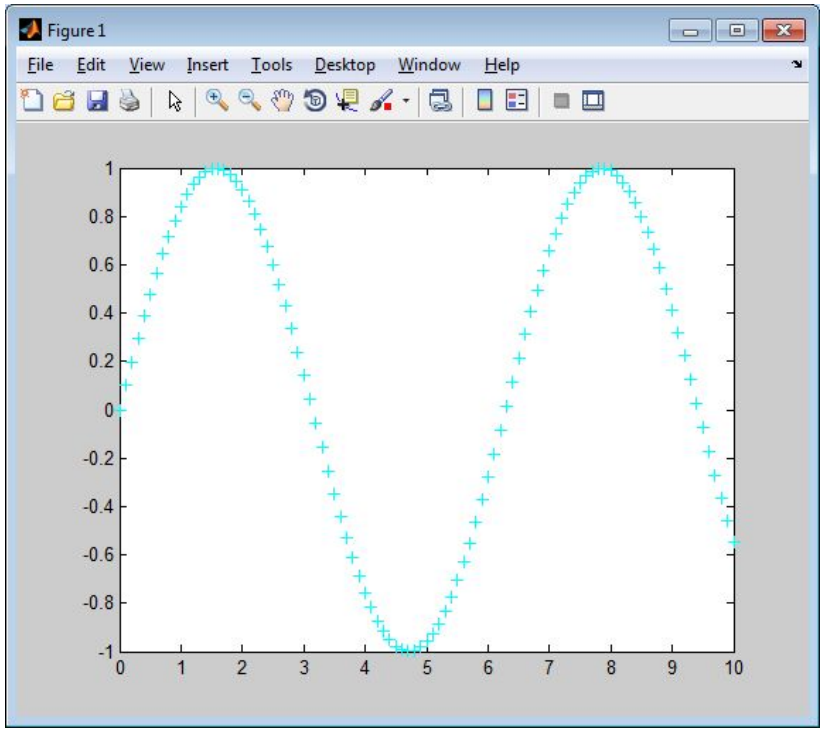
```
>> plot(x,y,x,z)
```

MATLAB надає користувачам можливість змінювати стиль ліній графіків (колір і тип ліній, тип маркерів). Для цього служить додатковий аргумент функції `plot`, що задається після кожної пари векторів:

```
plot(x,y,'колір_стиль_маркер')
plot(x1,y1,'колір_стиль_маркер',…,xn,yn,'колір_стиль_маркер')
```

Стилі графіків

```
>> x=0:0.1:10;
>> y=sin(x);
>> plot(x,y,'c+')
```



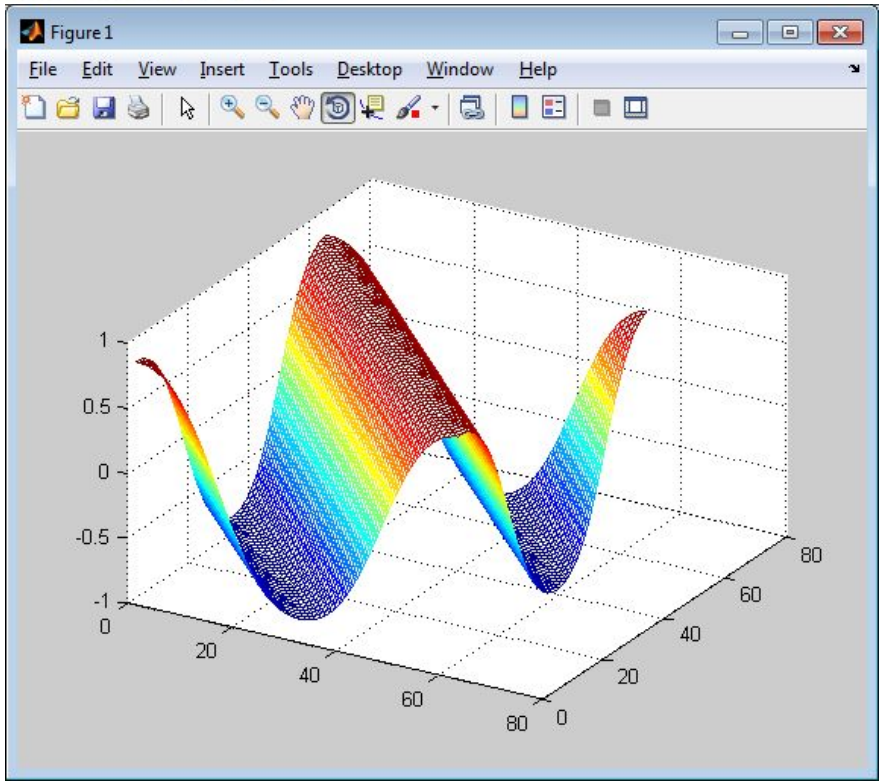
Колір лінії		Тип лінії		Тип маркера	
y	жовтий	-	суцільна	.	крапка
m	рожевий	:	пунктирна	o	круг
c	блакитний	-.	штрих-пунктирна	x	хрест
r	червоний	--	штрихова	+	знак “плюс”
g	зелений			*	зірочка
b	синій			s	квадрат
w	білий			d	ромб
k	чорний			p	п’ятикутна зірка

Графіки функцій

Для побудови графіків тривимірних поверхонь, спочатку необхідно задати прямокутну область визначення функції за допомогою команди `meshgrid`, а потім скористатись командою `mesh` або `surf` для побудови поверхні.

Приклад 11. Побудуємо графік функції $z(x,y)=\sin(\cos(x+y))$, де $x\in[-\pi,\pi]$, $y\in[-\pi,\pi]$:

```
>> [x,y]=meshgrid(-pi:0.1:pi,-pi:0.1:pi);
>> z=sin(cos(x+y));
>> mesh(z)
```



Елементи мови програмування MATLAB

Умовний оператор

Мова програмування MATLAB має декілька структур для керування потоками, аналогічні їх прототипам в інших мовах програмування:

1) *умовний оператор* – може використовуватись в одній із трьох форм:

```
if умова
    оператори
end
```

```
if умова
    оператори1
else
    оператори2
end
```

```
if умова1
    оператори1
elseif умова2
    оператори2
…
elseif умова_n
    оператори_n
else
    оператори
end
```

Оператори порівняння		Логічні оператори		Булеві величини	
==	Дорівнює	~ (not)	Заперечення	1	True
~=	Не дорівнює	& (and)	Логічне “і”	0	False
<	Менше	| (or)	Логічне “або”		
>	Більше				
<=	Менше або дорівнює				
>=	Більше або дорівнює				

Оператор вибору

2) *оператор вибору* – виконує ту чи іншу групу операторів в залежності від значення виразу (чи змінної), при цьому виконується лише перший відповідний випадок:

```
switch вираз
case значення1
    оператори1
case {значення2,…}
    оператори2
…
otherwise % може бути відсутнім
    оператори_n
end
```

Оператори циклу

3) *оператор циклу* `for` – виконує групу операторів всередині циклу фіксовану кількість разів:

```
for змінна циклу = початкове значення:крок:кінцеве значення
    оператори
end
```

Якщо крок дорівнює одиниці, то його можна не вказувати:

```
for змінна циклу = початкове значення:кінцеве значення
    оператори
end
```

4) *оператор циклу* `while` – виконує групу операторів всередині циклу, поки виконується логічна умова:

```
while умова
    оператори
end
```

М-файли

Розрізняють два типи М-файлів: сценарії та функції.

Файли-сценарії містять послідовність команд середовища MATLAB. Можуть мати довільну назву, обов'язковим є тільки розширення .m. Викликається сценарій введенням назви імені файлу в командному рядку. Сценарії не приймають та не повертають ніяких аргументів, вони оперують існуючими змінними в робочій області (глобальними) та можуть самі створювати нові змінні, що залишаються в Workspace. Типовим випадком використання сценаріїв є запис команд у М-файл, замість їх введення у командному рядку.

Файли-функції можуть мати вхідні та вихідні аргументи. Ім'я функції та М-файлу (з розширенням .m) повинні співпадати. Змінні, що використовуються в М-функціях по замовчуванню є локальними, хоча існує можливість оголошення глобальних змінних. Структура функції має вигляд:

```
function [вихідні аргументи]=ім'я функції(вхідні аргументи)
% Основний коментар
% Додатковий коментар
Тіло функції (виконуваний код)
```

М-файли. Приклад

Приклад 1. Визначимо функцію $f(x)=1+x-x^2/4$ в М-файлі з ім'ям `f.m`. В редакторі М-файлів вона записується наступним чином:

```
function y=f(x)
y=1+x-x.^2/4;
```

Функцію, одного разу збережену в М-файлі, можна викликати в командному вікні MATLAB так само, як і будь-яку іншу функцію. Наприклад, щоб обчислити $\cos(f(3))$ достатньо ввести в командному рядку:

```
>> cos(f(3))
ans =
      -0.1782
```

Зауважимо, що завдяки використанню поелементних операцій, вхідним аргументом функції `f` може бути не тільки одне число, але й вектор. Результатом обчислень в такому випадку також буде вектор:

```
>> x=[0 2 3];
>> f(x)
ans =
    1.0000    2.0000    1.7500
```

М-файли. Приклад

Приклад 2. Скласти в MATLAB файл-функцію для обчислення значення кусково-заданої функції:

$$f(x)=\begin{cases}1-e^{-1-x}, & x<-1\\ x^2-x-2, & -1\le x\le 2\\ 2-x, & x>2\end{cases}$$

Введемо в редакторі функцію `pwf` та збережемо її у М-файлі з ім'ям `pwf.m`:

```
function f=pwf(x)
if x<-1
    f=1-exp(-1-x);
elseif x<=2
    f=x.^2-x-2;
else
    f=2-x;
end
```

Щоб обчислити значення функції при $x=0$, необхідно в командному рядку ввести:

```
>> pwf(0)
ans =
    -2
```

М-файли. Приклад

Приклад 3. Обчислити суму $\sum_{k=1}^{n} \frac{1}{k!}$.

Введемо в редакторі наступні функції та збережемо їх у М-файлі з ім'ям `suma.m`:

```
function S=suma(n)
S=0;
for k=1:n
    S=S+1/fact(k);
end

function f=fact(n)
%підфункція для обчислення n!
f=1;
for i=1:n
    f=f*i;
end
```

М-файли. Приклад

Приклад 4. Знайти суму та добуток всіх додатних елементів заданої матриці.

Введемо наступну функцію в редакторі М-файлів та збережемо її у файлі з ім'ям `posdob.m`:

```
function [S,D]=posdob(A)
S=0;
D=1;
[n,m]=size(A);
for i=1:n
    for j=1:m
        if A(i,j)>0
            S=S+A(i,j);
            D=D*A(i,j);
        end
    end
end
```

Тут `size(A)` – вбудована функція для визначення розміру матриці `A`: `n` – кількість рядків, `m` – кількість стовпчиків. В MATLAB також є вбудована функція `length(x)`, яка знаходить довжину вектора `x`.