Лекція 7
Основи програмування в системі MATLAB

1. Особливості програмування у MATLAB

Більшість математичних систем створювалися зважаючи на те, що користувач буде розв'язувати свої задачі, практично не займаючись програмуванням. Однак, з самого початку було зрозуміло, що подібний підхід має недоліки й, загалом кажучи, є порочним. Адже значна кількість задач потребує розвинених засобів програмування, які спрощують запис алгоритмів задач та інколи відкривають нові методи створення алгоритмів.

З одного боку MATLAB містить величезну кількість операторів та функцій для розв'язання різноманітних практичних задач, що знімає потребу написання доволі складних програм. Приміром, це функції обернення та транспонування матриць, обчислення значень похідних та інтегралів тощо. Кількість таких функцій з урахуванням пакетів розширення системи вже сягає багатьох тисяч і безупинно збільшується.

Але з іншого боку, з моменту свого створення, система MATLAB розроблялась як потужна математико-орієнтована мова програмування високого рівня. Ця важлива перевага системи свідчить про можливості її застосування для розв'язування нових, найбільш складних математичних задач.

Система MATLAB має вхідну мову, що нагадує Basic (з домішкою Fortran та Pascal). Запис програм у системі є традиційним і звичним для більшості користувачів комп'ютерів. Крім того, система надає можливість редагувати програми за допомогою будь-якого звичного для користувача текстового редактора. Вона має і свій власний редактор з відлагоджувачем. Відмова від притаманної для системи Mathcad переваги – записування задач у вигляді формул – компенсується помітним збільшенням швидкості обчислень – за інших однакових умов вона майже на порядок є вищою, аніж у системи Mathcad, а це є суттєвою перевагою. Мова системи MATLAB у частині програмування математичних обчислень набагато багатше будь-якої універсальної мови програмування високого рівня. Вона реалізує майже всі відомі засоби програмування, у тому числі об'єктно-орієнтоване й (засобами Simulink) візуальне програмування. Це надає досвідченим програмістам неосяжні можливості для самовиразу.

2. Базові конструкції мови MATLAB

Програма мовою MATLAB є послідовністю команд, які в найпростіших випадках виконуються послідовно одна за одною. Однак досить часто виникає необхідність у більш складних алгоритмічних конструкціях, які потребують розгалужень, циклів тощо.

В алгоритмічній мові MATLAB використовуються програмні конструкції та оператори, що є традиційними для багатьох середовищ програмування.

Умовний оператор виступає в одній із наступних форм:

```
if <умова>
    <команди>
end
```

```
if <умова>
    <команди>
else
    <команди>
end
```

```
if <умова>
    <команди>
elseif <умова>
    <команди>
else
    <команди>
end
```

У ролі умови може використовуватися будь-який логічний вираз, побудований на основі операцій відношення і логічних. Якщо значення цього виразу є масивом, то умова вважається істинною, коли всі його елементи істинні (істина – 1, хибність – 0).

Наприклад, якщо треба визначити абсолютне значення дійсного числа x, це можна записати таким чином:

```
if x<0
  x=-x;
end
```

Реалізація формули $y=\begin{cases}\sin x, & x<5,\\ \cos x, & x\geq 5\end{cases}$ потребує розгалуження алгоритму на дві гілки, це можна записати як

```
if x<5
  y=sin(x);
else
  y=cos(x);
end
```

Зверніть увагу, що після `else` не треба писати `x>=5`, це мається на увазі автоматично. Після `else` ідуть оператори, які виконуються тоді, коли умова `x<5` не виконується. Слово `else` означає «у противному випадку, інакше».

І нарешті розглянемо приклад складнішої умови, яка потребує розгалуження алгоритму на три маршрути

$$y=\begin{cases} x^2+1, & x<5, \\ e^{-x}, & 5\le x\le 10, \\ \dfrac{2}{x-3}, & x>10. \end{cases}$$

Реалізація такої формули можлива з використанням третьої форми запису умовного оператора

```
if x<5
  y=x^2+1;
elseif x<=10
  y=exp(-x);
else
  y=2/(x-3);
end
```

Оператор циклу із заданим числом повторень, в основному використовуваний у формі:

for V=A:H:B	**for** V=A:B
<команди>	<команди>
end	**end**

(V – змінна/параметр циклу, A, B – початкове і кінцеве значення; H – збільшення, за замовчуванням 1). Допускаються і вкладені цикли, наприклад:

```
for i=1:n
   for j=1:m
      a(i,j)=x(i)^j;
   end
end
```

У заголовку циклу можна використати одновимірний масив. Так цикл

```
k=1;
for i=[0 5 7]
   x(k)=2^i;
   k=k+1;
   end
```

формує масив x=[1 32 128].

Оператор циклу із передумовою має традиційну конструкцію:

while <умова>
    <команди>
end

і забезпечує виконання команд тіла циклу, поки умова, що перевіряється, є істинною. Зауважимо, що робота циклу може бути перервана (вихід із внутрішнього циклу) оператором **break**:

```
while a<1
   n=n+1
   if n>250
      break
end ...
```

Оператор перемикання узагальнює умовний оператор на випадок більше двох умов і має конструкцію:

switch <вираз>
    case <значення 1>
        <команди>
    case <значення 2>
        <команди>
    .....
    otherwise % може бути відсутня
        <команди>
    end

Контрольні значення перевіряються на рівність і можуть задаватися й списком:

```
switch k
    case 0
        t=1
    case (1,2,5)
        t=2
    otherwise
        t=0
    end
```

Крім згаданих основних операторів, традиційних для будь-якої системи програмування, зупинимося на ряді операторів забезпечення інтерфейсу користувача.

Уведення із клавіатури реалізується командою виду

<змінна>= **input** ('підказка')

Наприклад,

```
» x=input(' степінь полінома: ');
степінь полінома: 3
```

Припинення виконання програми може бути передбачено включенням у текст команди **pause** (припинення до натискання будь-якої клавіші), **pause (n)** (припинення на n сек), **keyboard** (припинення з можливістю виконувати практично будь-які команди й наступне повернення в програму командою **return**).

Можна побудувати вибір варіанта із клавіатури створенням ***меню***:

<змінна>=**menu**('заголовок', 'вибір1', 'вибір2',...)

Наприклад, команда:

```
k=menu('Вибрати метод','Гауса','Крамера','Ньютона')
```

створить на екрані спливаюче меню (рис. 7.1) із зазначеними пунктами-клавішами і клацання мишкою по клавіші задасть значення змінної k, що дорівнює 1, 2 або 3.

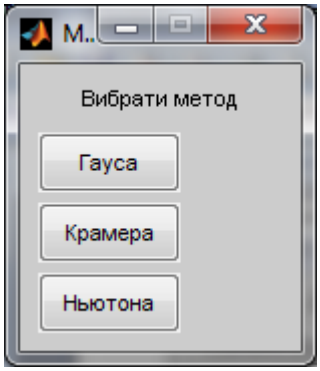

Рисунок 7.1 – Вікно меню

Ми не зупиняємося на різноманітті операторів, пов’язаних з виводом на екран (вивід значення **disp,** форматований вивід **fprint**), налагодженням і сигналізацією про помилки, аналізом списку аргументів і ін.

3. Редагування і відлагодження m-файлів

MATLAB дозволяє записувати послідовність дій програми в окремих текстових файлах з розширенням .m, або m-файлах. Підготовлений і записаний на диску m-файл стає частиною системи, і його можна викликати як з командного рядка, так і з іншого m-файла. При створенні m-файли проходять синтаксичний контроль за допомогою вбудованого в систему MATLAB редактора / відлагоджувача m-файлів.

Існує два різновиди m-файлів: файли-функції та файли-сценарії:

файли-функції, що мають вхідні параметри, список яких зазначається у круглих дужках. Застосовувані у файлі-функції змінні є локальними змінними;

файл-сценарій чи script-файл є простим записом команд вхідних параметрів.

Для m-функції допускаються вхідні й вихідні аргументи, локалізація внутрішніх її змінних і можливість звертання до неї з інших програм. m-функції включаються в бібліотеку функцій системи у вигляді текстових файлів.

Заголовок m-функції має вигляд:

function [<список вихідних змінних>**]=**
<ім'я функції> **(**<список вхідних змінних>**)**

наприклад, функція обчислення факторіала додатного числа і його оберненої величини може бути описана файлом fact.m:

```
function [f, g]=fact(n)    % факторіал і обернена
                           %величина
f=prod(1:n);
g=1/f;
```

Вихід з функції в програму забезпечується виконанням останнього її оператора або командою **return.**

Для запуску script-файла на виконання з командного рядка MATLAB достатньо записати його ім’я в цьому рядку:

```
>> ім’я_файла.m
```

Для того, щоб створити файл, слід виконати команди з меню *File / New*, після чого відкриється вікно створення й редагування текстів програм, в якому послідовно записують і виконують програмні команди (рис. 7.2).

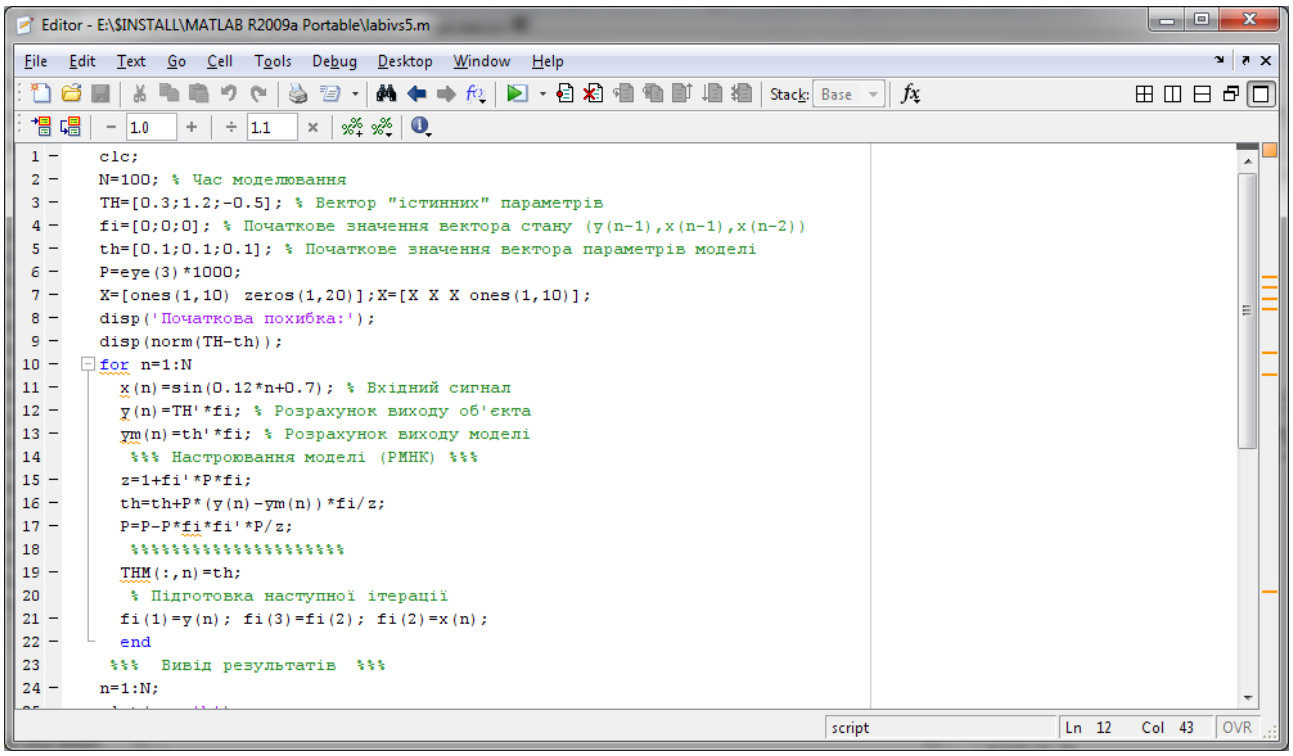

Рисунок 7.2 – Вікно редактора програм

Редактор/відлагоджувач m-файлів виконує синтаксичний контроль програмного коду по мірі введення тексту. При цьому використовуються такі кольорові виокремлення:

ключові слова мови програмування – синій колір;

оператори, константи і змінні – чорний колір;

коментарі після знаку % – зелений колір;

символьні змінні (в апострофах) – зелений колір;

синтаксичні помилки – червоний колір.

Завдяки кольоровим виокремленням вірогідність синтаксичних помилок знижується. Однак далеко не всі помилки діагностуються. Помилки, пов’язані з неправильним застосування операторів чи функцій (приміром, застосування оператора “–” замість “+” чи функції `cos(x)` замість `sin(x)` тощо), не здатна виявити жодна система програмування. Усунення такого роду помилок (їх називають семантичними) – справа користувача, що відлагоджує свої алгоритми і програми.