4. Работа с файлове

В MATLAB се използват три вида файлове:

- Скриптови файлове (`Script Files`);
- Файл-функции (`File Functions`);
- Файлове с данни (`Data Files`).

4.1. Скриптови файлове (Script)

Скриптовите файлове са обикновени ASCII-файлове, които съдържат команди на MATLAB, оформени като програма, която извършва обработка на данни според зададен/разработен алгоритъм. Скриптовият файл се създава в работната среда на MATLAB в следната последователност:

• Избира се последователността: меню File → New → M-file или се натиска първия бутон от лентата с инструменти, изобразяващ празен бял лист. При това се стартира редакторът `medit` на MATLAB**;**

• От клавиатурата се въвеждат командите на MATLAB, като се спазват познатите вече правила за въвеждане на команди в диалогов режим;

• От основното меню се избира командата File → Save As… .

• В появилия се прозорец Save file as се избира папката (директорията), въвежда се името на файла в полето File name и се натиска бутона Save. С това файлът е записан на твърдия диск с разширение **.m.**

За изпълнението на файла е достатъчно да се въведе името му (без разширението `.m`) в командния прозорец.

>> **`file_name`** Enter

В резултат на това действие системата ще претърси най-напред текущата директория за файл с това име и разширение .m. Ако не го открие в текущата директория, продължава търсенето във всички останали директории в последователност, съответстваща на реда им в списъка на достъпните директории. След като търсения файл е намерен, системата изпълнява всички команди, записани в него. Работната папка по подразбиране, в която се съхраняват файловете, е Work.

Ако във файла има синтактични грешки, системата спира изпълнението при първата грешка и извежда съобщение за грешка. Текстът на съобщението е с червен цвят и започва с три въпросителни. Съобщението за грешка указва името на файла, реда и позицията, където е открита грешката, както и типа на тази грешка:

Пример: съобщение за грешка

```
??? Error: File: C:\MATLAB6p5\work\semicircle2.m Line: 9
Column: 1 "end" expected, "End of Input" found.
```

Това съобщение означава: открита е грешка във файл с име `semicircle2.m`, който се намира в папка C:\MATLAB6p5\work. Грешката е на ред 9 в позиция (колона) 1. Типа на грешката е, че липсва (очаква се да има) служебната дума " end " като част от оператор.

За да се коригира грешката се отваря файла с командата File → Open… или с двойно щракване върху името му в прозореца Current Directory. Коригират се грешките и се записва коригираната версия на файла с командата File → Save. За да се изпълни коригираната програма отново се въвежда името на файла на командния ред. Ако отново се получи съобщение за грешка е необходимо да се коригира и тази грешка по описания начин. Това се

повтаря, докато се коригират всички грешки от програмата. Удобно е, докато трае тази процедура, да не се затваря редактора, а само да се превключва от прозореца на редактора към командния прозорец и обратно.

Описаната процедура на корекции коригира само синтактичните грешки. За да се коригират логически грешки в програмата, тя трябва да се тества с контролни входни данни, за които предварително се знае какво е вярното решение.

Името на скрипт файл не може да съвпада със служебно име (дума) от MATLAB. Не може да съществуват два файла с едно и също име.

Характерни особености на скриптовите файлове са:

- Скриптовите файлове представляват самостоятелно изпълнявани блокове от оператори и команди;
- В скриптовия файл не се използва служебна дума за начало на файла;
- Всички променливи на изпълняваните в текущата сесия скриптови файлове се съхраняват в едно и също работно пространство (Work Space). Това означава, че имената на променливите в тези файлове трябва да са съгласувани, ако решават съвместно дадена задача. Ако даден скриптов файл не ползва резултатите от друг, изпълняван преди него скриптов файл, най-добре е да започва с командата **clear** за изчистване на работното пространство. Така се избягва опасността от дублиране на имена, което би довело до объркващи и непредвидими резултати. Ако сте сигурни, че програмата ви е тествана и работи вярно, а изведнъж започне да дава грешни резултати или съобщения за грешки, се препоръчва да изчистите работното пространство с командата clear и да се изпълни програмата отново.
- При обръщане към скриптов файл не се подават никакви аргументи. Входните данни се въвеждат с помощта на команди, записани в самия файл. Удобен начин е да се организира въвеждане на данните в диалогов режим, като се използва командата **input**.

Пример: Да се състави програма (скрипт файл) за изчисляване на хипотенузата на правоъгълен триъгълник по зададени два катета, като се използва формулата на Питагор $c^2=a^2+b^2$.

Програма:

```
%програма pitagor_script
% пресмятане на хипотенуза
a=input('въведи катет a =');
b=input('въведи катет b =');
c=sqrt(a^2 + b^2);
disp(['хипотенузата е c=', num2str(c)])
```

Програмата е записана като скрипт файл **pitagor_script.m**. За изпълнение на програмата в командния ред се въвежда името на файла.

```
>> pitagor_script
Въведи катет a=3
Въведи катет b=4
хипотенузата е c=5
```

4.2. Файл-функции (`File Functions`)

Всяка файл-функция започва с ключовата дума **`function`**. Основна разлика спрямо скриптов файл е, че при файл-функция обменът на данни се извършва с помощта на списъци от входни и изходни аргументи.

Файл-функция, която описва функция на една променлива от вида $y = f(x)$, има следния общ вид:

```
function y = fname(x)  % заглавен ред на функцията
y = f(x);  % пресмятане на функцията
```

където

- **`function`** е ключовата дума;
- `y` – изходен аргумент (формален);
- *`fname`* – име на функцията - произволно;
- `x` – входен аргумент (формален).

За пресмятане на стойността на функцията няма ограничения в броя на използваните команди (оператори).

Една файл-функция може да има няколко входни и изходни аргумента. В случаите, когато изходните аргументи са повече от един, е задължително списъкът на изходните аргументи да се загради с квадратни скоби, а отделните входни аргументи да се разделят със запетаи. Файл-функция с няколко входни и изходни аргументи има следния общ вид:

```
function [y1, y2, …, ym] = fname(x1, x2, …, xn)
y1 = f1(x1, x2, …, xn);
y2 = f2(x1, x2, …, xn);
...................
ym = fm(x1, x2, …, xn);
```

където

- **`function`** е ключовата дума;
- `y1, y2, …, ym` – изходни аргументи;
- *`fname`* – име на функцията;
- `x1, x2, …, xn` – входни аргументи.

Не трябва да се бърка списъкът с изходни аргументи `[y1, y2, …, ym]` с вектор. При задаване на вектор като разделители между елементите могат да се използват и интервали, което не е допустимо в списъка на аргументите.

Допустимо е една файл-функция да няма нито един входен или изходен аргумент.

Задължително условие е файл-функцията да се записва във файл, който носи нейното име. В противен случай MATLAB няма да може да открие и изпълни тази функция.

Веднъж създадена, файл-функцията се използва по правилата на вградените в MATLAB функции.

Следващите примери демонстрират различни варианти за създаване на файл-функция. Представени са 3 варианта на задачата за пресмятане на хипотенуза по зададени катети, оформени по различен начин като файл-функции.

Пример: **Файл-функция без входни аргументи**

В този случай въвеждането на стойностите за входните данни е включено в самата функция.

```
%програма pitagor_func1
% пресмятане на хипотенуза
% файл-функция без входни аргументи
function c1 = pitagor_func1
a=input('въведи катет a =');
b=input('въведи катет b =');
c1=sqrt(a^2 + b^2);
disp(['хипотенузата е '])
```

Тази файл-функция е съхранена с име **pitagor_func1.m.** Използването на файл-функцията е като тя се присвои на избрана променлива, например:

```
>> c=pitagor_func1
въведи катет a=3
въведи катет b=4
хипотенузата е
c =
     5
```

Пример: Файл-функция с два входни аргументи – стойностите за катетите a и b.

В този случай се пресмята хипотенузата за една двойка стойности за катетите.

```
% пресмятане на хипотенуза
% файл-функция с два входни аргумента
% a и b, стойности за двата катета
function c2 = pitagor_func2(a, b)
c2=sqrt(a^2 + b^2);
disp(['хипотенузата е '])
```

Тази файл-функция е съхранена с име **pitagor_func2.m.** При извикване на функцията се задават фактическите стойности за аргументите a и b, например :

```
>> c=pitagor_func2(3,4)
Хипотенузата е
c =
     5
```

Пример: Файл-функция с която се пресмятат едновременно няколко стойности за хипотенузата за множество двойки стойности за катетите.

В този случай входните аргументи са вектори a1 и b1 със стойностите на двата катета.

```
% пресмятане на хипотенуза
% файл-функция с два входни аргумента –
% вектори a1 и b1 с множество двойки стойности за катетите
% за едновременно пресмятане на множество стойности
% за хипотенузата
function c3 = pitagor_func3(a1, b1)
c3=sqrt(a1.^2 + b1.^2);
disp(['хипотенузата е '])
```

Тази файл-функция е съхранена с име **pitagor_func3.m**. В този случай, преди извикване на функцията е необходимо да се зададат стойности за векторите a1 и b1.

```
>> a1=[3 5 6]
a1 =
     3     5     6

>> b1=[4 6 8]
b1 =
     4     6     8

>> c=pitagor_func3(a1,b1)
Хипотнузата е
c =
    5.0000    7.8102   10.0000
```

Този резултат се разчита така:
За стойности на катетите a=3 и b=4 хипотенузата е c=5.
За стойности на катетите a=5 и b=6 хипотенузата е c=7.8102.
За стойности на катетите a=6 и b=8 хипотенузата е c=10.

4.3. Глобални и локални променливи

Има същесвена разлика в достъпа до различните променливи величини, използвани при скриптовите файлове и файл-функциите.

При работа със Script файлове променливите образуват обща работна област и се съхраняват в обща памет, наречена Workspace. Това на практика означава, че променливите, които се използват в един Script файл, могат да се използват в други такива файлове. Ето защо такива променливи се наричат глобални променливи. По този начин е възможно една сложна програма да се раздели на няколко по-прости програми, всяка от които е записана в отделен файл, които да се извикват последователно от една главна програма. В същото време, обаче, програмистът трябва да се съобразява с имената на използваните до момента променливи, за да не възникнат грешки при използването им в различни програми.

До променливите, които се пресмятат в тялото на една файл-функция и не са включени в списъка на изходните аргументи, има достъп само тази функция. Тези променливи се наричат локални променливи. След излизане от функцията локалните променливи не се запазват.