Contents


```
function Functii
```

Fisiere functie

```
%Daca prima linie a fisierului contine cuvantul “function” atunci fisierul respectiv
%este declarat ca fisier functie. O functie difera de un “script” prin faptul ca poate lucra
%cu  argumente.  Variabilele  definite si  manipulate  in  interiorul fisierului functie  sunt
%localizate  la  nivelul  acesteia.  Prin  urmare,  la  terminarea  executiei  unei  functii,  in
%memoria calculatorului nu raman decat variabilele (parametrii) de iesire ale acesteia.
%Cand un fisier functie este executat/apelat, MATLAB foloseste un spatiu de memorie diferit de Workspace.
%Forma generala a primei linii unui fisier functie este:
%     function [param_iesire1, param_iesire2,....]=nume_functie(param_intrare1, param_intrare2,...)

%Dupa prima linie urmeaza corpul functiei care se va incheia cu end.

%function [param_iesire1, param_iesire2,....]=nume_functie(param_intrare1, param_intrare2,...)
%param_iesire1 = expresie1;
%param_iesire2 = expresie2;
%...
%end

%Observatii: Parametrii de iesire sunt scrisi intre paranteze drepte,
%inaintea operatorului de atribuire.
%In corpul functiei este necesar ca fiecare parametru de iesire sa fie
%atribuit, altfel functia nu va fi utilizabila.

%De regula numele fisierului coincide cu numele functiei, i.e. nume_functie, iar ulterior
```

```
%functia este apelata cu numele functiei.

%O functie poate fi apelata in alte fisiere de tip script sau function, sau
%in Comand Window, unica conditie este ca functia apelata sa fie salvata in
%acelasi dosar cu fisierele pe care acestea o apeleaza, iar dosarul
%respectiv sa figureze ca dosar curent.

%Sintaxa de apelare este la fel cu prima linie din functie, exceptand cuvantul function:
%[param_iesire1, param_iesire2,....]=nume_functie(param_intrare1, param_intrare2,...)
%unde lista parametrilor de intrare a fost  definita in prealabil.
```

Subfunctii

```
%Un fisier functie poate contine mai multe functii definite de utilizator,
%Prima functie s.n. functie primara, iar restul functiilor s.n. functii secundare.
% Numele fisierului functie in care este salvata o astfel de functie
% coincide cu cel al functiei primare.

% Exemplu
%Sa se defineasca functia f(x,y) = (x+y,x-y,x^2) si sa se ecalueze f(2,3).
%Metoda I
    function [f] = fun1(x,y)
        f=[x+y; x-y;x^2];
    end
f = fun1(2,3);
fprintf('Metoda I\n')
fprintf('f(2,3)=');
fprintf('\n %5.2f',f);
fprintf('\n');

%Metoda II
    function [f] = fun2(x)
        f(1) = x(1) + x(2);
        f(2) = x(1) - x(2);
        f(3) = x(1)^2;
    end

f = fun2([2,3]);

fprintf('Metoda II:\n')
fprintf('f(2,3)=');
fprintf('\n %5.2f',f);
fprintf('\n');
```

```
Metoda I
f(2,3)=
  5.00
 -1.00
  4.00
Metoda II:
f(2,3)=
  5.00
 -1.00
  4.00
```

Variabile globale

```
%global VarName1 VarName2 VarName3
%Pentru a da posibilitatea ca mai multe functii sa utilizeze aceleasi variabile,
%trebuie declarate globale, in fiecare functie.

%Variabilelor globale li se pot atribui sau reatribui o valoare in oricare
%dintre locurile in care au fost definite.

 function [f] = fun3(x)
     global a b
     a = 2; b = 3;
     f = a*x^2+b*x;
 end

function [f] = fun4(x)
     global a b
     f = 3*a*x^2+2*b*x;
end
fun3(2); %Se apeleaza functia fun3 pentru actualizarea variabilelor globale a si b
f = fun4(2);
%Daca aceste variabile vor fi modificate in fun3, functia fun4 va prelua valorile modificate

fprintf('Fie functia f(x)  = 3*a*x^2 + 2*b*x\n');
fprintf('Valoarea f(2) in cazul in care a = 2, respectiv b=3 este: \n');
fprintf('f(2)= %5.2f\n',f)
```

```
Fie functia f(x)  = 3*a*x^2 + 2*b*x
Valoarea f(2) in cazul in care a = 2, respectiv b=3 este:
f(2)= 36.00
```

Variabile handle de functii

```
%O variabila tip handle de functii contine toate informatiile necesare despre o functie
%care poate fi apelata mai tarziu. De obicei, o variabila tip handle de functii poate fi argumentul
%unei alte functii. Instructiunea
                                %fh = @NumeFunctie
%memoreaza in variabila fh un handle al functiei NumeFunctie.
%Variabila fh poate fi trecuta ca argument al altei functii.
%Apelarea functiei NumeFunctie  printr-o variabila handle se face astfel
                                %fh(lista de parametri)
%unde lista de parametri contine parametrii functiei  NumeFunctie.
function y = fun5(x)
    y = (x + 1) ^ 2;
end
f = fun5(3); %Apelarea functiei fun5 prin numele functiei
%sau
fh = @fun5; %Definirea variabilei fh de handle
f = fh(3); %Apelarea functiei fun5 prin intermediul variabilei de tip handle

fprintf('Fie functia f(x) = (x+1)^2\n')
fprintf('f(2)=');
fprintf(' %5.2f',f);
fprintf('\n');

function y = fun6(g,x) %Functia fun6 are drept argument functia g
     y = g(x)+x;
end
f = fun6(fh,2); %Apelarea functiei fun6 in care primul argument este o variabile de tip handle

fprintf('Fie functia f(x) = g(x) + x, unde g(x)= (x + 1) ^ 2\n')
fprintf('f(2)=');
fprintf('%5.2f',f);
fprintf('\n');
```

```
Fie functia f(x) = (x+1)^2
f(2)= 16.00
Fie functia f(x) = g(x) + x, unde g(x)= (x + 1) ^ 2
f(2)=11.00
```

Functii anonime

```
%In cazurile in care se doreste evaluarea unei expresii matematice
%simple de mai multe ori in program, se poate utiliza functia anonima (handle).
% O functie anonima (handle) este definita in corpul unui program si nu printr-un fisier de tip functie.

%NumeFunctie = @(var1, var2, . . ., var3) Expresie
fun7=@(x,y) [x+y; x-y;x^2];

f = fun7(2,3);

fprintf('f(2,3)=');
fprintf('\n%5.2f',f);
fprintf('\n');
```

```
f(2,3)=
 5.00
-1.00
 4.00
```

Functia inline

```
%Functia inline este o varianta mai noua a functiei handle
%NumeFunctie = inline ('Expresie', 'var1', 'var2',...,'var3')
fun8=inline('[x+y; x-y;x^2]','x','y');
f = fun8(2,3);

fprintf('f(2,3)=');
fprintf(' \n%5.2f',f);
fprintf('\n');
```

```
f(2,3)=
 5.00
-1.00
 4.00
```

Functii de functii

```
%Functiile definite cu ajutorul sintaxei function pot avea ca parametru de intrare variabile de tip handle.
%Variabila handle se poate defini printr-o functie definita cu function, printr-o functie handle,
```

```
%sau printr-o functie standard Matlab. Astfel, o functie poate avea ca parametri alte func?ii.

    function f = fun9(x,y)
        f = [x+y; x-y;x^2];
    end
    function z = fun10(fun,x,y)
        z = fun(x,y) + [x;y;3];
    end

f = fun10(fun7,1,2) %fun7 si fun8 sunt functii de tip handle (sau variabile de tip handle de functii)
f = fun10(fun8,1,2)
f = fun10(@fun9,1,2) %@fun9  - se trece de la  functia definita cu function la variabila de tip handle
```

```
f =

     4
     1
     4


f =

     4
     1
     4


f =

     4
     1
     4
```

Comanda feval

```
%Matlab are functia standard feval care se utilizeaza pentru evaluarea functiilor
%standard si a celor definite de utilizator. Func?ia feval utilizeaz? o variabila
%tip handle pentru a evalua functiile.
          %feval(handle, var1, var2,..., varn)
%unde handle este variabila handle a functiei sau functia handle ce va fi evaluata
% si var1, var2, ..., varn sunt parametrii de intrare ai functiei ce va fi evaluata
```

```
feval(fun7,1,2)
feval(fun8,1,2)
feval(@fun9,1,2)
```

```
ans =

     3
    -1
     1


ans =

     3
    -1
     1


ans =

     3
    -1
     1
```

```
end
```

Published with MATLAB® R2014a