

Utilizarea fișierelor în Matlab.

1. Obiective urmărite.

- prezentarea funcțiilor de nivel scăzut, tip C pentru realizarea operațiilor I/E cu fișiere (sintaxă, semantică, exemple de utilizare)
- scrierea unor funcții M utile în lucrul cu fișiere

2. Prezentarea funcțiilor și comenzilor pentru operații I/E.

a. Utilizarea comenzilor **save** și **load**.

Fișierele **MAT** (cu extensia **.mat**) sunt fișiere binare, în precizie dublă folosite în Matlab. Cel mai simplu mod de scriere într-un fișier binar se realizează folosind comanda **save**.

>>save

salvează toate variabilele din program (din spațiul de lucru) în fișierul **matlab.mat**

Dacă dorim crearea unui fișier MAT cu alt nume (de exemplu **date.mat**), conținând numai anumite variabile, vom da comanda:

>>save date.mat a, b, c

sau funcția:

>>save ('date.mat', 'a', 'b', 'c')

Datele pot fi salvate, mai puțin efficient în format text, folosind opțiunea **-ascii**:

>>save ('-ascii', 'date.txt', 'a', 'b', 'c') sau

>>save -ascii date.txt a b c

Creați un fișier text care conține un pătrat magic 4 x 4:

>>a=magic(4);

>>save -ascii mag.txt;

Citirea în memorie dintr-un fișier **MAT**, creat cu comanda **save** se face cu **load**:

>>load date.mat a, b, c

sau funcția:

>>load ('date.mat', 'a', 'b', 'c')

Citirea dintr-un fișier text (ascii) se face cu:

>>load ('-ascii', 'date.txt')

>>load mag.txt

Fișierul este încărcat într-o variabilă matrice având numele același cu fișierul: **mag**

b. Deschiderea fișierelor

Înainte citirii și scrierii dintr-un fișier text sau binar acesta trebuie deschis folosind comanda **fopen** cu una din formele:

fid = fopen('numefis', 'mod')

[fid, mesaj]= fopen('numefis', 'mod')

Modul (sau permisiunea) poate fi una din alternativele:

- **r** numai pentru citire

- **w** numai pentru scriere
- **a** numai pentru adăugare
- **r+** atât pentru citire cât și pentru scriere

Dacă operația de deschidere reușește, **fopen** întoarce un întreg nenegativ, numit *identificator de fișier* (**fid**). Valoarea aceasta este transmisă ca argument altor funcții de I/E care accesează fișierul deschis.

Dacă deschiderea fișierului eșuează, întrucât fișierul nu există, **fid** primește valoarea -1. Fișierele standard nu trebuie deschise. Fișierul standard de ieșire are identificatorul **fid=1**, iar fișierul standard de eroare **fid=2**.

Deschideți pentru citire un fișier, al cărui nume îl introduceți de la tastatură. Afișați mesajul care specifică dacă operația de deschidere a reușit sau nu.

```
fid=0;
while fid < 1
    numefis=input('Deschide fisier: ', 's');
    [fid,mesaj] = fopen(numefis, 'r');
    if fid == -1
        disp(mesaj)
    end
end
```

Funcția **input** permite introducerea de date de la tastatură. Șirul de caractere dat ca prim parametru va fi afișat. Al doilea parametru '**s**' arată că datele introduse sunt caractere.

Funcția **disp** afișează un șir de caractere.

c. Citirea datelor binare

Funcția **fread** citește dintr-un fișier binar (specificat prin **fid**) și memorează datele într-o matrice. În forma cea mai simplă, se citește tot fișierul, fiecare octet devenind un element al matricei. Exemplu:

```
>>fid = fopen('date.dat','r');
>>A = fread(fid);
```

Ecoul datelor pe ecran după citire, folosește comanda **disp** pentru a afișa datele sub formă de caractere:

```
>>disp(char(A'))
```

Numărul datelor citite poate fi controlat printr-un parametru contor:

```
>>fid = fopen('date.dat','r');
>>A = fread(fid, 100);
```

Datele citite în vectorul **A** pot fi interpretate în mod diferit: întregi cu sau fără semn, pe 8, 16, 32 sau 64 biți sau reali, pe 32 sau 64 biți, sau chiar un șir de biți, de lungime specificată. În acest scop se folosește un al 3-lea parametru, care poate lua una din valorile: '**schar**', '**uchar**', '**int8**', '**int16**', '**int32**', '**int64**', '**float32**', '**float64**', '**bitN**', sau '**ubitN**', cu **N** între 1 și 64.

```
>>A = fread(fid, 100, 'int16');
```

Formatul de afișare al datelor este specificat sub forma: **'intrare=>iesire'**.

Fișierul 'alfa.txt' conține cele 26 litere majuscule din alfabet. Deschideți fișierul și citiți primele 5 elemente în vectorul a.

```
>>f=fopen('alfa.txt','r');
>>a=fread(f,5)
a =
    65    66    67    68    69
>>fclose(f);
```

Rezultatul întors de **fread** este un vector coloană, pe care îl transpunem, pentru afișarea pe orizontală.. Pentru a vedea rezultatul nu am terminat instrucțiunea cu ;
Dacă specificăm formatul de intrare **'uint8'** și afișare **'char'** obținem:

```
>>f=fopen('alfa.txt','r');
>>a=fread(f,5, 'uint8=>char')      %am avea acelasi efect cu
a =
    ABCDE
>>fclose(f);                        % 'char=>char' sau 'char*'
```

Citiți din fișierul cu literele alfabetului, câte o literă, și apoi săriți peste două litere.

În acest scop vom utiliza și un al 4-lea parametru – de salt, care va indica numărul de caractere peste care se sare după o citire. Operația de citire este reluată până la sfârșitul fișierului.

```
>>f=fopen('alfa.txt','r');
>>a=fread(f, '*char', 2)';
a =
    ADGJMPSVY
>>fclose(f);
```

Am omis cel de-al doilea parametru în **fread**, ceea ce a determinat citirea până la sfârșitul fișierului.

Dacă nu se specifică contorul și nici formatul, atunci se consideră valorile implicite **inf** și **'*char'**.

d. Scrierea datelor binare.

Scrierea datelor din memorie (dintr-o matrice), într-un fișier binar se face cu **fwrite**, cu sintaxa:

```
contor=fwrite(fid, A)
contor=fwrite(fid, A, precizie)
contor=fwrite(fid, A, precizie, salt)
```

1. scrie datele din matricea A în fișierul specificat. Întoarce numărul de elemente scrise.
2. realizează în plus și o conversie de format
3. sare peste numărul de octeți de salt, după care face scrierea

Creați un fișier binar de 100 octeți, conținând cele 25 de elemente ale unui pătrat magic 5x5, memorate ca întregi pe 4 octeți fiecare.

```
fid=fopen('magic5.bin', 'wb');
fwrite(fid, magic(5), 'int*4');
```

e. Scrierea datelor formatate în fișiere text.

contor = fprintf(fid, format, A, ...)

funcția întoarce numărul de octeți transferați.

Descriptorii de format sunt aceiași din C. Există descriptorii specifici Matlab:

%bx – afișare valoare **double** în hexazecimal

%tx – afișare valoare **float** în hexazecimal

Descriptorii pot fi precedați de caracterele: **'-'**, **'+'**, **' '**, **'0'**, cu semnificațiile:

'-' aliniere stânga

'+' afișează întotdeauna cu semn

' ' inserează un spațiu înaintea valorii afișate

'0' pune zerouri în locul spațiilor

*Creați fișierul **expo.txt** cu valorile funcției $f(x) = e^{-x^2}$ pe intervalul **[0, 1]** cu pasul **0.1**.*

```
x=0:0.1:1;  
y=[x; exp(-x*x)];  
fid=fopen('expo.txt','wt');  
fprintf(fid,'%6.2lf%10.5lf\n',y);  
fclose(fid);
```

f. Citirea datelor formatate din fișiere text.

A=fscanf(fid, format);

[A, contor]=fscanf(fid, format, dimensiune);

Prima formă citește până la sfârșitul fișierului. Cea de-a doua formă citește un număr specificat de dimensiune. Aceasta se specifică sub una din formele:

n – cel mult **n** numere, caractere sau șiruri

inf – până la sfârșitul fișierului

[m,n] - cel mult **m*n** valori, care completează o matrice pe coloane

Funcția **fscanf** este vectorizată și întoarce un argument matrice. Funcția acceptă și valorile **'-inf'**, **'+inf'**, **'NaN'**, pe care le convertește în reprezentările numerice corespunzătoare.

g. Controlul poziției în fișier.

Funcția **fseek** ne permite să ne poziționăm oriunde în fișier

stare = fseek(fid,deplasare,origine)

Originea ia una din valorile:

'bof' – față de începutul fișierului

'cof' – față de poziția curentă

'eof' – față de sfârșitul fișierului

Deplasarea este o valoare pozitivă sau negativă exprimată în octeți și raportată la origine.

Funcția **ftell** determină poziția curentă în fișier, față de începutul fișierului:

```
pozitie = ftell(fid)
```

h.Citirea unei linii din fișierul text.

h.1. până la sfârșitul de linie

```
linie = fgetl(fid)
```

dacă se întâlnește **eof** funcția întoarce -1.

Citiți un fișier text linie cu linie și afișați-l pe ecran.

```
fid=fopen('text.m');  
while 1  
    linie = fgetl(fid);  
    if ~ischar(linie), break, end  
    disp(linie)  
end  
fclose(fid);
```

h.2. inclusiv sfârșitul de linie

```
linie = fgets(fid)  
linie = fgets(fid, nrcar)
```

3. Probleme propuse.

- 3.1. Definiți o funcție, fișier M pentru determinarea dimensiunii unui fișier exprimată în octeți. Funcția are ca parametru numele fișierului.
- 3.2. Definiți o funcție care citește date numerice dintr-un fișier text, delimitate prin virgule.
- 3.3. Definiți o funcție care citește un fișier text, linie cu linie și întoarce numărul total de apariții ale unui anumit șir de caractere în fișier. Funcția va afișa fiecare linie din fișier, precedată de numărul de apariții ale șirului în linie.

Funcția are semnătura: `function y = contor(numefis, sir)`