

TP N° 4 : Outils de développement

I-Scripts et fonctions

I-1-Scripts

Il est possible de stocker une séquence d'instructions Matlab dans un fichier texte, appelé script ou bien M-File, afin de les utiliser plus tard. Un tel script est tout simplement un fichier texte qui doit **obligatoirement** avoir l'extension « .m », d'où le nom M-file.

Un script est un ensemble d'instructions MATLAB (qui joue le rôle de programme principal) identifiées par un nom, il est réutilisable à chaque fois qu'on fait appel à son nom (exécution du script). Si le script est écrit dans le fichier de nom « test.m » on l'exécute dans la fenêtre command Window de MATLAB en tapant « test ». Les scripts sont très utiles, il est en effet beaucoup plus simple de modifier des instructions dans un fichier à l'aide d'un éditeur de texte que de retaper un ensemble d'instructions MATLAB dans la fenêtre de commande. Les scripts doivent contenir des commandes de Matlab tels que on les utilise dans command Window.

Vous pouvez écrire votre script en utilisant un simple éditeur de texte comme (le bloc note de Windows à titre d'exemple), mais pour l'exécution de votre script vous avez certainement besoin de Matlab. Ce dernier nous offre également un éditeur de texte dédié spécialement pour l'écriture des scripts et des fonctions.

Pour utiliser l'éditeur de script de Matlab, cliquez sur le bouton New script en haut à gauche de la fenêtre Matlab.

Exemple

1. Création du script

Démarrer l'éditeur de script de Matlab et écrivez les lignes de codes suivantes :

```
R=3;  
A = 4*pi*R^2;  
V = 4/3*pi*R^3;
```

2. Enregistrement du script

Cliquez sur le bouton Save ou bien sur la petite flèche en-dessous du bouton Save et cliquez sur le bouton Save As. Sauvegardez votre travail dans un dossier de votre choix.

3. Exécution du script

Pour exécuter votre script il suffit de taper son nom dans la ligne de commande (>>).

NB. : Avant d'exécuter votre script il faut s'assurer que le script est dans le current folder. Dans le cas contraire il faut modifier le current folder.

Matlab exécute alors l'intégralité de votre script. Ainsi, dans la zone des variables, vous verrez trois nouvelles variables apparaître : R, A et V que vous créez dans le script.

Exercices d'application

Exercice 1

Écrire un script qui calcule et affiche le diamètre, la circonférence et la surface d'un cercle de rayon 20

Solution

```
r=20;
d=2*r;
p=2*pi*r;
s=pi*r^2;
disp(['un cercle de rayon ' num2str(r) ' a pour diametre ' num2str(d) ',pour
circonference ' num2str(p) 'et pour surface ' num2str(s) '.'])
%pour afficher le résultat vous pouvez également utiliser la fonction fprintf
fprintf('un cercle de rayon %d a pour diametre %d pour circonference %.2f et
pour surface %.2f.\n',r,d,p,s)
```

Exercice 2

Modifier le script précédant afin qu'il permet à l'utilisateur d'entrer le rayon

Solution

```
r= input('entrer le rayon du cercle:');
d=2*r;
p=2*pi*r;
s=pi*r^2;
disp(['un cercle de rayon ' num2str(r) ' a pour diametre ' num2str(d) ',pour
circonference ' num2str(p) 'et pour surface ' num2str(s) '.'])
%pour afficher le résultat vous pouvez également utiliser la fonction fprintf
fprintf('un cercle de rayon %d a pour diametre %d pour circonference %.2f et
pour surface %.2f.\n',r,d,p,s)
```

Exercice 3

Ecrire un script qui demande deux valeurs a et b à l'utilisateur et qui les affiche, ensuite il permute leurs contenus et les affiche à nouveau.

Solution :

```
a = input('entrez la valeur de a : ');
b = input('entrez la valeur de b : ');
disp(['les valeurs avant permutation sont : a = ' a ' ; b = ' b]);
c = a;
a = b;
b = a;
disp(['les valeurs avant permutation sont : a = ' num2str(a) ' ; b = ' num2str(b)]);
```

Exercice 4

Ecrire un script qui reçoit trois nombres et affiche leur la moyenne

Solution :

```
a=input('1er nombre:');
b=input('2eme nombre:');
c=input('3eme nombre:');
m = (a+b+c)/3
disp(['la moyenne est ', num2str(m)]);
```

Ou bien :

```
a=input('1er nombre:');
b=input('2eme nombre:');
c=input('3eme nombre:');
fprintf('la moyenne est %.2f\n',(a+b+c)/3);
```

Exercice 5

Ecrire un script qui reçoit un nombre et affiche son factoriel

Solution

```
n = input('entrer un nombre') ;
fac= prod(1:n) ;
disp(['le factoriel de ' num2str(n) ' est ' num2str(fac)])
```

I-2-Fonctions

Une fonction est un script qui commence par le mot clé function

Exemple

Ecrire dans un script de code suivant et enregistrer le sous le nom `sommeDeDeuxNombre.m`

```
function res = sommeDeDeuxNombre(a,b)
    res = a+b;
end
```

utiliser alors la fonction `sommeDeDeuxNombre()` pour calculer la somme de deux nombre

NB : Nous allons traiter les fonctions en détail dans une série indépendante

II - Structures itératives

II-1- Boucle for

Permet de répéter un bloc d'instruction un nombre de fois

Syntaxe

```
for index = values
    statements
end
```

Exemple

```
% afficher les éléments d'un vecteur
for v = [1 5 8 17]
    disp(v)
end
```

```
% afficher les colonnes d'une matrice
for I = eye(4,3)
    disp('Current unit vector:')
    disp(I)
end
```

II-2- Boucle while

Permet de répéter un bloc d'instruction tant qu'une condition est vraie

Syntaxe

```
while expression
    statements
end
```

Exemple

Utiliser la boucle while pour calculer le factoriel de 10.

```
n = 10;
f = n;
while n > 1
    n = n-1;
    f = f*n;
end
disp(['n! = ' num2str(f)])
```

II-3- Exercices

Exercice 1

Ecrire un script qui demande deux entiers a et b et qui affiche le résultat de la somme suivante :

$$S = \sum_{k=1}^b k^a$$

Réponse :

```
a = input('entrez a : ');
b = input('entrez b : ');
S=0;
for k=1:b
    S=S+k^a;
end
disp(S)
```

Exercice 2

Ecrire un script qui calcule le 1000^{ème} terme de la suite de Fibonacci

$$u_0 = 0$$

$$u_1 = 1$$

$$u_{n+2} = u_{n+1} + u_n$$

Réponse :

```
u0=0;
u1=1;
for k=1:998
    u2 = u1 + u0; % calcul de u_n+2
    u0 = u1; % préparation des variables pour l'étape du calcul
    u1 = u2;
end
disp(u2); % u2 contient en fait à ce moment u_1000
```

Exercice 3

Ecrire un script qui calcule le 100^{ème} terme de la suite suivante :

$$u_0 = 2$$

$$u_1 = 1$$

$$u_2 = 3$$

$$u_{n+3} = 2u_{n+2} + 3u_{n+1} + u_n$$

Réponse :

```
u0=2; % terme representant u_n
u1=1; % terme representant u_n+1
u2=3; % terme representant u_n+2
for k=1:99
    u3 = 2*u2+3*u1 + u0; % calcul de u_n+3
    u0 = u1; % preparation des variables pour l'étape du calcul suivante
    u1 = u2; %
    u2 = u3;
end
disp(u3)
```

Exercice 4

En utilisant la boucle while, écrire un script qui calcule la plus petite puissance de 7 supérieure à mille

Réponse :

```
x = 1;
while x < 1000
    x = 7*x;
end
disp(x)
```

Exercice 5

Ecrire un script qui demande de l'utilisateur un nombre n et affiche le plus grand cube inférieur ou égal à n.

Réponse :

```
n = input ('entrer un entier n = ');
i=0;
while i^3<=n
    i=i+1;
end
r=(i-1)^3;
disp(r)
```

Exercice 6

Ecrire un script qui demande de l'utilisateur un entier n et calcul le plus grand entier k vérifiant

$$\sum_{i=1}^k i^2 \leq n$$

Réponse :

```
n = input ('entrer un entier n = ');
i=0;
S=0;
while S<=n
    i=i+1;
    S=S+i^2;
end
r=i-1;
disp(r)
```

III - Structures conditionnelles

III-1-Instruction conditionnelle if ... elseif ... else

Exécute un bloc d'instructions si la condition est vrai

Syntaxe

```
if expression
    statements
elseif expression
    statements
else
    statements
end
```

Exemple :

Ce script demande de l'utilisateur deux nombres x et y, les compares et affiche l'expression « x est inférieur à y » dans le cas où x est égale à y.

```
x= input ('entrer la valeur de x = ');
y= input ('entrer la valeur de y = ');
if x<y
    disp(' x est inferieur à y ')
end
```

Dans le cas où x et y sont différents, le script ne va pas réagir. Par contre le script suivant va afficher le message « x est supérieur ou égale à y » :

```
x= input ('entrer la valeur de x = ');
y= input ('entrer la valeur de y = ');
if x<y
    disp(' x est inferieur à y ')
end
```

```

else
    disp('x est superieur ou egale à y')
end

```

si en plus nous voulons distinguer le cas où x égale à y, il faut ajouter un bloc elseif

```

x= input ('entrer la valeur de x = ');
y= input ('entrer la valeur de y = ');
if x<y
    disp(' x est inferieur ? y ')
elseif x==y
    disp('x egale ? y')
else
    disp('x est superieur ou egale ? y')
end

```

III-2-Instruction switch ... case ... otherwise

Exécuter l'un des nombreux groupes d'instructions

Syntaxe

```

switch switch_expression
    case case_expression
        statements
    case case_expression
        statements
    ...
    otherwise
        statements
end

```

Exemple :

Le script suivant demande un nombre n, selon la valeur de n il affiche l'une des expressions : 'negative one', 'zero', 'positive one' ou 'other value'.

```

n = input('Enter a number: ');
switch n
    case -1
        disp('negative one')
    case 0
        disp('zero')
    case 1
        disp('positive one')
    otherwise
        disp('other value')
end

```

Exercices

Exercice 1

Ecrire un script qui reçoit trois nombres et affiche le plus petit

Solution :

```

a=input('1er nombre:');
b=input('2eme nombre:');
c=input('3eme nombre:');
if a>b
    if b>c
        disp(['min=' num2str(c)])
    else
        disp(['min=' num2str(b)])
    end
else
    if a>c

```

```

        disp(['min=' nub2str(c)])
    else
        disp(['min=' nub2str(a)])
    end
end
end

```

ou bien

```

a=input('1er nombre:');
b=input('2eme nombre:');
c=input('3eme nombre:');
fprintf('min=');
if a>b
    if b>c
        fprintf('%d',c)
    else
        fprintf('%d',b)
    end
else
    if a>c
        fprintf('%d',c)
    else
        fprintf('%d',a)
    end
end
fprintf('\n');

```

Exercice 2

Ecrire un script qui reçoit trois nombres et affiche le plus grand

Solution :

A faire

Exercice 3

Ecrire un script qui exécute l'un des deux scripts des exercices 1 et 2, selon le choix de l'utilisateur

```

disp('choisir une op?ration :')
disp('1-min')
disp('2-max :')
a=input('entrer votre choix : ');
switch a
    case 1
        run exercice1.m
    case 2
        run exercice2.m
    otherwise
        disp('Erreur !!!!')
end

```

voici une autre solution qui permet de créer un menu

```

a=menu('choisir une opération', 'min', 'max');
switch a
    case 1
        run exercice1.m
    case 2
        run exercice2.m

    otherwise
        disp('Erreur !!!!')
end

```

Exercice 4

Ecrire un script qui reçoit trois nombres a, b et c et résoudre et affiche la solution de l'équation : $ax^2 + bx + c = 0$

Solution (sans utiliser la fonction roots)

```
a=input('a=');
b=input('b=');
c=input('c=');
if a==0
    if b==0
        if c==0
            fprintf('solution=R')
        else
            fprintf('pas de solution\n')
        end
    else
        fprintf('x=%.2f',-c/b)
    end
else
    d=b^2-4*a*c;
    if d>0
        fprintf('x1=%.2f x2=%.2f\n', (-b-sqrt(d))/(2*a), (-b+sqrt(d))/(2*a))
    elseif d==0
        fprintf('x=%.2f \n', (-b)/(2*a))
    else
        fprintf('pas de solution\n')
    end
end
end
```

Solution avec roots

```
a=input('a=');
b=input('b=');
c=input('c=');
p=[a b c];
r=roots(p);
disp(r)
```

Exercice 5

Le jeu dichotomie est un jeu qui se joue à deux joueurs. Le premier joueur choisit un nombre entre 1 et 100, et le second joueur essaie de deviner ce nombre. À chaque essai du second joueur, le premier indique si le nombre est plus grand ou plus petit que le nombre à deviner (ou égal, et dans ce cas le jeu est terminé).

Ecrire un script qui permet de jouer à dichotomie contre l'ordinateur (le script choisit un nombre, et le joueur essaie de le deviner).

Réponse :

```
k=randi(100);
a=k-1;
while a~=k
    a=input('Entrez un nombre :');
    if a>k
        disp('Le nombre est plus grand');
    elseif k>a
        disp('Le nombre est plus petit');
    end
end
disp('Vous avez gagné');
```

Exercice 6

Parce que les étudiants attendent avec impatience le weekend, on souhaite écrire un programme qui reçoit le nom du jour actuel et renvoie une expression de type :

« Il reste Encore XX jours avant le weekend »

Indication : utiliser l'instruction switch

Réponse :

```
% Un exemple d'utilisation de switch
jour = input('Quel jour sommes nous ? ','s');
switch jour
case 'lundi'
jr = 5;
case 'mardi'
jr = 4;
case 'mercredi'
jr = 3;
case 'jeudi'
jr = 2;
case 'vendredi'
jr = 1;
otherwise
jr = 0;
end
disp(['Encore ' num2str(jr) ' jours avant le weekend' ])
```

Exercice 7

Nous voulons implémenter un script qui permet à l'utilisateur d'apprendre le tableau de multiplication. Le script demande à l'utilisateur d'entrer un entier n entre 1 et 9 en suite il affiche le tableau de multiplication de n : ligne par ligne (une seule ligne sera affichée à l'écran) avec une pause de 5 seconde à chaque nouveau affichage

Indication : les fonctions input et disp pour la communication avec l'utilisateur, la boucle for pour générer le tableau de multiplication et la fonction pause pour la gestion de pause

Exercice 8

Soit M, la matrice définie par :

12	24	36	58
2	6	8	10
7	9	11	13

← La sous matrice M1

Ecrire un script Matlab qui permet d'extraire :

- L'élément situé dans la troisième ligne et la quatrième colonne.
- Le vecteur L3 contenant tous les éléments de la troisième ligne.
- Le vecteur C4 contenant tous les éléments de la quatrième colonne.
- La sous matrice M1 du matrice M, constitué à partir de l'intersection de la deuxième et la troisième colonne avec la deuxième et la troisième ligne.
- La matrice M2 dont ses lignes sont celles de la matrice M mais afficher en sens inverse.

$$M2 = \begin{pmatrix} 7 & 9 & 11 & 13 \\ 2 & 6 & 8 & 10 \\ 12 & 24 & 36 & 58 \end{pmatrix}$$

- Mettre tous les éléments de la deuxième ligne égaux à deux.
- Créer une matrice MM par duplication de la matrice M 3fois pour les lignes et 2 fois pour les colonnes
- Extraire la matrice MMM a partir de la matrice MM en prenant une ligne et laissant l'autre