TP 3
INFORMATIQUE 3

Objectif : Dans ce TP, nous verrons l'utilisation des boucles (***for***, ***while*** *et* ***if***) avec Matlab, et nous écrirons des programmes plus complexes pour avoir des comportements plus intéressant.

1. La boucle *for* dans Matlab

1.1 La boucle *for* sur les vecteurs

Pour commencer, nous allons nous familiariser avec la boucle ***for***. Commencez par écrire un programme très simple qui parcourt un vecteur ligne :

```
v = rand(1, 20);
disp(v);
for i=v
        disp('La variable i vaut ');
        disp(i);
end
```

- Exécuter le programme ci-dessus.
- Que fait le code ? Quel est le rôle de « rand » et « disp »

Questions

Répondez aux questions dans le compte –rendu.

- Comment faire pour afficher les éléments de v du dernier au premier ?
- Indice : regardez la fonction flip.
- Comment faire pour n'afficher que les éléments de v d'indice pair ?
- Nous aimerions modifier l'affichage du précédent programme afin qu'il n'affiche pas le message
- La variable i vaut, mais plutôt Voici ce que contient la case On veut donc `a chaque tour afficher le numéro de la case et son contenu.

1.2 La boucle *for* sur les matrices

A. Exécuter le programmé suivant :

```
for i=1:5;
for j=1:5;
M(i,j)=i+j;
end;
end;
```

- Que fait le code ?

B. Exécuter le programmé suivant :

```
A = rand(5, 5);
disp(A);
for i=A
        disp('La variable i vaut
        ');disp(i);
end
```

- Ce programme affiche-t-il tous les éléments de A un par un ? Que fait, donc ?

Questions

Répondez aux questions dans le compte –rendu.

- Comment faire pour afficher tous les éléments de A un par un `a l'aide de deux boucles for ?
- Comment faire pour afficher tous les éléments de A un par un `a l'aide d'une seule boucle for ? Comment contrôler si on souhaite afficher d'abord les éléments par colonne, ou d'abord ceux par ligne ?

  Indice : Comment faire pour transformer la matrice A en un vecteur ?

1.3 Les performances de la boucle *for*

A. Parcours avec for

Nous allons essayer de réaliser la somme pondéré des ẽléments de deux vecteurs de même taille. On commence par d'déclarer les deux vecteurs (v,w) et une variable (s) qui servira au calcul :

```
v = and(1,100000);
w = rand(1,100000);
s=0;
```

Ensuite, on utilise une boucle ***for*** afin de parcourir les deux vecteurs, et un indice pour parcourirl'autre :

```
i=1;
for k = v
        s = s + k*w(i);
        i=i+1;
end
```

- Créer un code calculant la somme pondéré des ẽléments de deux vecteurs.
- A l'aide des mots clefs ***tic*** et ***toc***, mesurez le temps d'exécution de ce programme ? La valeur de (s) que vous obtenez est-elle en accord avec le fait que la fonction rand réaliser une distribution uniforme entre 0 et 1 ?

- Maintenant, calculant la somme en utilisant le code suivant :

```
for k = [v;w]
        s = s + k(1)*k(2);
end
```

- Mesurez le temps d'exécution de ce dernier, et que remarquer vous ?

B. Préallocation

Nous allons chercher à calculer, étant donné $a \in \mathrm{R}^+$, les termes de la suite :

$$u_0 = 1$$
$$u_n = 0.5(u_{n-1} + a/u_{n-1})$$

Pour ce faire, nous proposons ce code :

```
a=16;
lim = 1000000;

un = 1;
for k = [2:lim]
        un_1 = un;
        un = 0.5*(un_1 + a/un_1);
end
```

- Exécuter le programme. Quel est le temps de calcul de votre programme ?
- Nous souhaitons stocker les valeurs successives de la suite dans un vecteur. Nous modifions ainsi notre code :

```
tic;
a=16;
lim = 1000000;
h(1) = 1;

for k = [2:lim]
        h(k) = 0.5*(h(k-1) + a/h(k-1));
end
toc;
```

Questions

Répondez aux questions dans le compte –rendu.

- La toute première fois que vous exécutez ce code, va-t-il plus vite ou moins vite que l'autre version du code ? L'´ecart est-il important ?
- Si vous exécutez ce code une seconde fois, est-il plus rapide ?
- Afin de comprendre pourquoi ce code était bien moins rapide la première fois, regardez ce qu'il se passe pour le vecteur (h) à chaque tour de boucle (quelle sera sa taille) ? Ceci explique-t-il pourquoi, la seconde fois, votre code était plus rapide ?
- Pour éviter cela, comment allouer d`es le d´épart une certaine taille au vecteur h

? Après avoir supprimé la variable h de la zone des variables, et avoir effectué les modifications n´nécessaires

- à votre code, testez votre nouvelle version : s'exécute-t-telle plus vite ?

2. La boucle *while* dans Matlab

La boucle ***while*** de Matlab permet de répéter un morceau de code tant qu'une condition n'est pas satisfaite.

A. Tester le code suivant :

```
a = input('Entrez un nombre positif : ');
while a<0
    a = input('Entrez un nombre positif : ');
end

disp(a);
```

- Que fait ce code ?
- Que fait donc, *while* ?

B. Testez ce code :

```
a=rand(1,1);
b=1;

while a<30
    b = b+1;
end
```

Si l'exécution de ce code vous parait longue, c'est normal : le code de la boucle ***while*** ne fait pas ẽvoluer la condition qui restera pour toujours vraie. La boucle va donc tourner à l' infini. Pour interrompre votre programme, faîtes $CTRL+C$ dans la fenêtre de commande. Il est donc très important, dans une boucle ***while***, que le code de la boucle fasse ẽvoluer la condition.

C. Exécuter le code suivant :

```
n=0;
while (n < 10)
n.^2
n=n+1;
end
```

- Que fait le code ?

3. La boucle *if* dans Matlab

La boucle ***if*** permet de test si une condition est vérifiée afin de réaliser une action quelconque. Pour bien comprendre son fonctionnement ainsi que sa syntaxe réalisez sous Matlab l'exemple ci-dessous.

```
for i=1:5;
    for j=1:5;
      M(i,j)=i+j;
      if i>2
         if j>2
            M(i,j)=0;
         end
      end
    end
end
```

- Que fait le programme ?

Dans une boucle ***if*** complète, on implémente souvent la ***elseif*** et la ***else*** comme présente l'exemple suivant :

```
a = randi(100, 1);
if a < 30
   disp('small')
elseif a < 80
   disp('medium')
else
   disp('large')
end
```

- Que fait le code ?