TP5- MATLAB Scripts and Functions

Part A. MATLAB scripts: In order to be able to reuse the calculation lines, it is useful to put them in a script. A script is a text file that MATLAB can read and execute. To access it: (1) Open the MATLAB script editor either by clicking on the blank page in the toolbar, or by going to the "File/New/M-file" menu.

Exercise 1: Conditional Statements

1. Write the MATLAB script that requests two values x and y from the user and displays them, swaps their contents, and displays them again.
2. Write the MATLAB script that asks for a number and then displays its sign (positive, negative, or zero).

Correction:

1.

```
disp('Donner x') ;
x = input('entrez x : ') ;
disp('Donner y') ;
y = input('entrez y : ') ;
z=x;
x=y;
y=z;
x, y
```

2.

```
x=input('donner x: ')
if x<0
disp('nombre negatif')
elseif x>0
disp('nombre positif')
else
disp('nombre null')
end
```

Exercise 2: Repeating instructions (loops)

1. Write a program that requires two integers a and b and displays the result of the following sum:

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

2. Write a MATLAB program that takes as input a real $x \in]0.20[$, gives an error message if $x \notin]0.20[$, otherwise calculates and outputs the smallest n such that the sum is greater than x.

$$S_n = 1 + \frac{1}{2} + \frac{1}{3} + \cdots + \frac{1}{n}$$

3. Write a program that calculates the 10th term of the Fibonacci sequence:

$$u_0 = 0, u_1 = 1, u_{n+2} = u_{n+1} + u_n$$

Correction :

1.

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

s
2.
```
x=input('donner x');
if (x<=0)|(x>=20)
error('x non conforme')
else
s=0;n=0;
while s<=x
n=n+1;
s=s+(1/n);
end
end
n
```
3.
```
u0=0; % terme representant u_n
u1=1; % terme representant u_n+1
for k=1:10
u2 = u1 + u0;
u0 = u1;
u1 = u2;
end
disp(u0);
```

Part B. MATLAB Functions:
There are many predefined functions in MATLAB, but there will inevitably come a time when you want to use a function that is not defined. Fortunately, it is possible to define your own functions and use them exactly like pre-existing functions.

Exercise 3:

1. Write a function $pair$ that can tell whether an integer x is even or not;
2. Write the $fonction\ somme$ which calculates the sum of two matrices A and B.
3. Write the $fonction\ produit$ which calculates the product of two matrices A and B.

$$c_{i,j} = \sum_{k=1}^{n} a_{i,k} b_{k,j}$$

Correction :

1.
```
function r=pair(x)
if mod(x,2)==0
r='vrai';
else
r='faux';
end
end
```
2.
```
function C=sommematriciel (A,B)
[p,n]=size(A);[m,q]=size(B);
if (p==m)&(n=q)
C=zeros(p,n);
for i=1:p
for j=1:n
C(i,j)=A(i,j)+B(i,j);
end
end
else
error('taille de matrices incompatibles')
```

```
end
end
```
3.
```
function C=produitmatriciel (A,B)
[p,n]=size(A);[m,q]=size(B);
if n==m
C=zeros(p,q);
forli=1:p
for co=1:q
for k=1:n
C(li,co)=C(li,co)+A(li,k)*B(k,co);
end
end
end
else
error('taille de matrices incompatibles')
end
end
```
Exercise 4:

Let the following calculation function have two nested loops:

```
function M=calcul(M)
[n,m]=size(M);
for i=1:n
v=M(i,:);
for j=1:m
M(i,j)=v(m-j+1)
end
end
```

1. Give the value of B after executing the following instructions:
   ```
   >> A=[1 2 3 4;5 6 7 8 ;9 10 11 12];
   >> B=calcul(A)
   ```
2. Derive what this function does.
3. Rewrite the calculation function to obtain the same result using a single loop.

Correction :

1.
```
B =
4 3 2 1
8 7 6 5
12 11 10 9
```

2.
Renversement des lignes de la matrice.

```
function M=calcul1(M)
m=size(M,2);
N=M;
for j=1:m
M(:,j)=N(:,m-j+1)
End
end
```

Part C. Using functions and scripts together:
In case of complex and difficult to solve programs use functions with the MATLAB program.

Exercise 5 :

1. Write a MATLAB product function that takes two square matrices of the same dimensions A and B as argument and calculates the product $A \times B$, then displays the result to matrix C.

2. Write the MATLAB script which allows you to enter two square matrices of the same dimensions A and B, checks if their size is compatible. If they are not, gives an error message and no output, if they are, calculates $AB+BA$. The script calls the product function.

NB: We refrain from using operations of the type $A \times B$ or $A + B$, we must instead use an assignment element by element.

Correction :

1.

```
function C=produitmatriciel (A,B)
[p,n]=size(A);
C=zeros(p,n);
for i=1:p
for j=1:n
for k=1:n
C(i,j)=C(i,j)+A(i,k)*B(k,j);
end
end
end
end
```

2.

```
n1=input('donner nb ligne A'); %nombre de lignes de A
p1=input('donner nb colonnes A'); %nombre de colonnes de A
n2=input('donner nb ligne B'); %nombre de lignes de B
p2=input('donner nb colonnes B'); %nombre de colonnes de B
if (n1==p1 & n2==p2 & n1==n2) %A et B sont carrées et nombre de colonnes de A doit égale au
nombre de lignes de B
for i=1:n1 %remplissage de A
for j=1:p1
A(i,j)=input('donner une valeur A: ');
end
end
for i=1:n2 %remplissage de B
for j=1:p2
B(i,j) =input('donner une valeur B: ');
end
```

```
end
C1=produitmatriciel(A,B) %appel de la fonction produitmatriciel qui donne A*B
C2=produitmatriciel(B,A) %appel de la fonction produitmatriciel qui donne B*A
for i=1:n1 %calcul de A*B-B*A
for j=1:p1
C3(i,j)=C1(i,j)+C2(i,j)
end
end
else error('dimension non compatibles'); %affichage d'un message d'erreur
end
end
```

Exercise 6:

1. Write a MATLAB "fact" function that takes a positive integer $\boldsymbol{n}$ as an argument and returns $\boldsymbol{n}$! as an answer. Knowing that $n!=\mathbf{1}\times\mathbf{2}\times\mathbf{3}\times\ldots\times\boldsymbol{n}$.

2. Write a MATLAB script which allows you to read two numbers $\boldsymbol{n}$ and $\boldsymbol{p}$ and which calculates and displays

$$\boldsymbol{n!/(p! * (n - p)!)}$$

Correction:

1.

```
function r=fact(n);
r =1;
for i=2:n
r= r * i;
end
end
```

2.

```
n=input('donner n');
p=input('donner p');
if n<0 | p<0 error('nombres negatifs');
else
X=fact(n)/(fact(p)*fact(n-p));
end
disp(X);
```