ECS1.1 - Correction du TP N ° 4

Exercice 4

```
function[]=saisie()   // ni de variable d'entrée ni de variable de sortie
    A=input('donner une matrice de taille 3x3');
    [n,p]=size(A);
    while n~=3 | p~=3
                A=input('Taille incorrecte. Recommencez la saisie : ');
                [n,p]=size(A);             // ne pas oublier de recalculer n et p
    end
    disp('Saisie terminée')
endfunction
```

On a besoin des variables n et p car le test [3,3]=size(A) ne permet pas de vérifier que A est bien de taille 3×3 (mais alors que fait-il) ?

Exercice 5

```
function[T]=pascal(n)          // la variable de sortie T sera affichée
    T=zeros(n,n);             // matrice carrée d'ordre n nulle
    T(:,1)=ones(n,1)             // la première colonne de T est remplie de 1
    for i=2:n do
        for j=2:n do
             T(i,j)=T(i-1,j-1)+T(i-1,j);  // formule de Pascal
        end
    end
endfunction
```

Exercice 6

1.

```
function[x]=max(a,b)          // la variable de sortie x sera affichée
if a>b then x=a;
       else x=b;
       end
endfunction
```

2.

```
function[x]=maxvec(V)          // la variable de sortie x sera affichée
    x=V(1);
    n=length(V);
    for k=2:n do                     // on parcourt le vecteur et on
           if V(k)>x then x=V(k);   // enregistre la plus grand valeur trouvée
           end
    end
endfunction
```

Pour trouver le maximum de (15732) on exécute l'instruction `max([1,5,7,3,2])`.

3. L'idée est que si on trouve le maximum de chaque ligne, il suffit ensuite de trouver le maximum de ces maximums.

```
function[x]=maxmat(A)         // la variable de sortie x sera affichée
    [n,p]=size(A);
    maxtemp=zeros(n,1);  // vecteur qui stockera les maximums par ligne
    for k=1:n do
           maxtemp(k)=maxvec(A(k,:)); // maximum de la ligne k
    end
    x=maxvec(maxtemp);  // maximum des maximums par ligne
endfunction
```

4.

```
function[s]=moyvec(V)         // la variable de sortie m sera affichée
    n=length(V);
    s=0;  // intialisation de la variable qui stockera la somme des coeffs de V
    for k=1:n do
            s=s+V(k); // calcul de la somme des coefficients de V
    end
    s=s/n;  // moyenne des coefficients de V
endfunction
```

Remarquez que ce programme fonctionne indifféremment avec une matrice ligne ou une matrice colonne.

5.

```
function[i]=indicmoymax(A)         // la variable de sortie i sera affichée
    [n,p]=size(A);
    moytemp=zeros(1,p);   // initialisation de la variable qui stockera les moyennes
    for k=1:p do
          moytemp(k)=moyvec(A(:,k));  // moyenne de la colonne k
    end
    maxi=moytemp(1);
    i=1;
    for k=2:n do
          if moytemp(k)>maxi then
                      maxi=moytemp; // recherche de la plus grande moyenne
                      i=k;      // stockage du numéro de la colonne correspondante
          end
    end
endfunction
```

6.Nous donnons deux exemples d'algorithmes de tri.

Le "tri à bulle" consiste comparer les coefficients consécutifs à les mettre à chaque fois dans l'ordre croissant, ce qui déplace le plus grand élément à la fin de la matrice ligne. En réitérant on place le second plus grand élément en avant dernière position, et etc...

```
function[V]=tribulle(V)         // la variable de sortie V sera affichée
    n=length(V);
    for l=1:n-1 do     // on recommence n-1 fois l'algorithme
          for k=1:(n-1) do                  // algorithme qui met le plus grand coeff
                if V(k)>V(k+1) then         // en dernière position
                      temp=V(k+1);  // échange de V(k) et V(k+1) si V(k)>V(k+1)
                      V(k+1)=V(k);
                      V(k)=temp;
                end
          end
    end
endfunction
```

Exemple étape par étape pour la matrice ligne (5 1 4 2 8) :

Etape 1 : (**5 1** 4 2 8) $\longrightarrow$ (1 **5 4** 2 8) $\longrightarrow$ (1 4 **5 2** 8) $\longrightarrow$ (1 4 2 **5 8**) $\longrightarrow$ (1 4 2 5 8)
Etape 2 : (**1 4** 2 5 8) $\longrightarrow$ (1 **4 2** 5 8) $\longrightarrow$ (1 2 **4 5** 8) $\longrightarrow$ (1 2 4 **5 8**) $\longrightarrow$ (1 2 4 5 8)
Etape 3 : (**1 2** 4 5 8) $\longrightarrow$ (1 **2 4** 5 8) $\longrightarrow$ (1 2 **4 5** 8) $\longrightarrow$ (1 2 4 **5 8**) $\longrightarrow$ (1 2 4 5 8)
Etape 4 : (**1 2** 4 5 8) $\longrightarrow$ (1 **2 4** 5 8) $\longrightarrow$ (1 2 **4 5** 8) $\longrightarrow$ (1 2 4 **5 8**) $\longrightarrow$ (1 2 4 5 8)

On voit que la matrice est triée dès le début de la seconde étape mais l'algorithme ne le sait pas...

Le "tri par sélection" consiste à déplacer le plus petit élément en première position, on réitère avec les second plus petit élément qu'on place en seconde position et etc...

```
function[V]=triselection(V)          // la variable de sortie V sera affichée
    n=length(V);
    for l=1:n-1 do                   // on recommence n-1 fois l'algorithme
          coefmin=V(l);
          pos=l;
          for k=(l+1):n do                    // algorithme qui détermine la position du
                if V(k)<coefmin then          // l-ième plus petit coeff
                      coefmin=V(k);
                      pos=k;
                end
          end
          temp=V(l);  // échange de V(l) et V(pos)
          V(l)=V(pos);  // donc qui met le l-ième plus petit coeff en position l
          V(pos)=temp;
    end
endfunction
```

Exemple étape par étape pour la matrice ligne (8 5 2 6 9 3 1 4 0 7) :

$$\begin{aligned}
&\text{Etape 1} : (8\,5\,2\,6\,9\,3\,1\,4\,\mathbf{0}\,7) \longrightarrow (\mathbf{0}\,5\,2\,6\,9\,3\,1\,4\,8\,7)\\
&\text{Etape 2} : (0\,5\,2\,6\,9\,3\,\mathbf{1}\,4\,8\,7) \longrightarrow (0\,\mathbf{1}\,2\,6\,9\,3\,5\,4\,8\,7)\\
&\text{Etape 3} : (0\,1\,\mathbf{2}\,6\,9\,3\,5\,4\,8\,7) \longrightarrow (0\,1\,\mathbf{2}\,6\,9\,3\,5\,4\,8\,7)\\
&\text{Etape 4} : (0\,1\,2\,6\,9\,\mathbf{3}\,5\,4\,8\,7) \longrightarrow (0\,1\,2\,\mathbf{3}\,9\,6\,5\,4\,8\,7)\\
&\text{Etape 5} : (0\,1\,2\,3\,9\,6\,5\,\mathbf{4}\,8\,7) \longrightarrow (0\,1\,2\,3\,\mathbf{4}\,6\,5\,9\,8\,7)\\
&\text{Etape 6} : (0\,1\,2\,3\,4\,6\,\mathbf{5}\,9\,8\,7) \longrightarrow (0\,1\,2\,3\,4\,\mathbf{5}\,6\,9\,8\,7)\\
&\text{Etape 7} : (0\,1\,2\,3\,4\,5\,\mathbf{6}\,9\,8\,7) \longrightarrow (0\,1\,2\,3\,4\,5\,\mathbf{6}\,9\,8\,7)\\
&\text{Etape 8} : (0\,1\,2\,3\,4\,5\,6\,\mathbf{7}\,8\,9) \longrightarrow (0\,1\,2\,3\,4\,5\,6\,\mathbf{7}\,8\,9)\\
&\text{Etape 9} : (0\,1\,2\,3\,4\,5\,6\,9\,\mathbf{8}\,7) \longrightarrow (0\,1\,2\,3\,4\,5\,6\,7\,\mathbf{8}\,9)
\end{aligned}$$

Ces deux algorithmes sont aussi utilisables avec des matrices colonnes.

7.

```
function[j]=nbelementsdistincts(V)         // la variable de sortie j sera affichée
    n=length(V);
    V=tribulle(V); // ou V=triselection(V) pour trier V
    j=1;   // au moins une valeur dans les coeff de V
    for k=1:(n-1) do
         if V(k)<>V(k+1) then j=j+1  // on parcourt V et dès qu'on trouve une valeur
         end                         // distincte de la précédente cela donne une
    end                              // valeur de plus à comptabiliser puisque V
endfunction                          // est trié
```