*UNIVERSITE CENTRALE*
**Ecole IT**
**Département Informatique**
1ière année Licence
Atelier de programmation 1 : Langage C
Enseignants :
S. ZGHAL (Cours)
2020-2021

# TD 7 : Tableaux à deux dimensions

**Exercice 1 : Carré magique**

Un carré magique est un arrangement de nombres avec n lignes et n colonnes. La somme des valeurs dans chaque ligne, colonne et diagonale est la même. Par exemple, le carré suivant est magique.

| | | | |
|---|---|---|---|
| 16 | 9 | 2 | 7 |
| 6 | 3 | 12 | 13 |
| 11 | 14 | 5 | 4 |
| 1 | 8 | 15 | 10 |

- Ecrire un programme qui permet de déterminer si un carré est magique.
- Une autre condition pour un arrangement de nombres avec n lignes et n colonnes d'être un vrai carré magique est qu'il doit contenir tous les entiers 1, 2, ..., $n^2$. Améliorer le programme pour qu'il vérifie également cette condition.

**Exercice 2 : Multiplcation**

Ecrire un programme qui permet d'effectuer la multiplication de deux matrices A et B avec respectivement les dimensions m x n et n x p. Noter que le nombre de colonnes de A doit être égal au nombre de lignes de B. Le résultat de la multiplication est une matrice C de dimensions m x p avec :

$$c_{ij} = \sum_{k=1}^{n} a_{ik} * b_{kj}$$

**Exercice 3 : Extraction**

Ecrire un programme qui permet d'extraire une ligne et une colonne donnée à partir d'un tableau à deux dimensions.

**Exercice 4 : Symétrique**

Ecrire un programme qui permet de vérifier si une matrice d'ordre n (cad composée de n lignes et de n colonnes) est une matrice symétrique.

**Exercice 5 : Ventes**

On va considérer un tableau à deux dimensions qui regroupe les informations relatives aux ventes de voitures dans une concession. Une première dimension sert à représenter les différents modèles de voitures (une colonne pour chaque modèle). Une deuxième dimension sert à représenter les ventes d'un vendeur de l'entreprise (une ligne par vendeur). Une case contient le nombre de voitures d'un modèle donné vendu par un vendeur X.

- On suppose qu'il y a 4 modèles et 4 vendeurs. Ecrivez un programme qui crée le tableau des ventes et lit au clavier les données permettant de le remplir.
- Modifier le programme pour qu'il puisse donner le nombre d'exemplaires vendus pour chacun des modèles.
- On donne le prix de chaque modèle dans un second tableau à une seule dimension. Modifier le programme pour calculer le chiffre d'affaire généré par chacun des vendeurs, c'est à dire le total de ses ventes exprimé en dinars.

**Exercice 6 : Photo**

On veut représenter une photo en noir et blanc comme un ensemble de points, avec pour chaque point un niveau de gris codé par un entier compris entre 0 (pour noir) et 255 (blanc). Les points ont des coordonnées cartésiennes (x,y) indiquant leur position sur l'image.

Ecrire un programme qui contient successivement :

1. La saisie d'une image (de 200 points sur 100 points) au clavier. Pour cela, on pourra demander à l'utilisateur de rentrer le niveau de gris de chaque point. Il faut vérifier que ce niveau de gris est bien compris entre 0 et 255.
2. Le calcul du nombre de points blancs sur cette image et le pourcentage des points qui sont blancs. Ce pourcentage pourra être de type rél.

3. L'éclaircissement de l'image obtenu en ajoutant 30 à chaque niveau de gris, sans dépasser toutefois le nombre de 255 (par exemple 100 sera transformé en 130, 255 restera 255 et 240 deviendra 255).

# TD 7 : Tableaux à deux dimensions (Correction)

## Exercice 1 : Carré magique

```
#include<stdio.h>
void main()
{
   int M[10][10];
   int T[100];
   int i,j,n,d2,d1,magic,s,vmagic;

   printf("Saisir l'ordrede la matrice : ");
   scanf("%d",&n);
   printf("Remplir la matrice : \n");
   for(i=0;i<=n-1;i++)
   {
      for(j=0;j<=n-1;j++)
          {
             printf("M[%d][%d]=",i,j);
             scanf("%d",&M[i][j]);
          }
   }
   //diagonales
   d1=0;
   for(i=0;i<=n-1;i++)
   {d1=d1+M[i][i];}
   d2=0;
   j=n-1;
   for(i=0;i<=n-1;i++)
   {
      d2=d2+M[i][j];
          j--;
   }
   if(d2==d1)
   {magic=1;}
   else
   {magic=0;}
   //lignes
   i=0;
   while((i<=n-1)&&(magic))
   {
      s=0;
      for(j=0;j<=n-1;j++)
          {s=s+M[i][j];}
          if(s==d2)
          {i++;}
          else
          {magic=0;}
   }
   //colonnes
   j=0;
   while((j<=n-1)&&(magic))
   {
      s=0;
      for(i=0;i<=n-1;i++)
          {s=s+M[i][j];}
          if(s==d2)
          {j++;}
```

```
        else
        {magic=0;}
  }
  if(magic)
  {
        printf("Carré est magique\n");
        for(i=0;i<=(n*n)-1;i++)
        {T[i]=0;}
    for(i=0;i<=n-1;i++)
        {
      for(j=0;j<=n-1;j++)
          {T[M[i][j]]=T[M[i][j]]+1;}
        }
        vmagic=1;
        i=1;
        while(i<=(n*n)-1)
        {
          if(T[i]==1)
          {i++;}
          else
          {vmagic=0;}
        }
    if(magic)
        {printf("Carré est un vrai magique\n");}
        else
    {printf("Carré n'est pas un vrai magique\n");}
  }
  else
  {printf("Carré n'est pas magique\n");}
}
```

## Exercice 2 : Multiplcation

```
#include<stdio.h>
void main()
{
  int M1[3][2],M2[2][4],M[3][4];
  int i,j,k;

  printf("Remplir M1\n");
  for(i=0;i<=2;i++)
  {
    for(j=0;j<=1;j++)
        {
          printf("M1[%d][%d]=",i,j);
          scanf("%d",&M1[i][j]);
        }
  }
  printf("Remplir M2\n");
  for(i=0;i<=1;i++)
  {
    for(j=0;j<=3;j++)
        {
          printf("M2[%d][%d]=",i,j);
          scanf("%d",&M2[i][j]);
        }
  }
  //Multiplcation
  for(i=0;i<=2;i++)
```

```
  {
    for(j=0;j<=4;j++)
        {
          M[i][j]=0;
          for(k=0;k<=1;k++)
          {M[i][j]=M[i][j]+M1[i][k]*M2[k][j];}
        }
  }
  printf("Le contenu de la matrice résultat est :\n");
  for(i=0;i<=2;i++)
  {
    for(j=0;j<=3;j++)
        {printf("%d\t",M[i][j]);}
        printf("\n");
  }
}
```

## Exercice 3 : Extraction

```
#include<stdio.h>
void main()
{
  int M1[3][2];
  int i,j,ligne,colonne;
  int Tl[2],Tc[3];

  printf("Remplir M1\n");
  for(i=0;i<=2;i++)
  {
    for(j=0;j<=1;j++)
        {
          printf("M1[%d][%d]=",i,j);
          scanf("%d",&M1[i][j]);
        }
  }
  printf("Saisir le numero de la ligne : ");
  scanf("%d",&ligne);
  for(j=0;j<=1;j++)
  {Tl[j]=M1[ligne][j];}
  printf("La ligne contient :\n");
  for(j=0;j<=1;j++)
  {printf("%d\t",Tl[j]);}
  printf("\n");

  printf("Saisir le numero de la colonne : ");
  scanf("%d",&colonne);
  for(i=0;i<=2;i++)
  {Tc[i]=M1[i][colonne];}
  printf("La colonne contient :\n");
  for(i=0;i<=2;i++)
  {printf("%d\t",Tc[i]);}
  printf("\n");
}
```

## Exercice 4 : Symétrique

```
#include<stdio.h>
void main()
```

```
{
  int M[10][10];
  int i,j,sym,n;

  printf("Saisir l'ordrede la matrice : ");
  scanf("%d",&n);
  printf("Remplir la matrice : \n");
  for(i=0;i<=n-1;i++)
  {
    for(j=0;j<=n-1;j++)
        {
          printf("M[%d][%d]=",i,j);
          scanf("%d",&M[i][j]);
        }
  }
  i=0;
  sym=1;
  while((i<=n-1)&&(sym))
  {
    j=0;
    while((j<=n-1)&&(sym))
        {
          if(M[i][j]==M[j][i])
          {j++;}
          else
          {sym=0;}
        }
        if(sym)
        {i++;}
  }
  if(sym)
  {printf("Matrice est symetrique\n");}
  else
  {printf("Matrice n'est pas symetrique\n");}
}
```