Corrigé série 8.2 : Les tableaux

Exercice 1 : Somme des éléments d'un tableau 1D

Écrire un programme qui

- déclare un tableau de type **int** , d'une capacité de 50 valeurs
- demande à l'utilisateur le nombre de valeurs **n**
- remplit le tableau avec **n** valeurs entrées au clavier
- affiche les n premiers éléments tableau (on peut afficher les autres « pour voir »)
- Calcule la somme des valeurs saisies se trouvant dans le tableau

```
#include <stdio.h>

#define N 50        // ou   const int N=50;

int main()
{
  int tableau[N];   /* Déclaration du tableau   */
  int n;            /* nombre de valeurs      */
  int i;            /* indice courant */
  long somme;     /* somme des éléments - type long à cause
                  de la grandeur possible du résultat. */

  /* Saisie des données */
  printf("Nombre de valeurs (max.%d) : ", N);
  scanf("%d", &n );
  for (i=0; i<n; i++)
  {
     printf("Valeur %d ? ", i);
     scanf("%d", &tableau[i]);
  }

  /* Affichage du tableau */
  printf("Tableau :\n");
  for (i=0; i<n; i++)
     printf("%d ", tableau[i]);
  printf("\n");

  /* Calcul de la somme */
  somme=0;
  for (i=0; i<n; i++)
     somme += tableau[i];

  printf("Somme de éléments : %ld\n", somme);

  return 0;
}
```

Exercice 2 : {TABPOS} ← {TAB} → {TABNEG} –

```
#include <stdio.h>

#define N 50

int main()
{
  int tab[N], tabPos[N], tabNeg[N];
  int n;
  int i, iPos, iNeg;

  printf("Nombre de valeurs (max.%d) : ",N);
  scanf("%d", &n );

  for (i=0; i<n; i++)              /* saisie des valeurs */
  {
     printf("Valeur %d ? ", i);
     scanf("%d", &tab[i]);
  }

  iPos = 0;
  iNeg = 0;

  for (i=0; i<n; i++)              /* Copie des données */
  {
    if (tab[i]>0){
       tabPos[iPos]=tab[i];
       iPos++;
    }
    else
      if (tab[i]<0) {
          tabNeg[iNeg]=tab[i];
          iNeg++;
       }
  }

  printf("Tableau positif :\n");   /* Affichage tableau POSITIF  */
  for (i=0; i<iPos; i++)
     printf("%d ", tabPos[i]);
  printf("\n");

  printf("Tableau negatif :\n");   /* Affichage tableau NEGATIF  */
  for (i=0; i<iNeg; i++)
     printf("%d ", tabNeg[i]);
  printf("\n");
  return 0;
}
```

Exercice 3 : Somme des éléments d'un tableau 2D

Écrire un programme qui :

- déclare un tableau de type **`int`** (taille maximale: 50 lignes et 50 colonnes).
- lit les tailles (nombre de lignes et nombre de colonnes)
- remplit le tableau avec des valeurs entrées au clavier
- affiche le tableau
- affiche la somme de tous ses éléments.

```
#include <stdio.h>
#define N 50
int main
{
 int tableau[N][N];
 int nbLigne, nbColonne;
 int i, j;
 long somme;

 /* Saisie des données */
 printf("Nombre de lignes   (max.%d) : ",N);
 scanf(" %d", &nbLigne );
 printf("Nombre de colonnes (max.%d) : ",N);
 scanf(" %d", &nbColonne );

 for (i=0; i<nbLigne; i++)
    for (j=0; j<nbColonne; j++)
    {
      printf("Elément[%d][%d] : ", i, j);
      scanf(" %d", &tableau[i][j]);
    }

  /* Affichage du tableau */
 printf("Tableau donné :\n");
 for (i=0; i<nbLigne; i++)
 {
   for (j=0; j<nbColonne; j++)
      printf("%7d", tableau[i][j]);
   printf("\n");
 }

 /* Calcul de la somme  */
 somme=0;
 for (i=0; i<nbLigne; i++)
     for (j=0; j<nbColonne; j++)
         somme += tableau[i][j];

 /* Affichage du résultat */
 printf("Somme des éléments : %ld\n", somme);
 return 0;
}
```

Exercice 4 : Produit scalaire

a) Écrire un programme qui calcule le produit scalaire de deux vecteurs d'entiers **u** et **v** (de même dimension).

```
#include <stdio.h>
int main()
{
 /* Déclarations */
 int u[50], v[50]; /* vecteurs donnés */
 int n;     /* dimension        */
 int i;     /* indice courant   */
 long ps;   /* produit scalaire */

 /* Saisie des données */
 printf("Dimension des vecteurs (max.50) : ");
 scanf("%d", &n );
 printf("** Premier vecteur **\n");
 for (i=0; i<n; i++)
 {
     printf("Elément %d : ", i);
     scanf("%d", &u[i]);
 }
 printf("** Deuxième vecteur **\n");
 for (i=0; i<n; i++)
 {
     printf("Elément %d : ", i);
     scanf("%d", &v[i]);
 }
 /* Calcul du produit scalaire */
 for (ps=0, i=0; i<n; i++)
      ps += (long)u[i]*v[i];
 /* Edition du résultat */
 printf("Produit scalaire : %ld\n", ps);
 return 0;
}
```

b) Factoriser le code de ce calcul, en le mettant dans une fonction prenant en paramètre les deux vecteurs. Appeler cette fonction du programme principal pour vérifier son bon fonctionnement.

```
// calcul du produit scalaire de2 vecteurs (de dim. n)
long produitScalaire(int v1[], int v2[], int n)
{
  int i=0 ; long produit=0;
  for (i=0 ; i<n; i++)
      produit += (long)v1[i]*v2[i];
  return produit;
}
main(){…
   ps = produitScalaire(u, v, n);
   printf("Produit scalaire : %ld\n", ps);   ...
}
```

Exercice 5 : Recherche du min et du max dans tableau 1D

Ecrire un programme qui détermine la **plus grande** et la **plus petite valeur** dans un tableau d'entiers. Afficher ensuite la valeur et la position du maximum et du minimum.

Note: si le tableau contient plusieurs maxima ou minima, le programme retiendra la position du premier maximum ou minimum rencontré.

```
#include <stdio.h>
int main()
{
 /* Déclarations */
 int a[50]; /* tableau donné  */
 int n;     /* dimension      */
 int i;     /* indice courant */
 int iMin;   /* position du minimum */
 int iMax;   /* position du maximum */

 /* Saisie des données */
 printf("Dimension du tableau (max.50) : ");
 scanf("%d", &n );
 for (i=0; i<n; i++)
 {
     printf("Elément %d : ", i);
     scanf("%d", &a[i]);
 }

 /* Affichage du tableau */
 printf("Tableau donné :\n");
 for (i=0; i<n; i++)
     printf("%d ", a[i]);
 printf("\n");

 /* Recherche du maximum et du minimum */
 iMin = 0;
 iMax = 0;
 for (i=1; i<n; i++)
 {
      if( a[i] > a[iMax] )
          iMax = i;
      else if( a[i] < a[iMin] )
          iMin = i;
 }

  /* Edition du résultat */
 printf("Position du minimum : %d\n", iMin);
 printf("Position du maximum : %d\n", iMax);
 printf("Valeur   du minimum : %d\n", a[iMin]);
 printf("Valeur   du maximum : %d\n", a[iMax]);
 return 0;
}
```

Exercice 6 : Produit matriciel [Défi]

$$C = A \cdot B = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix} \cdot \begin{bmatrix} 10 & 11 \\ 12 & 13 \\ 14 & 15 \end{bmatrix} = \begin{bmatrix} 1\cdot10+2\cdot12+3\cdot14 & 1\cdot11+2\cdot13+3\cdot15 \\ 4\cdot10+5\cdot12+6\cdot14 & 4\cdot11+5\cdot13+6\cdot15 \end{bmatrix} = \begin{bmatrix} 76 & 82 \\ 184 & 199 \end{bmatrix}$$

```
/* Description: Multiplication de matrices C=A*B
** Auteur: FRT
** Date: 11.12.2104
*/

#include <stdio.h>
#include <stdlib.h>

void afficherMatrice(int* M, int n, int m)
{
    int i,j;
    for(i=0; i<n; i++) {
        printf("| ");
        for(j=0; j<m; j++) {
            printf("%4d",M[i*m+j]);
        }
        printf("|\n");
    }
}

void remplirMatrice(int* M, int n, int m)
{
    int i,j;
    printf("Matrice de %d x %d\n", n,m);
    for(i=0; i<n; i++) {
        for(j=0; j<m; j++) {
            printf("Valeur [%d][%d]\n",i,j);
            scanf(" %d",&M[i*m+j]);
        }
    }
}

//Suite à la page suivante
```

```
int main()
{
    int i,j,k;
    int nA,mA,nB,mB;

    printf("Matrice A (n m)");
    scanf(" %d %d",&nA,&mA);
    int A[nA][mA];

    printf("Matrice B (n m)");
    scanf(" %d %d",&nB,&mB);
    int B[nB][mB];

    if (mA!=nB)
    {
        printf("taille non correcte\n");
        exit(0);
    }
    int C[nA][mB];

    printf("A:\n"); remplirMatrice((int*)A, nA, mA);
    printf("B:\n"); remplirMatrice((int*)B, nB, mB);

    /* MULTIPLICATION */
    for(i=0;i<nA;i++)
    {
        for(j=0;j<mB;j++)
        {
            C[i][j]=0;
            for(k=0;k<mA;k++)
            {
                C[i][j] += A[i][k] * B[k][j];
            }
        }
    }

    printf("A:\n"); afficherMatrice((int*)A, nA, mA);
    printf("B:\n"); afficherMatrice((int*)B, nB, mB);
    printf("C:\n"); afficherMatrice((int*)C, nA, mB);

    return 0;
}
```