Ministère de l'Enseignement Supérieur et de la Recherche Scientifique
UNIVERSITE M'Hamed BOUGARA de BOUMERDES. FACULTE DES SCIENCES, DEPARTEMENT D'INFORMATIQUE

ETLD, Durée : 1h30 — Module : Algo. & Structures de Données Statiques 1
Barème approximatif (7+7+6) — Filière: 1ière Ing.Info- S1-2023/2024, Date : 11/01/2024

Vous pouvez utiliser le langage algorithmique du cours ou le langage C. Les mélanger dans le même exercice est interdit.

Exercice 1 (7pts)

La suite **(U_n)** est définie comme suit:

$$\begin{cases} U_1 = -2 \\ U_i = 3 \times (U_{i-1})^2 - 10 \times U_{i-1} & \text{si } i>1 \end{cases}$$

Écrire un algorithme ou un programme C qui sans utiliser de tableaux, affiche les valeurs des termes de la suite **(U_n)** inférieures ou égales à une valeur **val** donnée par l'utilisateur.

Exercice 2 (7pts)

T est un tableau de n entiers positifs ($3 \leq n \leq 50$). On appelle **poids** de **T**, noté p(**T**), la somme de ses éléments. Si **e** est un élément quelconque de T, celui-ci partitionne le tableau en deux parties: **T1_e** qui est le sous tableau contenant les éléments de **T** à partir du premier jusqu'à **e** , et **T2_e** est le sous-tableau contenant les éléments restants. Si **e** est le dernier, **T2_e** est vide et son poids est nul.
On appelle **pivot** de **T**, le premier élément **e*** tel que p(**T1_e***) ≥ p(**T2_e***).
En supposant **T** déjà rempli, écrire un algorithme ou un programme C qui détermine la position du pivot d'un tableau **T**, puis affiche sa valeur.
Exemple :

Si **T** =

8	3	1	2	5

, le pivot est dans la 2ieme position car 8 < 3+1+2+5 mais 8+3 ≥ 1+2+5. Sa valeur est 3.

Exercice 3 (6pts)

mat est une matrice quelconque d'entiers (10 lignes et 10 colonnes max.). On appelle diagonale de **mat** toute séquence complète d'éléments parallèle ou confondue avec la diagonale principale (voir exemple).
En supposant **mat** déjà remplie, écrire un algorithme ou un programme C qui met à zéro la diagonale de **mat** ayant la plus petite somme (s'il y en a plusieurs, mettre à zéro une seule suffit).
Exemple : une matrice 3 x 4 possède 6 diagonales. Les cases en gris sont des exemples de ses diagonales.

1) diagonale principale

2	-1	5	9
6	**7**	21	-7
13	5	**6**	6

2) une autre diagonale

2	-1	**5**	9
6	7	21	**-7**
13	5	6	6

3) encore une autre

2	-1	5	9
6	7	21	-7
13	5	6	6

4) la plus petite

2	-1	**5**	9
6	7	21	**-7**
13	5	6	6

5) le résultat est donc

2	-1	**0**	9
6	7	21	**0**
13	5	6	6

Remarques :
- Un bonus de 2 pts (note globale ≤ 20) pour l'utilisation juste des actions paramétrées dans les exos 2 et 3.
- Pour le C, il est interdit d'utiliser des bibliothèques autres que celle associée à `<stdio.h>`.
- Il est interdit d'utiliser **l'effaceur**, sinon **-2 pts**. En cas d'erreur, barrer proprement.

Ministry of Higher Education and Scientific Research
University M'Hamed Bougara of Boumerdes. Faculty of Sciences, Department of Computer Science

ETLD, Duration : 1h30	Module : Algo. & Structures de Données Statiques 1
Approximate grading (7+7+6)	Class: 1ière Ing.Info- S1-2023/2024, Date : 11/01/2024

You can use the algorithmic language or the C language. Mixing them in the same exercise is not allowed.

Exercise 1 (7pts)

The sequence **(U_n)** is defined as follows:

$$\begin{cases} U_1 = -2 \\ U_i = 3 \times (U_{i-1})^2 - 10 \times U_{i-1} & \text{if } i>1 \end{cases}$$

Write an algorithm or a C program that, without using arrays, displays the values of the terms of the sequence **(U_n)** that are less than or equal to a value **val** given by the user.

Exercise 2 (7pts)

T is an array of n positive integers ($3 \le n \le 50$). The weight of **T**, denoted as p(**T**), is the sum of its elements. If e is any element of **T**, it partitions the array into two parts: **T1_e**, which is the sub-array containing the elements of **T** from the first to **e**, and **T2_e**, which is the sub-array containing the remaining elements. If **e** is the last element, **T2_e** is empty, and its weight is zero.
The **pivot** of **T** is defined as the first element **e*** such that p(**T1_e***) ≥ p(**T2_e***).
Assuming **T** is already filled, write an algorithm or a C program that determines the position of the pivot in an array **T** and then displays its value.
Example :

Si **T** = | 8 | 3 | 1 | 2 | 5 | , the pivot is in the 2nd position because 8 < 3+1+2+5 and 8+3 ≥ 1+2+5. Its value is 3.

Exercise 3 (6pts)

mat is an arbitrary matrix of integers (no more than 10 rows and 10 columns). The diagonal of **mat** refers to any complete sequence of elements parallel or coincident with the main diagonal (see example).
Assuming **mat** is already filled, write an algorithm or a C program that sets to zero the diagonal of **mat** that has the smallest sum (if there are several diagonals with the same smallest sum, zeroing out any one of them is sufficient).
Example : A 3 x 4 matrix has 6 diagonals. The grey cells are examples of its diagonals.

1) main diagonal

2	-1	5	9
6	**7**	21	-7
13	5	**6**	6

2) another diagonal

2	-1	**5**	9
6	7	21	**-7**
13	5	6	6

3) yet another

2	-1	5	9
6	7	21	-7
13	5	6	6

4) the smallest

2	-1	**5**	9
6	7	21	**-7**
13	5	6	6

5) hence the result is

2	-1	**0**	9
6	7	21	**0**
13	5	6	6

Remarks:
- A bonus of 2 points (overall score ≤ 20) for the correct use of parametrized actions in exercises 2 and 3.
- For C programming, it is forbidden to use libraries other than the one associated with <stdio.h>.
- Using an ink eraser is prohibited, otherwise **-2 pts**. In case of an error, neatly cross it out.

```c
//**************************************************************
//******* Exercice 1 ********************************************
//**************************************************************
#include <stdio.h>
int val,u,i;                                        //declaration 0.5pt
int main(){
   printf("\n Donner val:");                        //
   scanf("%d",&val);                                //lecture 1pt
   printf("\n Les termes de la suite <= %d sont:",val);
   u = -2;                                          //initialisation 0.5
   for(i=1; u <= val; i++){                         //boucle 2pts
       printf("\n U%d=%d",i,u);                     //affichage 1pt
       u=3*(u*u)-10*u;                              //calcul termes 2pts
   }
   return 0;                                        //si un terme > val
}                                                   //est affiche -1pt

//**************************************************************
//******* Exercice 2 ********************************************
//**************************************************************
#include <stdio.h>
int T[50]={8,3,1,2,5},pos,n=5;                    //declaration 1pt
int getPivotPos(int vect[], int n);
int getTabWeight(int vect[], int n);
int main(){
   pos=getPivotPos(T,n);
   printf("\n La position du pivot (premiere case avce "
          "indice 1) est %d, sa valeur est %d", pos+1, T[pos]);
   return 0;
}
int getPivotPos(int vect[], int n){               //traitement 6pts dont:
   int pos,weight,sum;                            //determination pivot 4pts
   weight=getTabWeight(vect,n);                   //calcul de la somme 2pts
   sum=0;                                         //on accepte n'importe
   for(pos=0;pos<n;pos++){                        //quelle methode juste
      sum+=vect[pos];
      if(sum >= weight/2.0)break;
   }
   return pos;
}
int getTabWeight(int vect[], int n){
   int i,sum;
   sum=0;
   for(i=0;i < n;i++){
      sum+=vect[i];
   }
   return sum;
}

//**************************************************************
//******* Exercice 3 ********************************************
//**************************************************************
#include <stdio.h>                      //declarations 1pt
int mat[10][10]={{2,-1,5,9},{6,7,21,-7},{13,5,6,6}},n=3,m=4;
void lectMat(int mat[][10],int *n,int *m);
int annulLeastDiag(int mat[][10], int n, int m);
void affichMat(int mat[][10],int n,int m);

int main(){
   lectMat(mat,&n,&m);//en plus
   annulLeastDiag(mat,n,m);
   affichMat(mat,n,m);
   return 0;
}
void lectMat(int tab[][10],int *n,int *m){//en plus
   int i,j;
   printf("\nNombre de lignes (<=%d) et nombre de colonnes"
          " (<=%d):",10,10);
   scanf("%d %d",n,m);
```

```c
   for(i=0;i<*n;i++){
     for(j=0;j<*m;j++){
        printf("Mat[%d,%d]:",i+1,j+1);
        scanf("%d",&tab[i][j]);
     }
   }
}//sulp ne

int annulLeastDiag(int mat[][10], int n, int m){   //traitements 4pts dont:
   int sum,minSum,iminDiag,jminDiag,i,j,ii,jj;     //calcul min diag 3pts
   sum=0;                                           //annul min diag 1pt
   for(i=0,j=0;(i<n)&&(j<m);i++,j++){
      sum+=mat[i][j];
   }
   minSum=sum;iminDiag=0;jminDiag=0;
   for(j=1;j<m;j++){
      sum=0;
      for(i=0,jj=j;(i<n)&&(jj<m);i++,jj++){
         sum+=mat[i][jj];
      }
      if(sum<minSum){minSum=sum;iminDiag=0;jminDiag=j;}
   }
   for(i=1;i<n;i++){
      sum=0;
      for(ii=i,j=0;(ii<n)&&(j<m);ii++,j++){
         sum+=mat[ii][j];
      }
      if(sum<minSum){minSum=sum;iminDiag=i;jminDiag=0;}
   }
   for(i=iminDiag,j=jminDiag;(i<n)&&(j<m);i++,j++){
      mat[i][j]=0;
   }
   return 0;
}

void affichMat(int mat[][10],int n,int m){           //affichage 1pt
   int i,j;
   for(i=0;i<n;i++){
     for(j=0;j<m;j++){
        printf("%5d",mat[i][j]);
     }
     printf("\n");
   }
}
```