1. RECHERCHE DANS UN TABLEAU

EXERCICES TABLEAUX

Ecrire une fonction retournant le maximum d'un tableau de N éléments réels (N ≤ TAILLE)

```
CONST TAILLE=50
TYPE TABL = TABLEAU [1..TAILLE] DE REEL

FONCTION max_tableau(T:TABL; N:ENTIER): REEL
VAR         i :ENTIER
Maximum :REEL
DEBUT
            Maximum⇦ T[1]
            POUR i DE 2 A N
            FAIRE
                        SI (T[i] > Maximum)
                        ALORS
                                    Maximum⇦ T[i]
                        FINSI
            FINPOUR
            Retourner(Maximum)
FIN
```

EXERCICES TABLEAUX

Écrire un algorithme qui permet de compter le nombre des valeurs négatives dans un tableau de N entiers

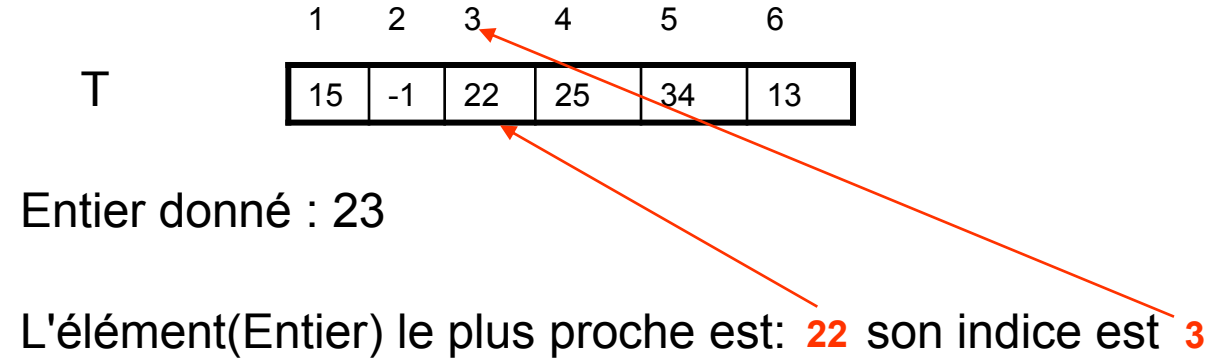
```
ALGORITHME compter_Nbre_Négatives
CONST      TAILLE=100
TYPE TABL=TABLEAU[1..TAILLE] d'ENTIER
VAR        T              : TABL
           N,i,Compteur: ENTIER
DEBUT
ECRIRE(" Entrer le nombre d'éléments dans le tableau N < ",TAILLE)
LIRE(N)
SAISIR (T,N)
Compteur ⇦ 0
i ⇦ 1
TANTQUE (i ≤ N)
FAIRE
           SI T[i] <  0
           ALORS
                 Compteur ⇦Compteur+1
           FINSI
           i⇦ i + 1
FINTANTQUE

ECRIRE(" Le nombre des  valeurs négatives est: ",Compteur)
FIN
```

EXERCICES TABLEAUX

Ecrire un algorithme qui permet de trouver l'élément le plus proche d'un entier donné dans un tableau d'entiers, de l'afficher ainsi d'afficher son indice dans le tableau.

Exemple:



Entier donné : 23

L'élément(Entier) le plus proche est: **22** son indice est **3**

Idée: comparer la différence entre l'élément donné ET l'élément du tableau

```
ALGORITHME PLUSPROCHE
CONST TAILLE=50
TYPE TABL = Tableau [1..TAILLE] de REEL
VAR        T                        :TABL
element, Element_proche,DIF              :REEL;
N, i , indice_proche                :ENTIER
DEBUT
ECRIRE (" Entrer le nombre d'élément du tableau (N <",TAILLE," )")
LIRE(N)
SAISIR(T,N)
ECRIRE (" Entrer un élément")
LIRE (element)
DIF  ⇦ abs( T[1]-element)
Element_proche⇦T[1]
Indice_proche⇦ 1
POUR i DE 2 A N
FAIRE
            SI DIF > abs(T[i] - element )
            ALORS
                        DIF  ⇦ abs(T[i] - element )
                        Element_proche⇦T[i]
                        Indice_proche ⇦ i
            FINSI
FINPOUR
ECRIRE( "L'élément le plus proche est :", Element_proche," son indice est ",
Indice_proche)
FIN
```

```
ALGORITHME PLUSPROCHE
CONST TAILLE=50
TYPE TABL = Tableau [1..TAILLE] de REEL
VAR        T                          :TABL
element, Element_proche,DIF           :REEL;
N, i , indice_proche                  :ENTIER
DEBUT
ECRIRE (" Entrer le nombre d'élément du tableau (N <",TAILLE," )")
LIRE(N)
SAISIR(T,N)
ECRIRE (" Entrer un élément")
LIRE (element)
DIF  ⇦ abs( T[1]-element)
Element_proche⇦T[1]
Indice_proche⇦ 1
POUR i DE 2 A N
FAIRE
        SI DIF > abs(T[i] - element )
        ALORS
                DIF  ⇦ abs(T[i] - element )
                Element_proche⇦T[i]
                Indice_proche ⇦ i
        FINSI
FINPOUR
ECRIRE( "L'élément le plus proche est :", Element_proche," son indice est ",
Indice_proche)
FIN
```

NB: Remplacer par T[Indice_proche]

```
ALGORITHME PLUSPROCHE
CONST TAILLE=50
TYPE TABL = Tableau [1..TAILLE] de REEL
VAR        T                    :TABL;
element, ,DIF          :REEL
N, i , indice_proche            :ENTIER
DEBUT
ECRIRE (" Entrer le nombre d'élément du tableau (N <",TAILLE," )")
LIRE (N)
SAISIR(T,N)
ECRIRE (" Entrer un élément")
LIRE (element)
DIF  ⇦ abs( T[1]-element)

Indice_proche⇦ 1
POUR i DE 2 A TAILLE
FAIRE
            SI DIF > abs(T[i] - element )
            ALORS
                        DIF  ⇦ abs(T[i] - element )

                        Indice_proche ⇦ i
            FINSI
FINPOUR
Ecrire "L'élément le plus proche est :", T[indice_proche]," son indice est ", Indice_proche)
FIN
```

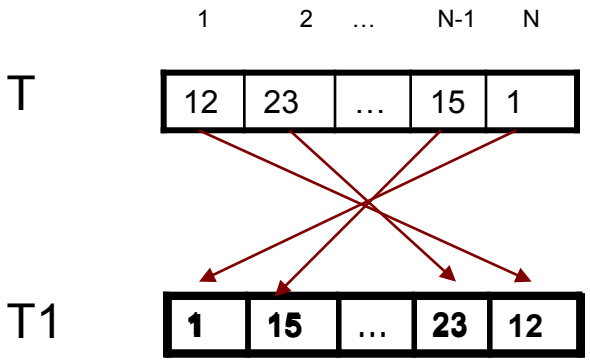
2. MODIFICATION ET TRI DANS UN TABLEAU

EXERCICES TABLEAUX

Ecrire algorithme qui permet de ranger les éléments d'un tableau de N entiers dans l'ordre inverse

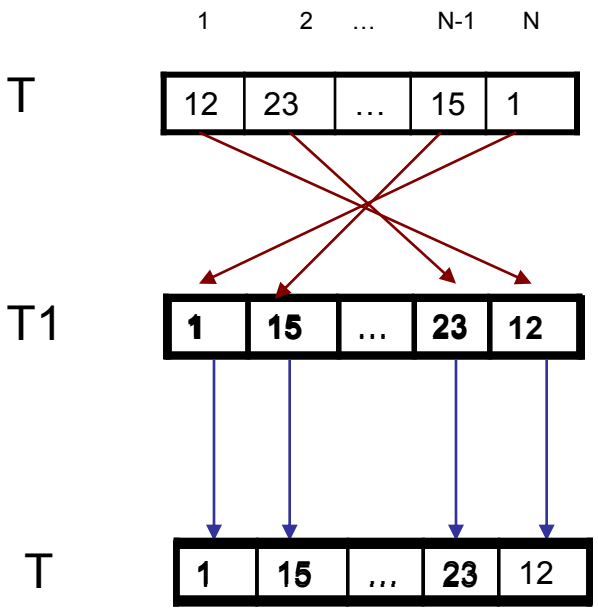
EXERCICES TABLEAUX

Ecrire algorithme qui permet de ranger les éléments d'un tableau de N entiers dans l'ordre inverse



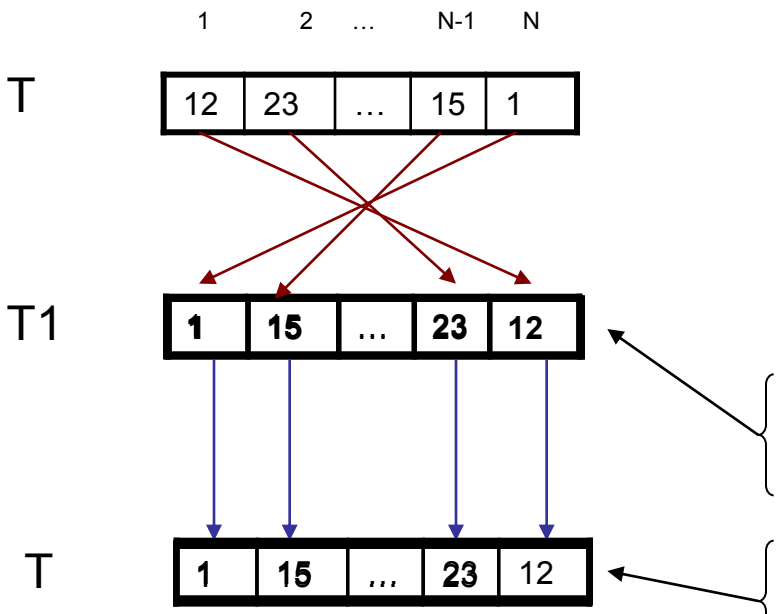
EXERCICES TABLEAUX

Ecrire algorithme qui permet de ranger les éléments d'un tableau de N entiers dans l'ordre inverse



EXERCICES TABLEAUX

Ecrire algorithme qui permet de ranger les éléments d'un tableau de N entiers dans l'ordre inverse

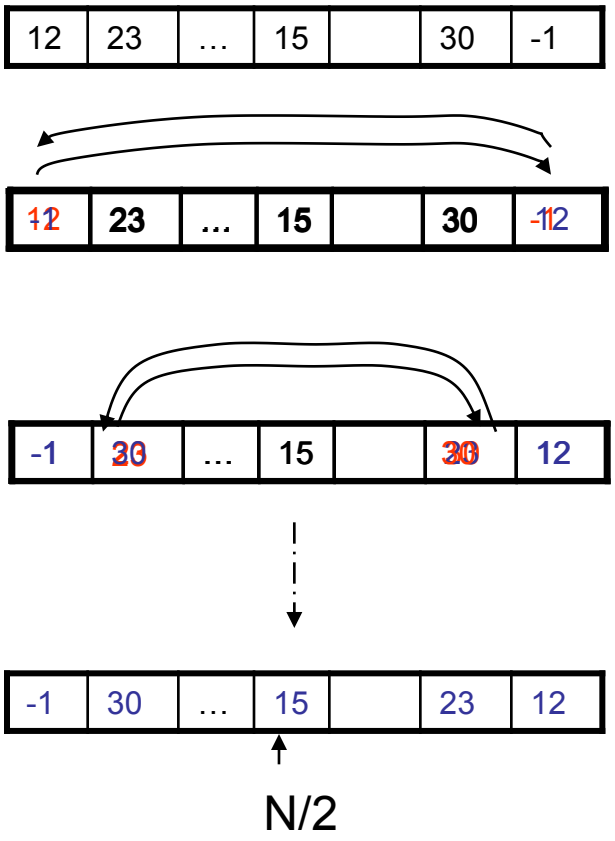



```
ALGORITHME inverseTab
CONST TAILLE=100
TYPE TABL=TABLEAU[1..TAILLE] d'ENTIER
VAR  T,T1: TABL
       i,N    : ENTIER
DEBUT
ECRIRE("Entrer le nombre d'élément du tableau N:")
LIRE(N)
SAISIR (T,N);
POUR i DE 1 A N FAIRE
       T1[i] ⇦ T[N+1-i]
FINPOUR

POUR i DE 1 A N FAIRE
       T[i] ⇦ T1[i]
FINPOUR
FIN
```

EXERCICES TABLEAUX

Ecrire algorithme qui permet de ranger les éléments d'un tableau de N entiers dans l'ordre inverse




```
ALGORITHME inverseTab
CONST TAILLE=100
TYPE TABL=TABLEAU[1..TAILLE] d'ENTIER
VAR T        : TABL
i, N,Milieu  : ENTIER
DEBUT
ECRIRE("Entrer le nombre N:")
LIRE(N)
SAISIR(T,N)
Milieu ⇦ N/ 2
POUR i  DE  1 A Milieu FAIRE
            Echanger (T,i, N+1-i)
FINPOUR
FIN
```

```
PROCEDURE Echanger(VAR T: TABL; i,j :ENTIER)
VAR aux :ENTIER
DEBUT
      aux⇦ T[i]
      T[i] ⇦ T[j]
      T[j] ⇦ aux
FIN
```

EXERCICES TABLEAUX

Ecrire algorithme qui permet d'insérer un élément dans un tableau d'entier trié

T

1	2	3	4	5	6	….	78	79	80
-12	-3	0	14	27	28	….	39	39	126

Tableau trié

Question: insérer le nombre 19

1	2	3	4	5	6	….	78	79	80
-12	-3	0	14	27	28	….	39	39	126

Décalage à droite

à Insérer ICI **T[i] > 19**

1	2	3	4	5	6	7	78	79	80	**81**
-12	-3	0	14	**19**	27	28	…	39	39	126

EXERCICES TABLEAUX

Ecrire algorithme qui permet d'insérer un élément dans un tableau trié

```
ALGORITHME inserer_element
CONST     TAILLE=100
TYPE TABL=TABLEAU[1..TAILLE] d'ENTIER
VAR       T              : TABL
          N,i,element: ENTIER
          Trouve      : BOOLEEN
DEBUT
ECRIRE(" Entrer le nombre d'éléments dans le tableau N < ",TAILLE)
LIRE(N)
SAISIR (T,N)
ECRIRE(" Entrer l'élément à insérer")
LIRE(element)
i ⇦1
Trouve ⇦ Faux
TANTQUE (i ≤N  ET NON Trouve )
FAIRE
      SI (T[i] > element) ALORS
            INSERER(T,N,i,element)
            Trouve ⇦ vraie
      FINSI
      i ⇦ i+1
FINTANTQUE
SI NON Trouve ALORS
            T[N+1] ⇦ element
FINSI
FIN
```

```
PROCEDURE INSERER(VAR T: TABL; N,i :ENTIER ;element:ENTIER)
VAR j : ENTIER
DEBUT
POUR j DE N A  i   par Pas -1
Faire
      T[j+1] ⇦ T[j]
FINPOUR
T[i] ⇦ element
FIN
```

N..N-1..N-2….i

Décalage à droite

Insérer element à la ième position

EXERCICES TABLEAUX

Écrire un algorithme qui permet de compter et éliminer les zéros d'un tableau de N entiers

```
ALGORITHME compter_zero
CONST      TAILLE=100
TYPE TABL=Tableau[1..TAILLE] d'ENTIER
VAR        T              : TABL
           N,i,Compteur: ENTIER
DEBUT
ECRIRE(" Entrer le nombre d'éléments dans le tableau N < ",taille)
LIRE(N)
SAISIR (T,N)
Compteur ⇦ 0
i ⇦ 1
TANTQUE (i ≤ N)
FAIRE
           Si T[i] = 0 alors
                 Compteur ⇦Compteur+1
                 decalage_gauche(T,N,i)
                 N⇦N-1
           sinon
           i⇦ i + 1
           Finsi
FINTANTQUE
FIN
```

```
Procedure decalage_gauche(VAR T :TABL;  N,  i:ENTIER)
VARindice :ENTIER
DEBUT
      POUR indice DE i A N-1  FAIRE
            T[indice] ⇦T[indice+1]
      FINPOUR
FIN
```

EXERCICES TABLEAUX

Écrire un algorithme qui permet de compter et éliminer les zéros d'un tableau de N entiers

```
ALGORITHME compter_zero
CONST      TAILLE=100
TYPE TABL=Tableau[1..TAILLE] d'ENTIER
VAR        T              : TABL
           N,i,Compteur: ENTIER
DEBUT
ECRIRE(" Entrer le nombre d'éléments dans le tableau N < ",taille)
LIRE(N)
SAISIR (T,N)
Compteur ⇦ 0
i ⇦ 1
TANTQUE (i ≤ N)
FAIRE
           Si T[i] = 0 alors
                 Compteur ⇦Compteur+1
                 decalage_gauche(T,N,i)
                 N⇦N-1
           Finsi
           i⇦ i + 1
FINTANTQUE
FIN
```

```
Procedure decalage_gauche(VAR T :TABL;  N,  i:ENTIER)
VARindice :ENTIER
DEBUT
      POUR indice DE i A N-1  FAIRE
            T[indice] ⇦T[indice+1]
      FINPOUR
FIN
```

EXERCICES TABLEAUX

Ecrire une **Procedure FUSION** qui construit un tableau **TAB_FUS** trié par ordre croissant avec les éléments de deux tableaux A et B triés par ordre croissant.
Pour deux tableaux de dimensions N et M, le tableau TAB_FUS aura la dimension N+M

```
TYPE  TABLN=TABLEAU[1..N] DE REEL
TYPE  TABLM=TABLEAU[1..M] DE REEL
TYPE  TABLF=TABLEAU[1..F] DE REEL     (* avec N+M <= F*)
```

```
PROCEDURE  FUSION( VAR TAB_FUS :TABLF ;  A:TABLN; B :TABLM )
VAR i, j, k :ENTIER
DEBUT
        i ⇦ 1
        j ⇦ 1
        k⇦ 1
        TANTQUE  (i ≤ N) ET ( j  ≤  M)
        FAIRE
                SI A[i] < B[j] ALORS
                        TAB_FUS[k] ⇦ A[i]
                        i ⇦ i+1
                SINON
                        TAB_FUS[k] ⇦ B[j]
                        j⇦ j+1
                FINSI
                k⇦ k+1
        FINTANTQUE
        TANTQUE i ≤  N              /* compléter le tableau  TAB_Fus par les éléments du tableau A */
        FAIRE                       /*N.B: Tous les elements du tableau B  ont été rempli dans TAB_FUS */
                        TAB_FUS[k] ⇦ A [i]
                        k  ⇦ k+1
                        i ⇦i +1
        FINFAIRE
        TANTQUE j ≤  M              /* compléter le tableau  TAB_Fus par les éléments du tableau B */
        FAIRE                       /*N.B: Tous les éléments du tableau A  ont été rempli dans TAB_FUS */
                        TAB_FUS[k] ⇦ B [j]
                        k  ⇦ k+1
                        j ⇦j +1
        FINFAIRE
FIN
```

EXERCICES TABLEAUX

Ecrire la fonction TRI_BULLE qui trie un tableau de N éléments entiers par ordre croissant en appliquant la méthode de la bulle

```
PROCEDURE TriBulle (var T :TABL ; N :ENTIER)
var     aux,borne: ENTIER
        Est_trié       :BOOLEEN
DEBUT
        Est_trié ⇦ faux              /*Détecte si le tableau est trié*/
        borne ⇦ N
        TANTQUE Non(Est_Trié )
        FAIRE
                Est_Trié ⇦ vrai
                POUR i  DE  1  A  borne-1
                        Si T[i] > T[i+1]            /* Compare 2 éléments successifs*/
                        Alors
                                Aux ⇦T[i]
                                T[i] ⇦ T[i+1]      } Permuter
                                T[i+1] ⇦  Aux
                                Est_Trié⇦faux
                        FinSi
                FINPOUR
                borne⇦ borne-1
        FINTANTQUE
FIN
```

FIN
&
Merci