Série d'exercices N° 04

Les Tableaux

Exercice 1 :

Écrire un programme qui déclare, remplisse et affiche un tableau contenant les six voyelles de l'alphabet français

```
Algorithme Exercice1
Var
    V : tableau[6] de caractère
    i : entier
Début
  V[0] <- 'a'
  V[1] <- 'e'
  V[2] <- 'i'
  V[3] <- 'u'
  V[4] <- 'o'
  V[5] <- 'y'

  V <- ['a','e','i','u','o','y']
  ecrireln(V)
  pour i de 0 à 5 faire
      ecrire(V[i],", ")
  FinPour
Fin
```

Exercice 2 :

Écrire un programme qui déclare et remplisse un tableau de 10 valeurs numériques en les mettant toutes à 5.

```
Algorithme Exercice2
Var
    T : Tableau[10] de réel
    i : entier
Début
  T <- [5,5,5,5,5,5,5,5,5,5]
  pour i de 0 à 9 faire
      T[i] <- 5
  FinPour
  ecrire("T = [")
  pour i de 0 à 9 faire
      si i < 9 alors
          Ecrire(T[i], ", ")
      sinon
          Ecrire(T[i])
      FinSi
  FinPour
  ecrireln("]")
Fin
```

Exercice 3 :

Écrire un programme qui déclare et remplisse un tableau de N valeurs numériques en les mettant toutes à zéro.(N est un nombre demandé à l'utilisateur qui ne dépasse pas 100)

```
Algorithme Exercice3
Const
    M = 100
Var
    T : Tableau[M] de Réel
    i, N : Entier
Début
      repeter
          Ecrire("Donner le nombre des éléments du tableau : ")
          Lire(N)
          Si N > M alors
              Ecrireln("erreur : entrez un nombre <= ", M)
          FinSi
      jusqua N > 0 et N <= M

      pour i de 0 à N - 1 faire
          T[i] <- 0
      FinPour

      Ecrire("T = [")
      pour i de 0 à N - 1 faire
          ecrire(T[i] ,", ")
      FinPour
      ecrire("]")
Fin
```

Exercice 4 :

Écrire un algorithme qui déclare et remplisse un tableau de N valeurs numériques saisies au clavier. Puis afficher le Tableau. (N est un nombre demandé à l'utilisateur qui ne dépasse pas 100)

```
Algorithme Exercice4
Const
    M = 100
Var
    T : Tableau[M] de Réel
    i, N : Entier
Début
      repeter
          Ecrire("Donner le nombre des éléments du tableau : ")
          Lire(N)
          Si N > M alors
              Ecrireln("erreur : entrez un nombre <= ", M)
          FinSi
      jusqua N > 0 et N <= M

      pour i de 0 à N - 1 faire
          Ecrire("Donner l'élément N° ", i+1 , " : ")
          Lire(T[i] )
      FinPour

      Ecrire("T = [")
      pour i de 0 à N - 1 faire
```

```
            ecrire(T[i] ,", ")
        FinPour
        ecrire("]")
Fin
```

Exercice 5 :

Soit T un tableau de N entiers saisis par l'utilisateur. Écrire un programme calculant la somme, le produit et la moyenne des valeurs d'un tableau

```
Algorithme Exercice5
Const
    M=100
var
    i,N,S,P : entier
    Moy : réel
    T : Tableau[M] de entier
Début
      Ecrire("Donner N : ")
      Lire(N)

      // Remplissage du tableau
      pour i de 0 à N - 1 faire
          Ecrire("Entrez T[",i,"] : ")
          Lire(T[i])
      FinPour

      S <- 0
      P <- 1
      pour i de 0 à N - 1 faire
          S <- S + T[i]
          P <- P * T[i]
      FinPour
      Moy <- S / N
      ecrire("La somme est ", S)
      Ecrire("Le produit est ", P)
      Ecrire("La moyenne est ", Moy)
Fin
```

Exercice 6 :

Écrire un programme qui permet de :

- saisir N valeurs par l'utilisateur.
- Augmenter d'une valeur X donnée par l'utilisateur toutes les valeurs du tableau.
- Afficher le tableau à l'écran.

```
Algorithme Exercice6
var
    i,N : entier
    X : entier
    T : Tableau[9] de Entier
Début
      T <- [1,2,3,4,5,6,7,8,9]
      N <- 9
      Ecrire("T = [")
```

```
        pour i de 0 à N -1 faire
          Ecrire(T[i],", ")
      FinPour
      Ecrireln("]")

      ecrire("Donner X : ")
      Lire(X)
      pour i de 0 à N - 1 faire
          T[i] <- T[i] + X
      FinPour

      Ecrire("T = [")
      pour i de 0 à N -1 faire
          si i <> N- 1 alors
              Ecrire(T[i],", ")
          sinon
              Ecrire(T[i])
          FinSi
      FinPour
      Ecrireln("]")
Fin
```

Exercice 7 :

Écrire un programme qui remplit un tableau par des valeurs entrées au clavier. Incrémenter ensuite de 10% les éléments supérieurs à 100 et afficher le tableau résultant.

```
Algorithme Exercice7
var
    i,N : entier
    T : Tableau[9] de Entier
Début
      T <- [1,2,130,4,5,600,7,8,9]
      N <- 9
      Ecrire("T = [")
        pour i de 0 à N -1 faire
          Ecrire(T[i],", ")
      FinPour
      Ecrireln("]")

      pour i de 0 à N - 1 faire
          Si T[i] >= 100 alors
              T[i] <- T[i] * 1.1 //  T[i] <- T[i]  + T[i]*0.1
          FinSi
      FinPour

      Ecrire("T = [")
      pour i de 0 à N -1 faire
          si i <> N- 1 alors
              Ecrire(T[i],", ")
          sinon
              Ecrire(T[i])
          FinSi
      FinPour
      Ecrireln("]")
Fin
```

Exercice 8 :

Soit T un tableau de N entiers saisis par l'utilisateur. Écrire le programme qui détermine le plus petit élément (noté min) et le plus grand élément (noté max) de ce tableau, ainsi que leurs positions.

```
Algorithme Exercice8
Var
    i,N,Max,Min,Pmax,Pmin : entier
    T : Tableau[9] de entier
Début
      T <- [1,2,130,4,5,600,7,8,9]
      N <- 9
      Max <- T[0]
      Min <- T[0]
      Pmax <- 0
      Pmin <- 0
      pour i de 0 à N -1 faire
          si T[i] > Max alors
              Max <- T[i]
              Pmax <- i
          FinSi

             Si T[i] < Min alors
              Min<- T[i]
              Pmin <- i
          FinSi
      FinPour
      Ecrireln("Le max est ", Max, " et sa position est ", Pmax)
      Ecrire("Le min est ", Min, " et sa position est ", Pmin)
Fin
```

Exercice 9 :

Soit T un tableau de N notes saisis par l'utilisateur. Écrire le programme qui permet de :

- Calculer la somme des éléments de ce tableau, ainsi que leur moyenne.
- Afficher les notes qui sont supérieures à la moyenne.

```
Algorithme Exercice9
Const
    M = 100
Var
    T : Tableau[M] de Entier
    i,N,S : Entier
    Moy : Réel
Début
      repeter
          Ecrire("Donner le nombre des éléments : ")
          Lire(N)
          si N > M alors
              Ecrireln("Entrez un nombre inférieur ou égale à  ", M)
          FinSi
      Jusqua N <= M
      //Remplissage du tableau
      pour i de 0 à N-1 faire
          Ecrire("Entrez T[",i,"] : ")
          Lire(T[i])
```

```
      FinPour
      // Calculer la somme et la moyenne
      S <- 0
      pour i de 0 à N -1 faire
          S <- S + T[i]
      FinPour
      Moy <- S / N
      Ecrireln("La somme est ", S)
      Ecrireln("La moyenne est ", Moy)
      // Afficher les notes > à la moyenne
      Ecrireln("Les notes > à la moyenne sont :")
      pour i de 0 à N- 1 faire
          Si T[i] >= Moy alors
              Ecrireln(T[i])
          FinSi
      FinPour

Fin
```

Exercice 10 :

Écrire un programme qui demande la taille N d'un tableau T du type entier, remplit le tableau par des valeurs entrées au clavier et calcule et affiche ensuite la somme des valeurs positives, et la somme des valeurs négatives

```
Algorithme Exercice10
Const
    M = 100
Var
    T : Tableau[M] de Entier
    i,N,Sp,Sn : Entier
Début
      repeter
          Ecrire("Donner le nombre des éléments : ")
          Lire(N)
          si N > M alors
              Ecrireln("Entrez un nombre inférieur ou égale à  ", M)
          FinSi
      Jusqua N <= M
      //Remplissage du tableau
      pour i de 0 à N-1 faire
          Ecrire("Entrez T[",i,"] : ")
          Lire(T[i])
      FinPour

      Sp <- 0
      Sn <- 0
      pour i de 0 à N -1 faire
          Si T[i] > 0 alors
              Sp <- Sp + T[i]
          sinon
              Sn <- Sn + T[i]
          FinSi
      FinPour

      ecrireln("La somme des positifs est ", Sp)
      ecrireln("La somme des négatifs est ", Sn)
```

```
Fin
```

Exercice 11 :

Écrire un programme qui lit les notes de N étudiants et qui affiche le nombre d'étudiants admis ($Note >= 10$).

```
Algorithme Exercice11
Const
    M = 100
Var
    T : Tableau[M] de Entier
    i,N,nbAdmis : Entier
Début
      repeter
          Ecrire("Donner le nombre des éléments : ")
          Lire(N)
          si N > M alors
              Ecrireln("Entrez un nombre inférieur ou égale à  ", M)
          FinSi
      Jusqua N <= M
      //Remplissage du tableau
      pour i de 0 à N-1 faire
          Ecrire("Entrez T[",i,"] : ")
          Lire(T[i])
      FinPour
      nbAdmis <- 0
      pour i de 0 à N-1 faire
          Si T[i] >= 10 alors
              nbAdmis <- nbAdmis + 1
          FinSi
      FinPour
      ecrire("Le nombre des admis est ", nbAdmis)
Fin
```

Exercice 12 :

Écrire un algorithme constituant un tableau, à partir de deux tableaux de même longueur préalablement saisis. Le nouveau tableau sera la somme des éléments des deux tableaux de départ.

4	8	7	9	1	5	4	6
			+				
7	6	5	2	1	3	7	4
			=				
11	14	12	11	2	8	11	10

```
Algorithme Exercice12
Var
    T1,T2,T3 : tableau[8] de entier
    i,N : entier
Début
      N <- 8
      T1 <- [4,8,7,9,1,5,4,6]
      T2 <- [7,6,5,2,1,3,7,4]
      pour i de 0 à N -1 faire
          T3[i] <- T1[i] + T2[i]
      FinPour
```

```
      ecrire("T1 = [")
      pour i de 0 à N -1 faire
          Ecrire(T1[i],",")
      FinPour
      ecrireln("]")

      ecrire("T2 = [")
      pour i de 0 à N -1 faire
          Ecrire(T2[i],",")
      FinPour
      ecrireln("]")

        ecrire("T3 = [")
      pour i de 0 à N -1 faire
          Ecrire(T3[i],",")
      FinPour
      ecrireln("]")
Fin
```

Exercice 13 :

Soit T un tableau contenant N entiers. On propose d'écrire un programme qui permet d'éclater T en deux tableaux : TN (contenant les éléments négatifs de T) et TP (contenant les éléments positifs de T).

```
Algorithme Exercice13
var
    T,TN,TP : Tableau[8] de entier
    i,N,Nn,Np : entier
Début
      T <- [1,-1,2,-2,3,4,5,-5]
      N <- 8
      Nn <- 0
      Np <- 0
      pour i de 0 à N-1 Faire
          Si T[i] >= 0 alors
              TP[Np] <- T[i]
              Np <- Np+1
          sinon
              TN[Nn] <- T[i]
              Nn <- Nn + 1
          FinSi
      FinPour

      ecrire("T = [")
      pour i de 0 à N -1 faire
          Ecrire(T[i],",")
      FinPour
      ecrireln("]")

            ecrire("TP = [")
      pour i de 0 à Np -1 faire
          Ecrire(TP[i],",")
      FinPour
      ecrireln("]")

            ecrire("TN = [")
      pour i de 0 à Nn -1 faire
```

```
            Ecrire(TN[i],",")
        FinPour
        ecrireln("]")

Fin
```

Exercice 14 :

Soit T un tableau contenant N entiers. On propose d'écrire un programme qui permet d'éclater T en deux tableaux : TP (contenant les éléments pairs de T) et TI (contenant les éléments impairs de T).

```
Algorithme Exercice14
var
    T,TI,TP : Tableau[8] de entier
    i,N,Ni,Np : entier
Début
      T <- [1,-1,2,-2,3,4,5,-5]
      N <- 8
      Ni <- 0
      Np <- 0
      pour i de 0 à N-1 Faire
          Si T[i] mod 2 = 0 alors
              TP[Np] <- T[i]
              Np <- Np+1
          sinon
              TI[Ni] <- T[i]
              Ni <- Ni + 1
          FinSi
      FinPour

      ecrire("T = [")
      pour i de 0 à N -1 faire
          Ecrire(T[i],",")
      FinPour
      ecrireln("]")

            ecrire("TP = [")
      pour i de 0 à Np -1 faire
          Ecrire(TP[i],",")
      FinPour
      ecrireln("]")

            ecrire("TN = [")
      pour i de 0 à Ni -1 faire
          Ecrire(TI[i],",")
      FinPour
      ecrireln("]")

Fin
```

Exercice 15 :

Soit T un tableau contenant N entiers. On propose d'écrire un programme qui permet de chercher l'existence d'un élément V donné, dans la liste de valeurs de T.

```
Algorithme Exercice15
Var
    T : Tableau[8] de entier
```

```
    i,N,V : entier
    trouve : Booléen
Début
      T <- [1,9,13,20,26,70,55,100]
      N <- 8
      ecrire("Donner la valeur recherchée : ")
      Lire(V)
      trouve <- Faux
      pour i de 0 à N - 1 faire
          Si T[i] = V alors
              trouve <- Vrai
          FinSi
      FinPour
      Si trouve = Vrai alors
          ecrire(V, " existe dans le tableau")
      sinon
          ecrire(V, " n'existe pas dans le tableau")
      FinSi
Fin

Algorithme Exercice15V2
Var
    T : Tableau[8] de entier
    i,N,V : entier

Début
      T <- [1,9,13,20,26,70,55,100]
      N <- 8
      ecrire("Donner la valeur recherchée : ")
      Lire(V)
      i <- 0
      tantque T[i] <> V et i < N -1 faire
          i <- i + 1
      FinTantQue
      Si T[i] = V  alors
          ecrire(V, " existe dans le tableau")
      sinon
          ecrire(V, " n'existe pas dans le tableau")
      FinSi
Fin
```

Exercice 16 :

Écrire un programme qui fait remplir un tableau T par les résultats de 100 lancements d'un dé. Le programme doit faire remplir par la suite un tableau fréquence F par le nombre de fois que chaque face est obtenue.

```
Algorithme Exercice16
Var
    T1 : tableau[100] de entier
    T2 : tableau[7] de entier
    i,index : entier
Début
      pour i de 0 à 99 faire
          T1[i] <- alea(1,6)
      FinPour
      T2 <-[0,0,0,0,0,0,0]
      pour i de 0 à 99 faire
```

```
          index <- T1[i]
          T2[index] <- T2[index] + 1
      FinPour

      pour i de 1 à 6 faire
          ecrireln(i," : " , T2[i],"%")
      FinPour
Fin
```

Exercice 17 :

Soit T un tableau contenant N entiers. Écrire un programme qui Insère une valeur X donnée au clavier dans la dernière case du tableau T de manière à obtenir un tableau de N+1 valeurs.

```
Algorithme Exercice17
Var
    T : Tableau[100] de entier
    i,N,X : entier
Début
  ecrire("Donner N : ")
  Lire(N)
  pour i de 0 à N -1 faire
      T[i] <- alea(0,9)
  FinPour
  ecrire("Donner la valeur à Ajouter : ")
  Lire(X)
  T[N] <- X
  N <- N + 1
  ecrire("T = [")
   pour i de 0 à N -1 faire
          Ecrire(T[i],",")
   FinPour
   ecrire("]")
Fin
```

Exercice 18 :

Soit T un tableau contenant N entiers. On propose d'écrire un programme qui permet d'inverser les éléments de T (permuter T [1] et T[n], puis T [2] et T [n-1],...).

```
Algorithme Exercice18
Var
    T : Tableau[10] de entier
    i,N,tmp : entier
Début
      T <- [0,1,2,3,4,5,6,7,8,9]
      N <- 10
      pour i de 0 à (N-1) / 2 faire
          tmp <- T[i]
          T[i] <- T[N-1-i]
          T[N-1-i] <- tmp
      FinPour
   ecrire("T = [")
   pour i de 0 à N -1faire
          Ecrire(T[i],",")
   FinPour
   ecrire("]")
Fin
```

Exercice 19 :

Écrire un programme qui pour chaque élément d'un tableau T ne garde que sa première occurrence et on remplace les autres par 0.

```
Algorithme Exercice19
Var
    T : tableau[10] de entier
    i,j,N : entier
Début
      T <- [1,2,3,1,2,2,3,5,5,6]
      N <- 10
        ecrire("T = [")
   pour i de 0 à N -1 faire
          Ecrire(T[i],",")
   FinPour
   ecrireln("]")
      pour i de 0 à N- 1 faire
          pour j de i+ 1 à N-1 faire
               Si T[i] = T[j] alors
                   T[j] <- 0
               FinSi
          Finpour
      FinPour
        ecrire("T = [")
   pour i de 0 à N -1 faire
          Ecrire(T[i],",")
   FinPour
   ecrireln("]")
Fin
```

Exercice 20 :

Écrire un programme qui permet d'insérer un élément dans un tableau à une position spécifiée. Le programme doit également affiche un message d'erreur si la position d'insertion n'est pas valide.

```
Algorithme Exercice20
Var
    T : tableau[10] de entier
    i,N,V,P : entier
Début
      T <- [1,2,3,4,6,7,8,9]
      N <- 8
    ecrire("T = [")
   pour i de 0 à N -1 faire
          Ecrire(T[i],",")
   FinPour
   ecrireln("]")
      Ecrire("Donner V : ")
      Lire(V)
      Ecrire("Donner P : ")
      Lire(P)
      pour i de N à P+1 pas -1 faire
          T[i] <- T[i-1]
      FinPour
      T[P] <- V
      N <- N + 1
   ecrire("T = [")
```

```
   pour i de 0 à N -1 faire
          Ecrire(T[i],",")
   FinPour
   ecrireln("]")

Fin
```

Exercice 21 :

Écrire un programme qui permet de supprimer un élément dans un tableau à une position spécifiée. Le programme doit également affiche un message d'erreur si la position de suppression n'est pas valide.

```
Algorithme Exercice21
Var
    T : Tableau[10] de entier
    i,N,P : Entier
Début
      N <- 10
      T <- [0,1,2,3,4,5,6,7,8,9]
      //Afficher le tableau avant la suppression
         Ecrire("T = [")
       pour i de 0 à N -1 faire
          Ecrire(T[i],",")
       FinPour
       Ecrireln("]")

      repeter
          Ecrire("Entrez la position de l'élément à supprimer : ")
          Lire(P)
          si p<0 ou P >= N alors
              Ecrireln("La position doit être comprise entre 0 et ",N-1)
          FinSi
      jusqua P>=0 et P<N

      // Supprimer l'élément à la position P
      pour i de P à N-2 faire
          T[i] <- T[i+1]
      FinPour

            //Afficher le tableau Après la suppression
         Ecrire("T = [")
       pour i de 0 à N -1 faire
          Ecrire(T[i],",")
       FinPour
       Ecrireln("]")
Fin
```

Exercice 22 :

Écrire un programme qui pour chaque élément d'un tableau T ne garde que sa première occurrence et on supprime les autre.

```
Algorithme Exercice22
Var
   T : Tableau[10] de entier
    i,j,k,N : Entier
Début
      N <- 10
```

```
      T <- [1,2,3,1,2,4,1,3,2,5]
      //Afficher le tableau avant la suppression
         Ecrire("T = [")
       pour i de 0 à N -1 faire
          Ecrire(T[i],",")
       FinPour
       Ecrireln("]")
      i <-0
      tantque i < N -1 faire
      j <- i +1
      tantque T[i] <>T[j] et j < N  faire
          j <- j+1
      Fintantque
      Si T[i] = T[j] alors
            k <-j
            tantque k < N-1 faire
                T[k] <- T[k+1]
                k <- k +1
            FinTantQue
        N <- N -1
      sinon
          i <- i +1
      FinSi
      FinTantQue
      //Afficher le tableau Après la suppression
        Ecrire("T = [")
       pour i de 0 à N -1 faire
          Ecrire(T[i],",")
       FinPour
       Ecrireln("]")
Fin
```

Exercice 23 :

Écrire un programme qui lit la taille N de deux tableaux T1 et T2 du type entier, remplit les tableaux par des valeurs entrées au clavier, puis compter et afficher le nombre d'élément tels que T1(i) = T2(i).

```
Algorithme Exercice23
var
    T1,T2  : Tableau[10] de entier
    i,N,NB : entier
Début
      T1 <- [1,3,7,12,20,7,4,0,9,6]
      T2 <- [5,3,7,16,20,7,66,44,9,23]
      NB <- 0
      pour i de 0 à N-1 faire
          si T1[i] = T2[i] alors
              NB <- NB + 1
          FinSi
      FinPour
      ecrire("Le nombre des éléments égaux au même indice est ", NB)
Fin
```

Exercice 24 :

Écrire un programme qui demande à l'utilisateur de saisir les notes d'une classe et qui calcule et affiche ensuite le pourcentage de notes supérieures à la moyenne de la classe.

```
Algorithme Exercice24
Const
    M = 100
Var
    T : Tableau[M] de Réel
    i, N,Nb : Entier
    S,Moy : réel
Début
      repeter
          Ecrire("Donner le nombre des etudiants : ")
          Lire(N)
          Si N > M alors
              Ecrireln("erreur : entrez un nombre <= ", M)
          FinSi
      jusqua N > 0 et N <= M

      pour i de 0 à N - 1 faire
          Ecrire("Donner la note ", i+1 , " : ")
          Lire(T[i] )
      FinPour
      S<- 0
      pour i de 0 à N-1 faire
          S <- S + T[i]
      FinPour
       Moy <- S / N
       Nb <- 0
       pour i de 0 à N-1 faire
           Si T[i] >= Moy alors
               Nb <- Nb +1
           FinSi
       FinPour
       Ecrire("Le nombre des notes supérieurs à la moyenne da la classe est ", Nb)

Fin
```

Exercice 25 :

On dispose d'un tableau MOY qui contient la liste des moyennes de N élèves. On propose d'écrire un programme qui permet de déterminer et d'afficher le rang de chaque élève.

```
Algorithme Exercice25
Var
    T : Tableau[10] de réel
    Rang : Tableau[10] de entier
    i,j,N : entier
Début
      T <-[17,12,10,18,14,11,20,6,9,17]
      N <- 10
      pour i de 0 à N-1 faire
          Rang[i] <- 0
              pour j de 0 à N -1 Faire
                  Si T[j] >= T[i] alors
                      Rang[i] <- Rang[i] +1
                  FinSi
              FinPour
      FinPour
      pour i de 0 à N-1 Faire
          ecrireln(T[i] , " est en classement N° : ", Rang[i])
```

```
      FinPour
Fin
```

Exercice 26 :

On dispose d'un tableau T rempli par N caractères. Écrire un programme permettant d'insérer un caractère C donné à la k^{ieme} position (avec $K <= N$).

```
Algorithme Exercice26
Var
    T :Tableau[12] de caractère
    i,N,p : entier
    C : caractère
Début
      T <-['a','b','c','d','e','f','g','h','i','j']
      N <- 10
        Ecrire("T = [")
       pour i de 0 à N -1 faire
          Ecrire(T[i],",")
       FinPour
       Ecrireln("]")
      Ecrire("entrez la caracère à insérer : ")
      Lire(C)
      répéter
          Ecrire("Entrer la postion d'insertion : ")
          Lire(p)
          Si p < 0 ou p >= N alors
              Ecrireln("erreur : entrez une position comprise entre 0 et ", N-1)
          FinSi
      jusqua p >= 0 et p<N

      N <- N + 1
      pour i de N-1 à p+1 pas -1 faire
          T[i] <- T[i-1]
      FinPour
      T[p] <- C
      Ecrire("T = [")
       pour i de 0 à N -1 faire
          Ecrire(T[i],",")
       FinPour
       Ecrireln("]")
Fin
```

Exercice 27 :

Étant donné un tableau d'entiers non nuls, trouvez combien il y a d'entiers X distincts positifs dont l'opposé (-X) est aussi dans le tableau

```
Algorithme Exercice27
Var
    T : Tableau[10] de entier
    i,j,N,Nb : entier
Début
      T = [1,2, -1, 3, -2, 4, 5, 6, 7,-6]
      N <- 10
      Nb <- 0
      pour i de 0 à N -1 faire
          si T[i] > 0 alors
```

```
            j <- 0
            tantque j < N-1 et T[i] <> - T[j] faire
                j <- j + 1
            FinTantQue
            Si T[i] = - T[j] alors
                Nb <- Nb +1
            FinSi
        FinSi
    FinPour
    Ecrire("Le nombre des éléments dont leur opposé se trouve dans le tableau est ", Nb)
Fin
```

Exercice 28 :

Écrire un programme qui demande à l'utilisateur de remplir un tableau de N entiers, puis le programme détermine et affiche les éléments uniques du tableau.

```
Algorithme Exercice28
Var
    T,U : Tableau[100] de entier
    i,j,N,Nu : entier
Début
    T = [1,2,3,4,5,2,7,6,3,8,1]
    N <- 11
    Nu <- 0
    pour i de 0 à N -2 faire
        j <- i +1
        tantQue T[i] <> T[j] et j < N-1 Faire
            j <- j+ 1
        FinTantQue
        Si T[i] <> T[j] alors
            U[Nu] <- T[i]
            Nu <- Nu +1
        FinSi
    FinPour
       Ecrire("T = [")
     pour i de 0 à N -1 faire
        Ecrire(T[i],",")
     FinPour
     Ecrireln("]")

     Ecrire("U = [")
     pour i de 0 à Nu -1 faire
        Ecrire(U[i],",")
     FinPour
     Ecrireln("]")
Fin
```

Exercice 29 :

Écrire un programme qui permit de trier un tableau T de N éléments

```
Algorithme Exercice29
Var
    T : Tableau[100] de entier
    i,j,N,tmp : entier
Début
```

```
    T = [1,9,5,7,2,4,3,8,0,6]
    N <- 10
    Ecrire("T = [")
       pour i de 0 à N -1 faire
          Ecrire(T[i],",")
       FinPour
    Ecrireln("]")

    // Trier le tableau
    pour i de 0 à N-2 faire
        Pour j de i+1 à N-1 Faire
            Si T[i] > T[j] Alors
                tmp <- T[i]
                T[i] <- T[j]
                T[j] <- tmp
            FinSi
        FinPour
    FinPour
    Ecrire("T = [")
       pour i de 0 à N -1 faire
          Ecrire(T[i],",")
       FinPour
    Ecrireln("]")

Fin
```

Exercice 30 :

Écrire un programme qui vérifie si un nombre est palindrome ou non (un nombre palindrome peut se lire indifféremment de gauche à droit ou de droite à gauche, exemple 161)

```
Algorithme Exercice30
Var
    T : Tableau[100] de Entier
    i,j,N,Nb,R : entier
Début
      Ecrire("Entrez un nombre : ")
      Lire(Nb)
      R <- Nb
      N <- 0
      tantque R <> 0 Faire
          T[N] <- R mod 10
          R <- R Div 10
          N <- N+ 1
      FinTantQue

      i <- 0
      j <- N-1
      Tantque T[i] = T[j] et i < j Faire
          i <- i+1
          j <- j-1
      FinTantQue
      Si T[i] = T[j] alors
          Ecrire(Nb, " est un prindrome")
      Sinon
          Ecrire(Nb, " n'est pas un prindrome")
      FinSi
```

```
Fin
```

Exercice 31 :

Une chaîne de caractère peut se présenter comme un tableau de caractères. Déterminer si une chaîne de caractère est un palindrome
Exemples : Esope reste ici et se repose

```
Algorithme Exercice31
Var
    ch : chaine
    i,j : entier
Début
      Ecrire("Entrez une chaîne de caractères : ")
      Lire(ch)

      i <- 0
      j <- Long(ch) -1
      tantque ch[i] = ch[j] et i < j Faire
          i <- i+1
          j <- J-1
      FinTantQue
      Si ch[i] = ch[j] alors
          Ecrire(ch, " est un prindrome")
      Sinon
          Ecrire(ch, " n'est pas un prindrome")
      FinSi
Fin
```