ALGORITHME DE TRI

LANGAGE C – LES TABLEAUX

TRI A BULLE

Soit T un tableau de n entiers. La méthode de tri à bulles nécessite deux étapes :

- Parcourir les éléments du tableau de 1 à (n–1) ; si l'élément i est supérieur à l'élément (i+1), alors on les permute.
- Le programme s'arrête lorsqu'aucune permutation n'est réalisable après un parcours complet du tabeau.

```
Procédure Tri_Bulle (Var T : Tab)
Variables
        i, x : Entier
        échange : Booléen
Début
        Répéter
                échange ← Faux
                Pour i de 1 à (n-1) Faire
                        Si (T[i] > T[i+1]) Alors
                                x ← T[i]
                                T[i] ← T[i+1]
                                T[i+1] ← x
                                échange ← Vrai
                        FinSi
                FinPour
        Jusqu'à (échange = Faux)
Fin
```

Tableau initial	6	4	3	5	2
Après la 1ère itération	4	3	5	2	6
Après la 2ème itération	3	4	2	5	6
Après la 3ème itération	3	2	4	5	6
Après la 4ème itération	2	3	4	5	6

TRI PAR SELECTION

Soit T un tableau de n entiers. La méthode de tri à bulles nécessite deux étapes :

- Parcourir les éléments du tableau de 1 à (n–1) ; si l'élément i est supérieur à l'élément (i+1), alors on les permute.
- Le programme s'arrête lorsqu'aucune permutation n'est réalisable après un parcours complet du tabeau.

```
Procédure Tri_Bulle (Var T : Tab)
Variables
        i, x : Entier
        échange : Booléen
Début
        Répéter
                échange ← Faux
                Pour i de 1 à (n-1) Faire
                        Si (T[i] > T[i+1]) Alors
                                x ← T[i]
                                T[i] ← T[i+1]
                                T[i+1] ← x
                                échange ← Vrai
                        FinSi
                FinPour
        Jusqu'à (échange = Faux)
Fin
```

Trace d'éxécution

Tableau initial	6	4	2	3	5
Après la 1ère itération	2	4	6	3	5
Après la 2ème itération	2	3	6	4	5
Après la 3ème itération	2	3	4	6	5
Après la 4ème itération	2	3	4	5	6

TRI PAR SELECTION

tri selection.c

```c
#include <stdio.h>

int main()
{
int T[10] = { 3, -2, 7, 10, -5, 22, 1, 27, 25, 30};
int i,j,c;

for(i=0;i<9;i++)
    for(j=i+1;j<10;j++)

        if ( T[i] > T[j] ) {
            c = T[i];
            T[i] = T[j];
            T[j] = c;
        }

        printf("\n******** tableau triée par ordre croissant ********\n");

  for (i=0; i < 10; i++)
     printf("%4d", T[i]);

        getch();
  return 0;
}
```

TRI PAR INSERTION

Cette méthode consiste à prendre les éléments de la liste un par un et insérer chacun dans sa bonne place de façon que les éléments traités forment une sous-liste triée.

Pour ce faire, on procède de la façon suivante :

- comparer et permuter si nécessaire T[1] et T[2] de façon à placer le plus petit dans la case d'indice 1
- comparer et permuter si nécessaire l'élément T[3] avec ceux qui le précèdent dans l'ordre (T[2] puis T[1]) afin de former une sous-liste triée T[1..3]
- …
- comparer et permuter si nécessaire l'élément T[n] avec ceux qui le précèdent dans l'ordre (T[n-1], T[n-2], …) afin d'obtenir un tableau trié.

Trace d'éxécution

Tableau initial	6	4	3	5	2
Après la 1ère itération	4	6	3	5	2
Après la 2ème itération	3	4	6	5	2
Après la 3ème itération	3	4	5	6	2
Après la 4ème itération	2	3	4	5	6

```
Procédure Tri_Insertion(Var T : Tab)
Variables
        i, j, x, pos : Entier
Début
        Pour i de 2 à n Faire
                pos ← i - 1
                TantQue (pos>=1) et (T[pos]>T[i]) Faire
                        pos ← pos – 1
                FinTQ
                pos ← pos + 1
                x ← T[i]
                Pour j de (i-1) à pos [pas = -1] Faire
                        T[j+1] ← T[j]

                FinPour
                T[pos] ← x
        FinPour
Fin
```

TRI PAR INSERTION

tri insertion.c

```c
#include <stdio.h>


int main()
{
  int tab[10] = { 3, -2, 7, 10, -5, 22, 1, 27, 25, 30};
  int i, j, tmp;

   //afficher les éléments du tableau
   for (i=0; i < 10; ++i)
   {
      printf("%4d", tab[i]);
   }

  for (i=1 ; i <= 9; i++) {
    j = i;

    while (j > 0 && tab[j-1] > tab[j]) {
      tmp = tab[j];
      tab[j] = tab[j-1];
      tab[j-1] = tmp;

      j--;
    }
  }

  printf("\n******** tableau triée par ordre croissant ********\n");

  for (i=0; i < 10; i++)
     printf("%4d", tab[i]);
 getch();
  return 0;
}
```