

Tri à bulles

Principe

Soit T un tableau d'entiers. Le tri à bulles consiste à :

- 1). Comparer, deux à deux, les éléments consécutifs d'un tableau ($T[i]$ et $T[i+1]$).
- 2). Effectuer une permutation si $T[i] > T[i+1]$.
- 3). On continue de trier jusqu'à ce qu'il n'y ait plus de permutation.

Code C de l'algorithme tri à Bulles

```
do
{
    permut = 0;          // initialement pas de permutation
    for( i = 0 ; i < n-1 ; i++ )
    {
        if ( T [i] > T [i+1] )
        { // permutation
            aux = T[i];
            T[i] = T[i+1];
            T[i+1] = aux ;
            permut = 1;      // on a fait une permutation
        }
    }
}
while (permut == 1);
```

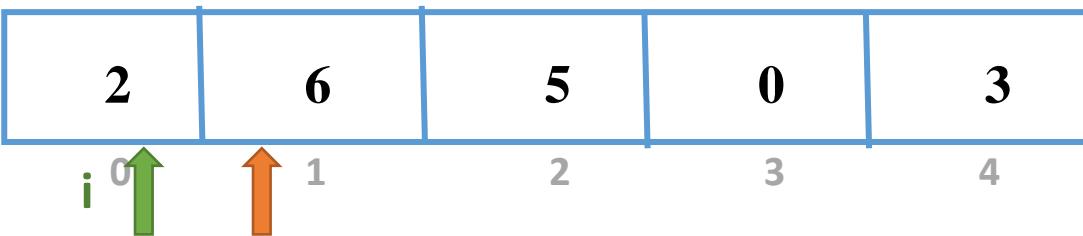
Exemple d'exécution

Soit le tableau T suivant composé de 5 entiers :

```
T          n = 5
indices
do
{
    permut = 0;
    for( i = 0 ; i < n-1 ; i++ )
    {
        if ( T [i] > T [i+1] )
            { // permutation
                aux = T[i];
                T[i] = T[i+1];
                T[i+1] = aux ;
                permut = 1;
            }
    }
}
while (permut == 1);
```

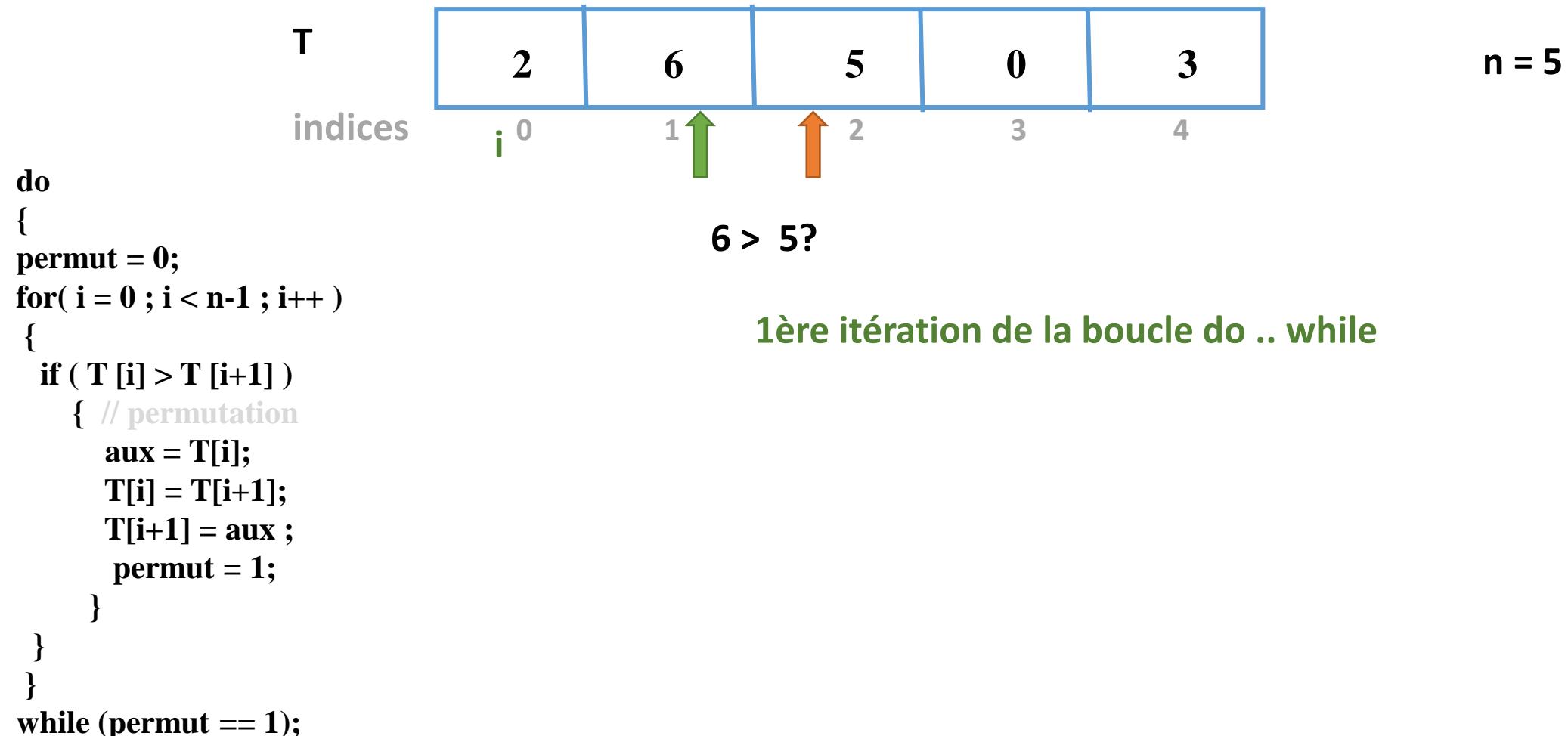
2 > 6?

1ère itération de la boucle do .. while



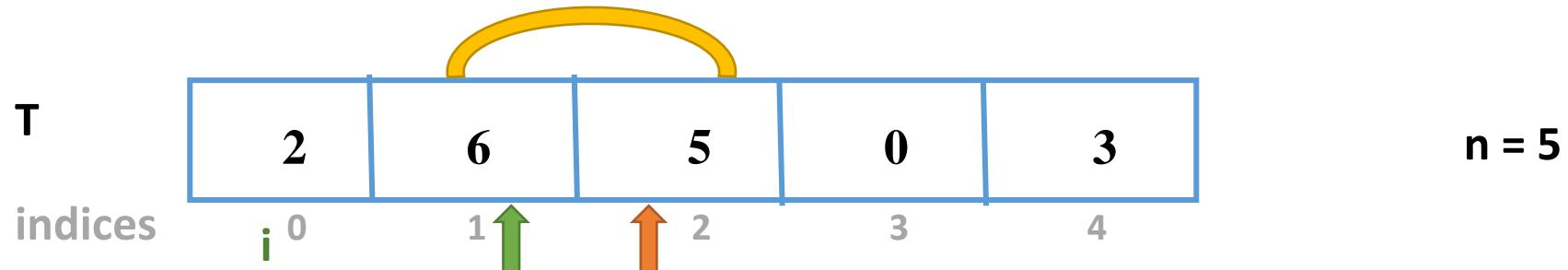
Exemple d'exécution

Soit le tableau T suivant composé de 5 entiers :



Exemple d'exécution

Soit le tableau T suivant composé de 5 entiers **Permutation**



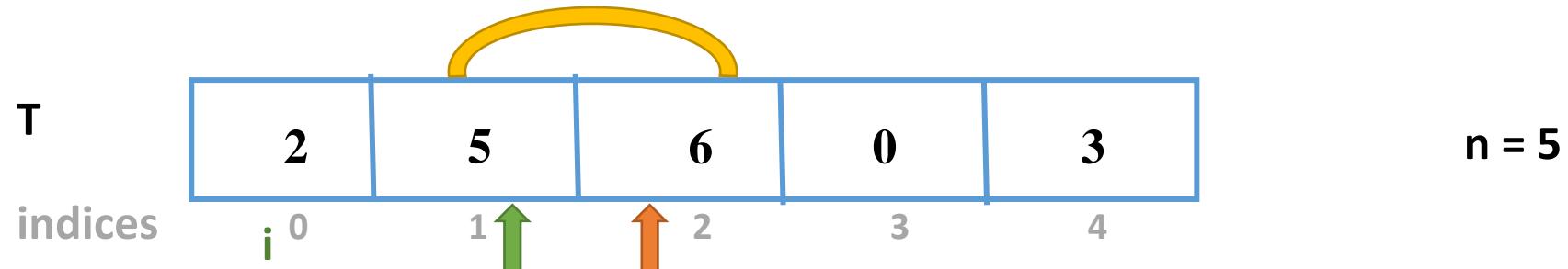
```
do
{
    permut = 0;
    for( i = 0 ; i < n-1 ; i++ )
    {
        if ( T [i] > T [i+1] )
        { // permutation
            aux = T[i];
            T[i] = T[i+1];
            T[i+1] = aux ;
            permut = 1;
        }
    }
    while (permut == 1);
```

6 > 5?

1ère itération de la boucle do .. while

Exemple d'exécution

Soit le tableau T suivant composé de 5 entiers **Permutation**

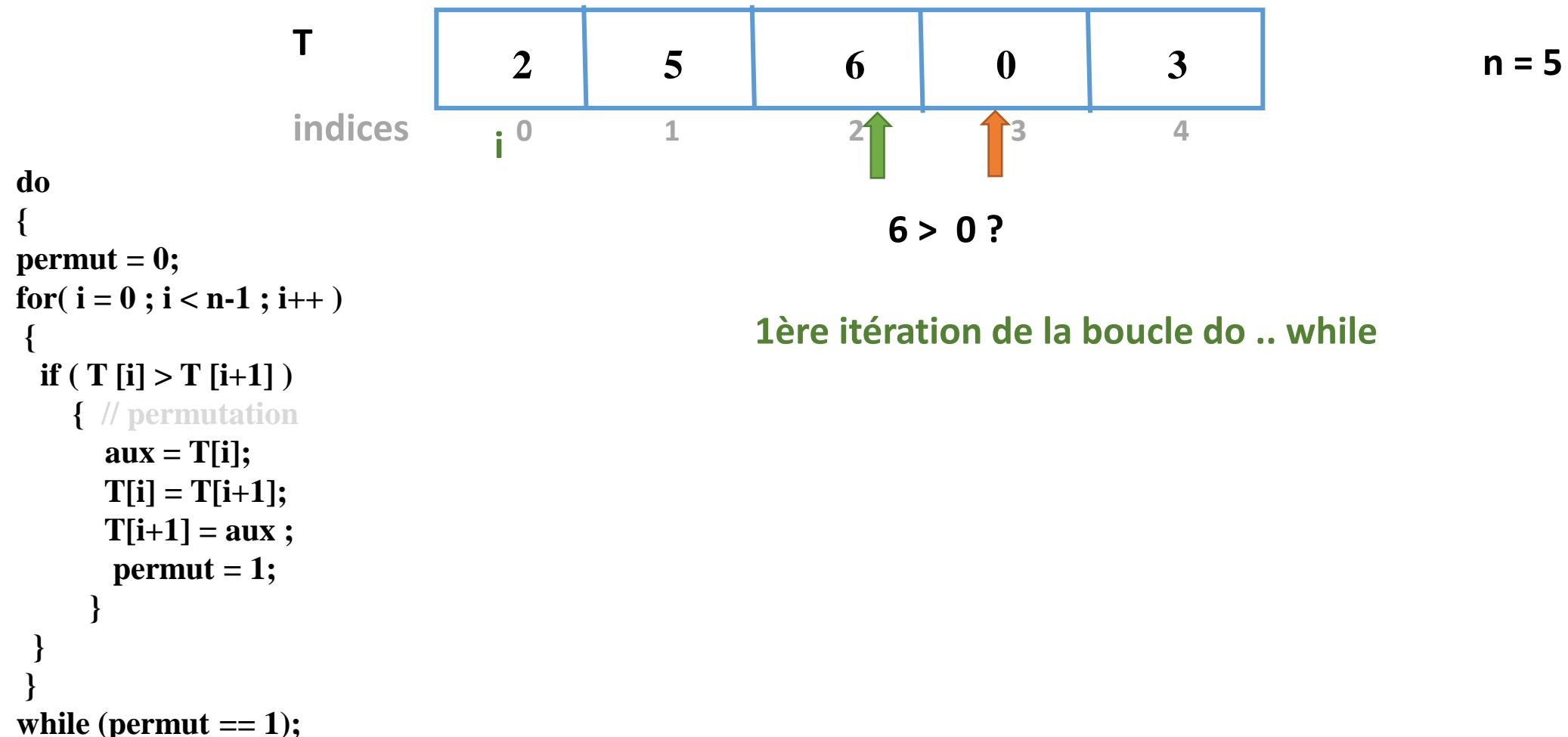


```
do
{
    permut = 0;
    for( i = 0 ; i < n-1 ; i++ )
    {
        if ( T [i] > T [i+1] )
        { // permutation
            aux = T[i];
            T[i] = T[i+1];
            T[i+1] = aux ;
            permut = 1;
        }
    }
    while (permut == 1);
```

1ère itération de la boucle do .. while

Exemple d'exécution

Soit le tableau T suivant composé de 5 entiers :



Exemple d'exécution

Soit le tableau T suivant composé de 5 entiers :

```
T
indices      i 0   1   2   3   4
          2   5   6   0   3
do
{
permut = 0;
for( i = 0 ; i < n-1 ; i++ )
{
  if ( T [i] > T [i+1] )
    { // permutation
      aux = T[i];
      T[i] = T[i+1];
      T[i+1] = aux ;
      permut = 1;
    }
}
while (permut == 1);
```

Permutation

6 > 0 ?

1ère itération de la boucle do .. while

n = 5

Exemple d'exécution

Soit le tableau T suivant composé de 5 entiers :

```
T
indices      i 0   1   2   3   4
          2   5   0   6   3
do
{
permut = 0;
for( i = 0 ; i < n-1 ; i++ )
{
  if ( T [i] > T [i+1] )
    { // permutation
      aux = T[i];
      T[i] = T[i+1];
      T[i+1] = aux ;
      permut = 1;
    }
}
while (permut == 1);
```

Permutation

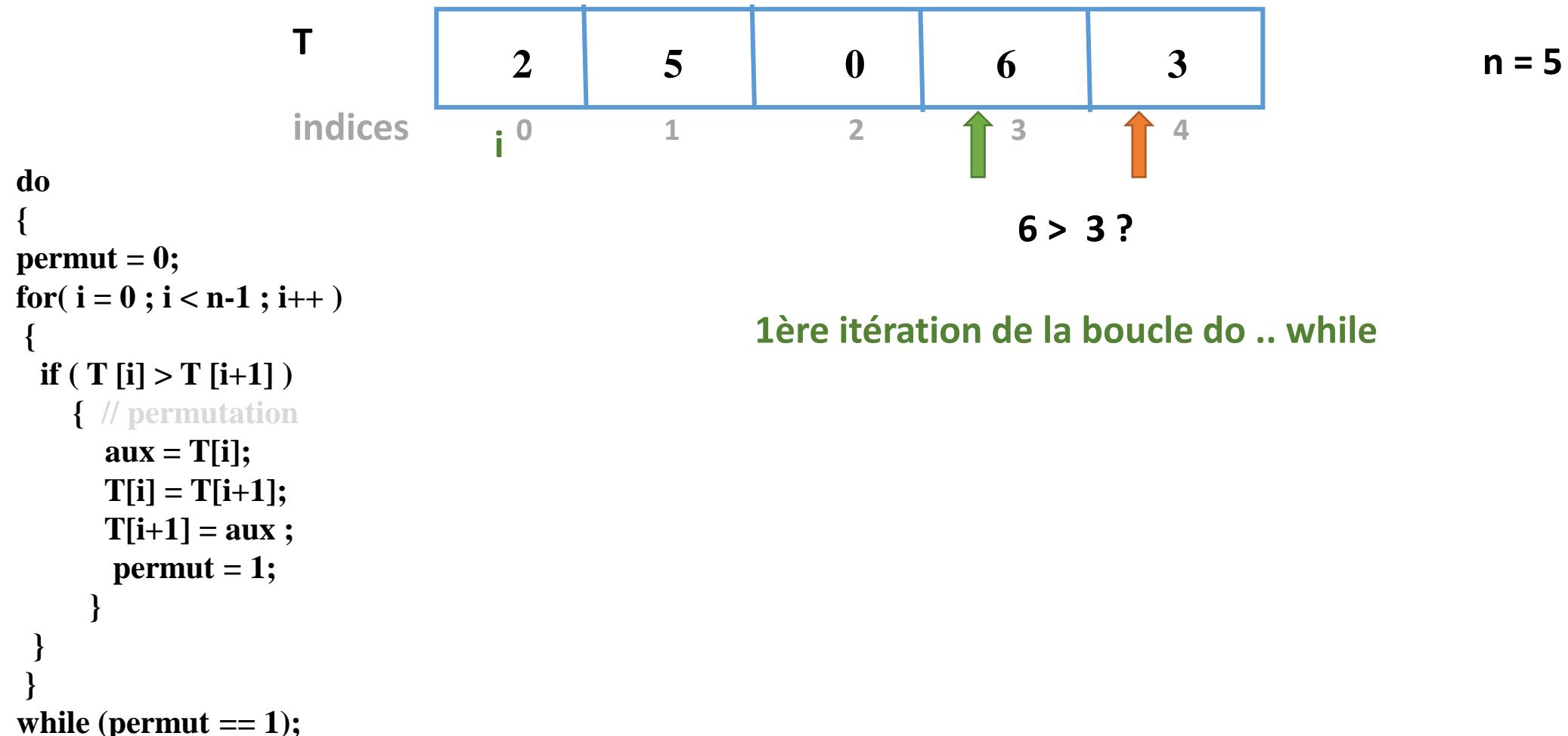
6 > 0 ?

1ère itération de la boucle do .. while

n = 5

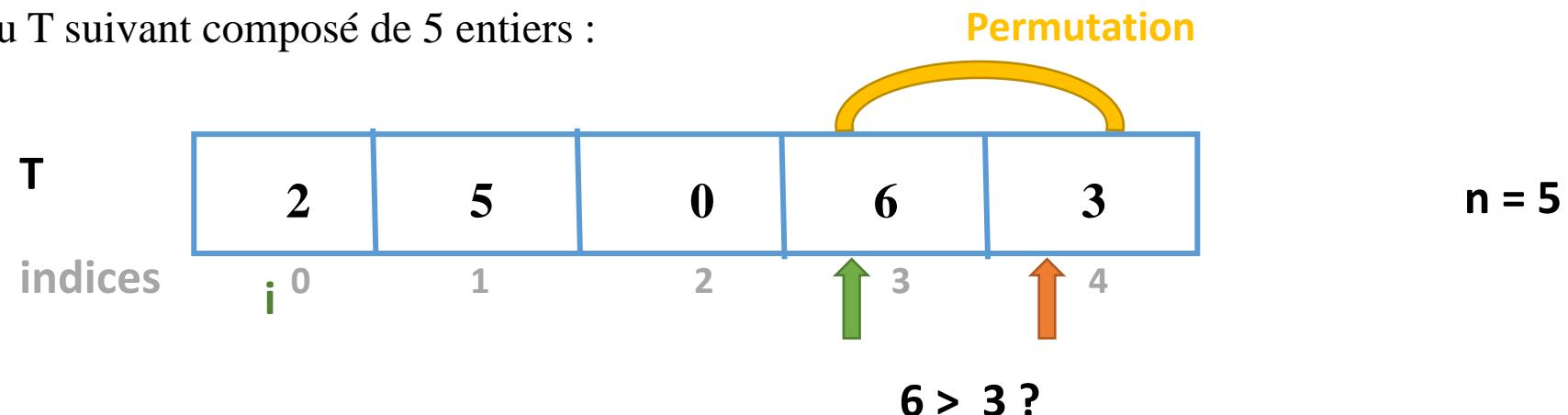
Exemple d'exécution

Soit le tableau T suivant composé de 5 entiers :



Exemple d'exécution

Soit le tableau T suivant composé de 5 entiers :



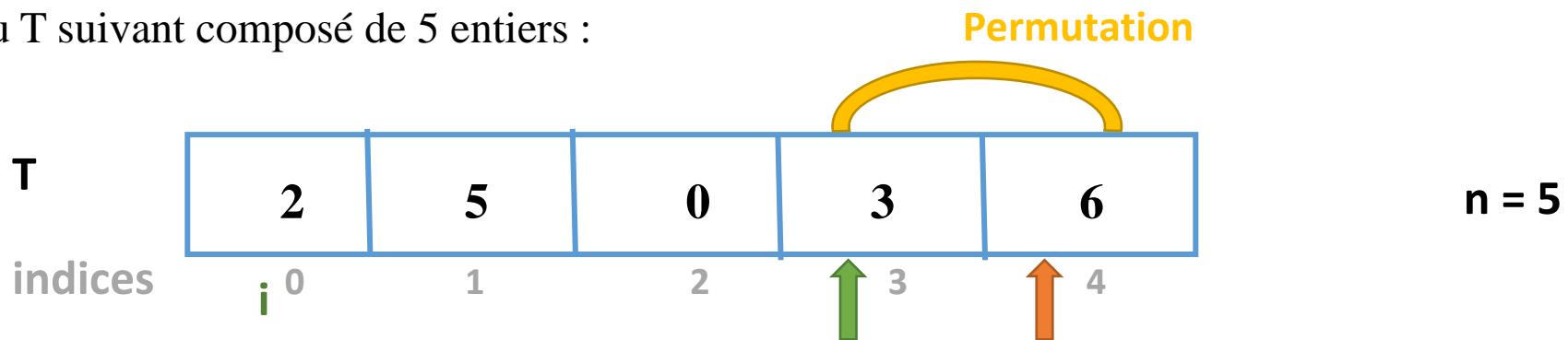
```
do
{
    permut = 0;
    for( i = 0 ; i < n-1 ; i++ )
    {
        if ( T [i] > T [i+1] )
            { // permutation
                aux = T[i];
                T[i] = T[i+1];
                T[i+1] = aux ;
                permut = 1;
            }
    }
    while (permut == 1);
```

1ère itération de la boucle do .. while

Exemple d'exécution

Soit le tableau T suivant composé de 5 entiers :

```
do
{
permut = 0;
for( i = 0 ; i < n-1 ; i++ )
{
    if ( T [i] > T [i+1] )
        { // permutation
            aux = T[i];
            T[i] = T[i+1];
            T[i+1] = aux ;
            permut = 1;
        }
}
while (permut == 1);
```



$6 > 3 ?$

1ère itération de la boucle do .. while



Le plus grand élément se trouve à la dernière case du tableau

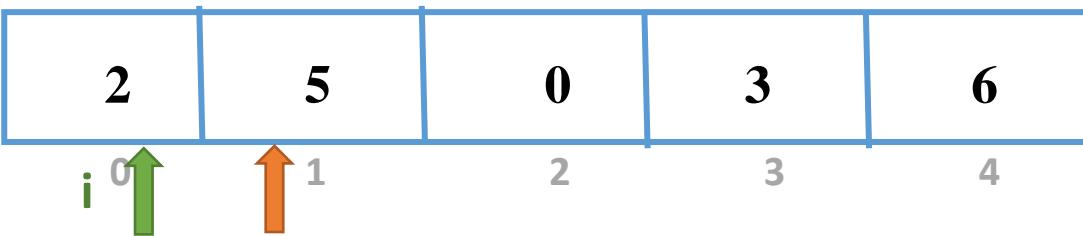
Exemple d'exécution

Soit le tableau T suivant composé de 5 entiers :

```
T          n = 5
indices
do
{
permut = 0;
for( i = 0 ; i < n-1 ; i++ )
{
    if ( T [i] > T [i+1] )
        { // permutation
            aux = T[i];
            T[i] = T[i+1];
            T[i+1] = aux ;
            permut = 1;
        }
}
while (permut == 1);
```

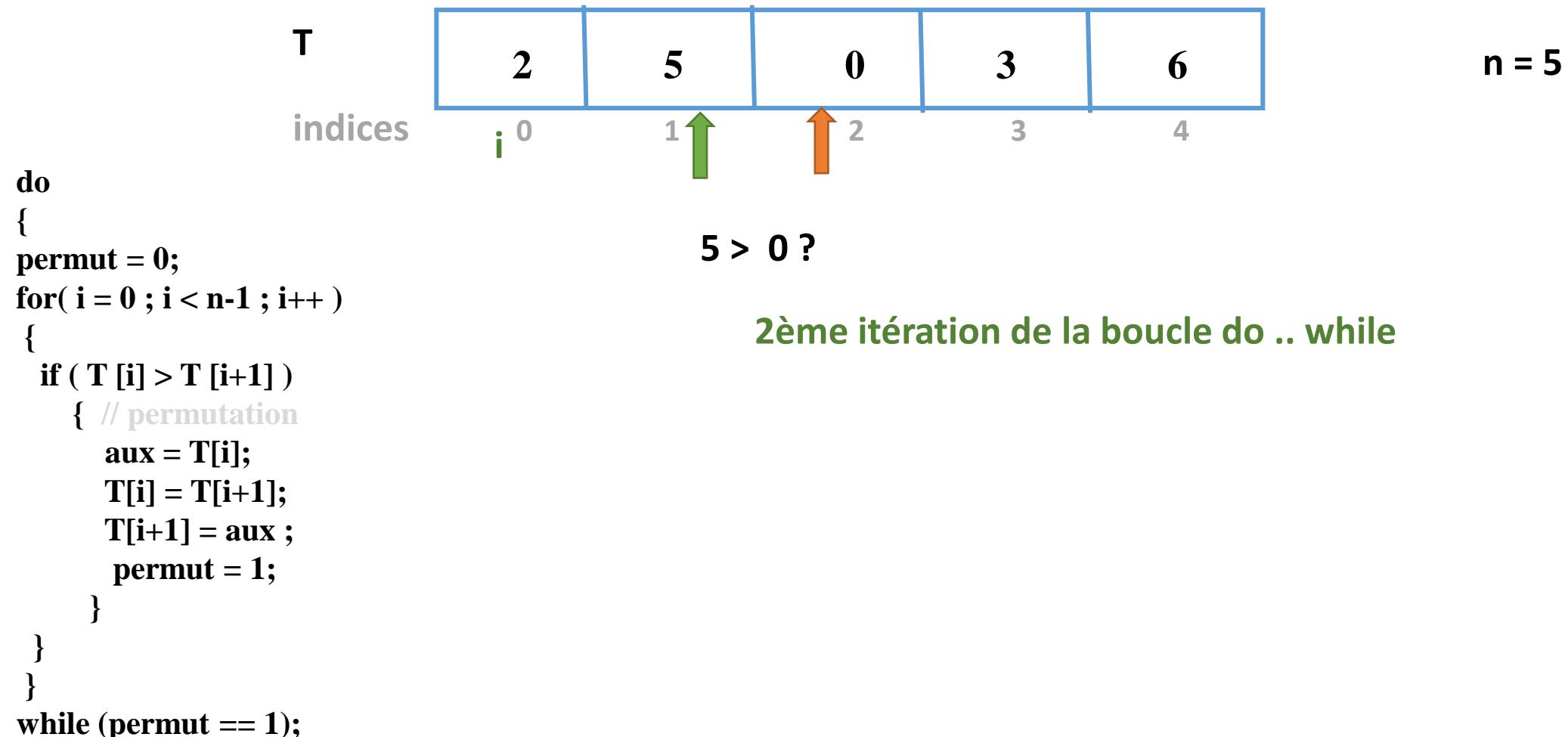
2 > 5 ?

2ème itération de la boucle do .. while



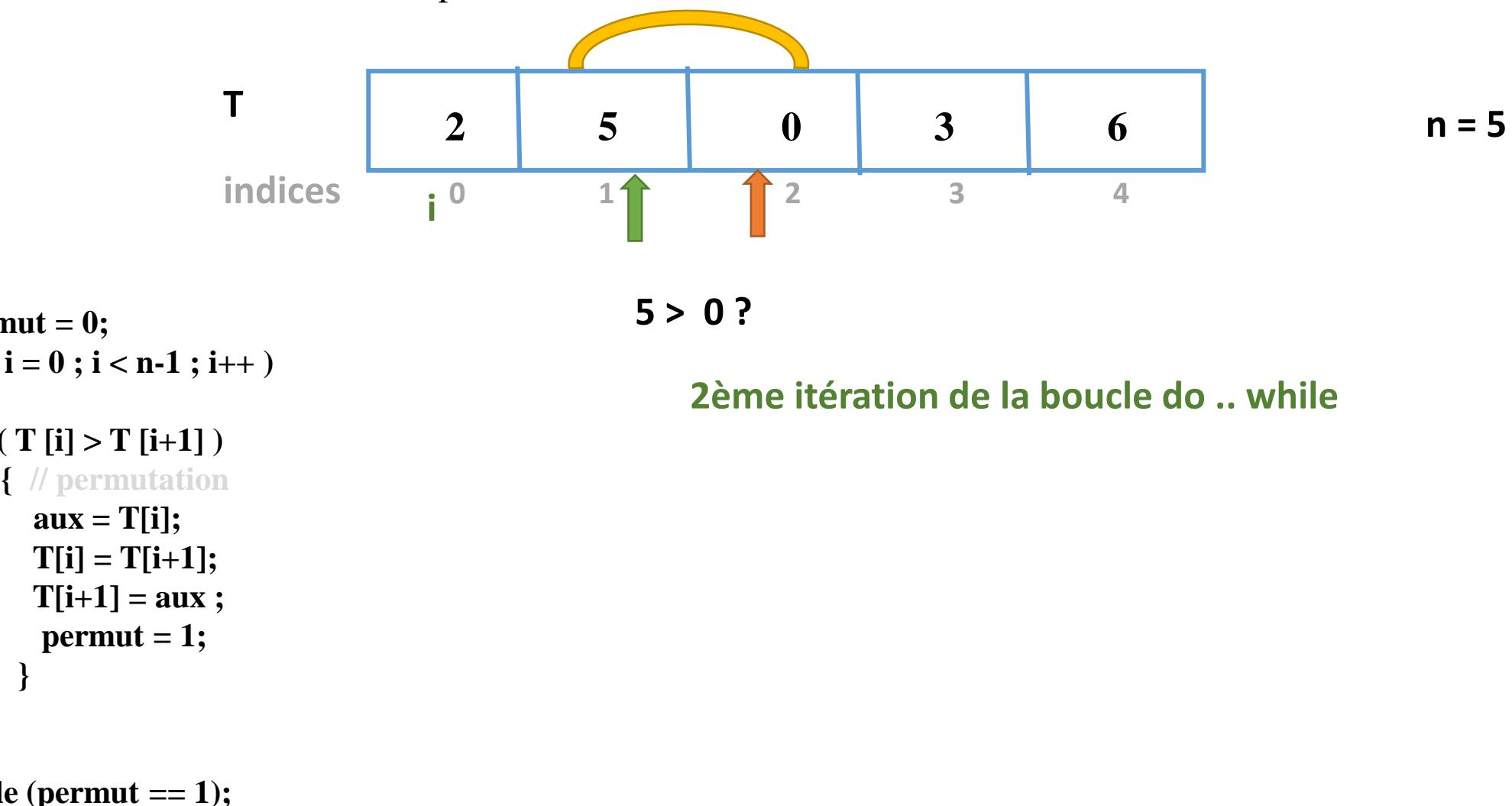
Exemple d'exécution

Soit le tableau T suivant composé de 5 entiers :



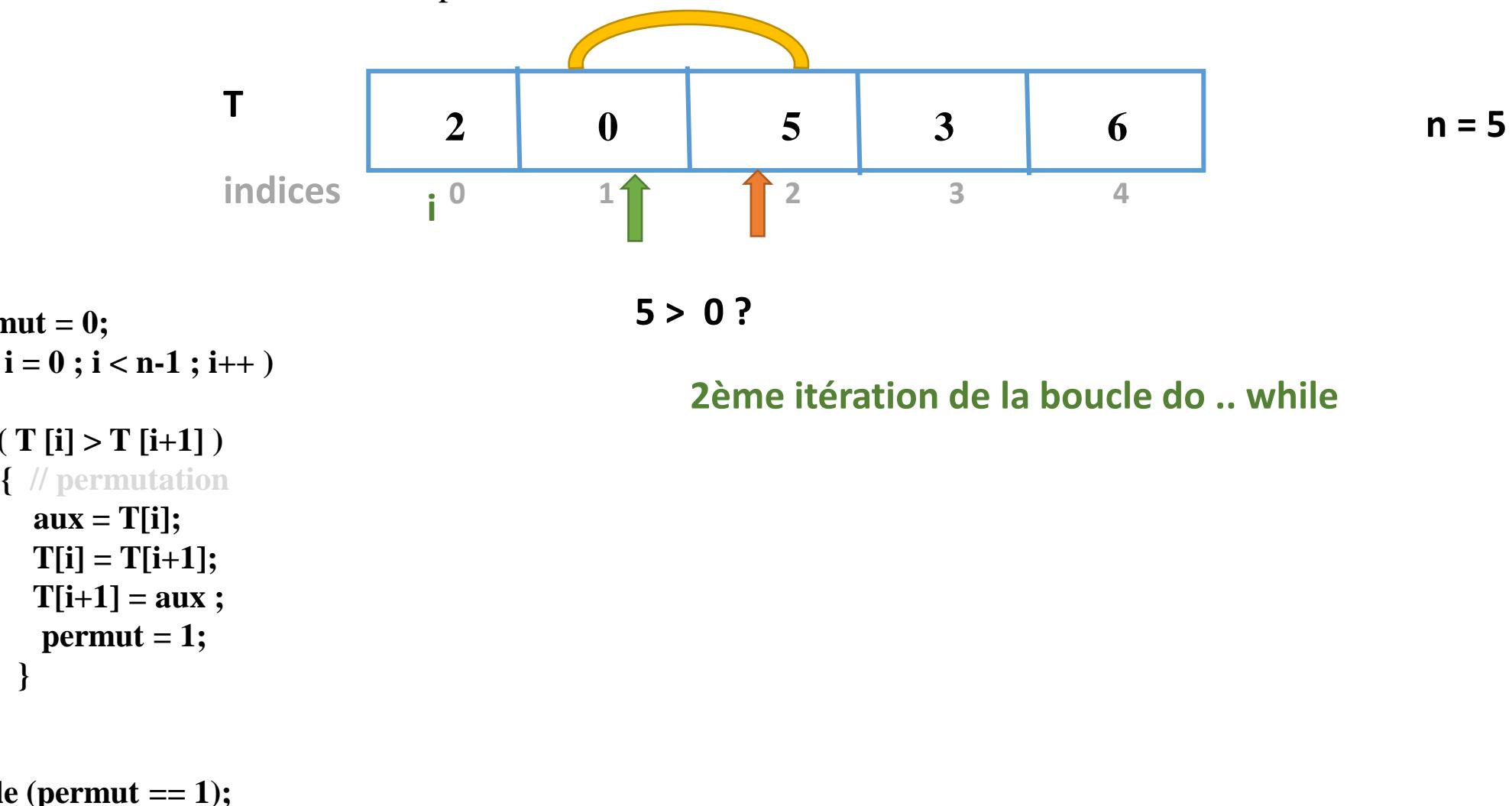
Exemple d'exécution

Soit le tableau T suivant composé de 5 entiers



Exemple d'exécution

Soit le tableau T suivant composé de 5 entiers



Exemple d'exécution

Soit le tableau T suivant composé de 5 entiers :

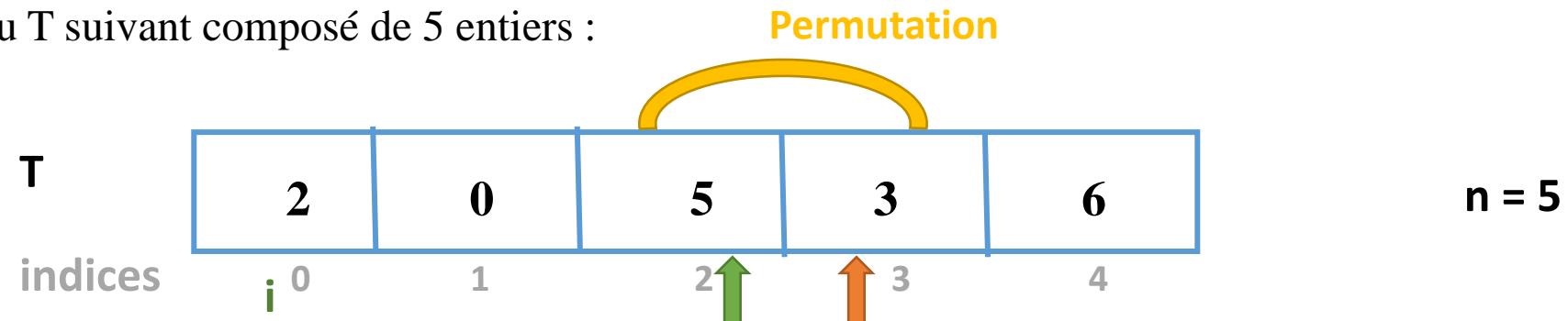
```
T          n = 5
indices    i 0      1      2      3      4
           2      0      5      3      6
do
{
permut = 0;
for( i = 0 ; i < n-1 ; i++ )
{
  if ( T [i] > T [i+1] )
    { // permutation
      aux = T[i];
      T[i] = T[i+1];
      T[i+1] = aux ;
      permut = 1;
    }
}
while (permut == 1);
```

5 > 3 ?

2ème itération de la boucle do .. while

Exemple d'exécution

Soit le tableau T suivant composé de 5 entiers :

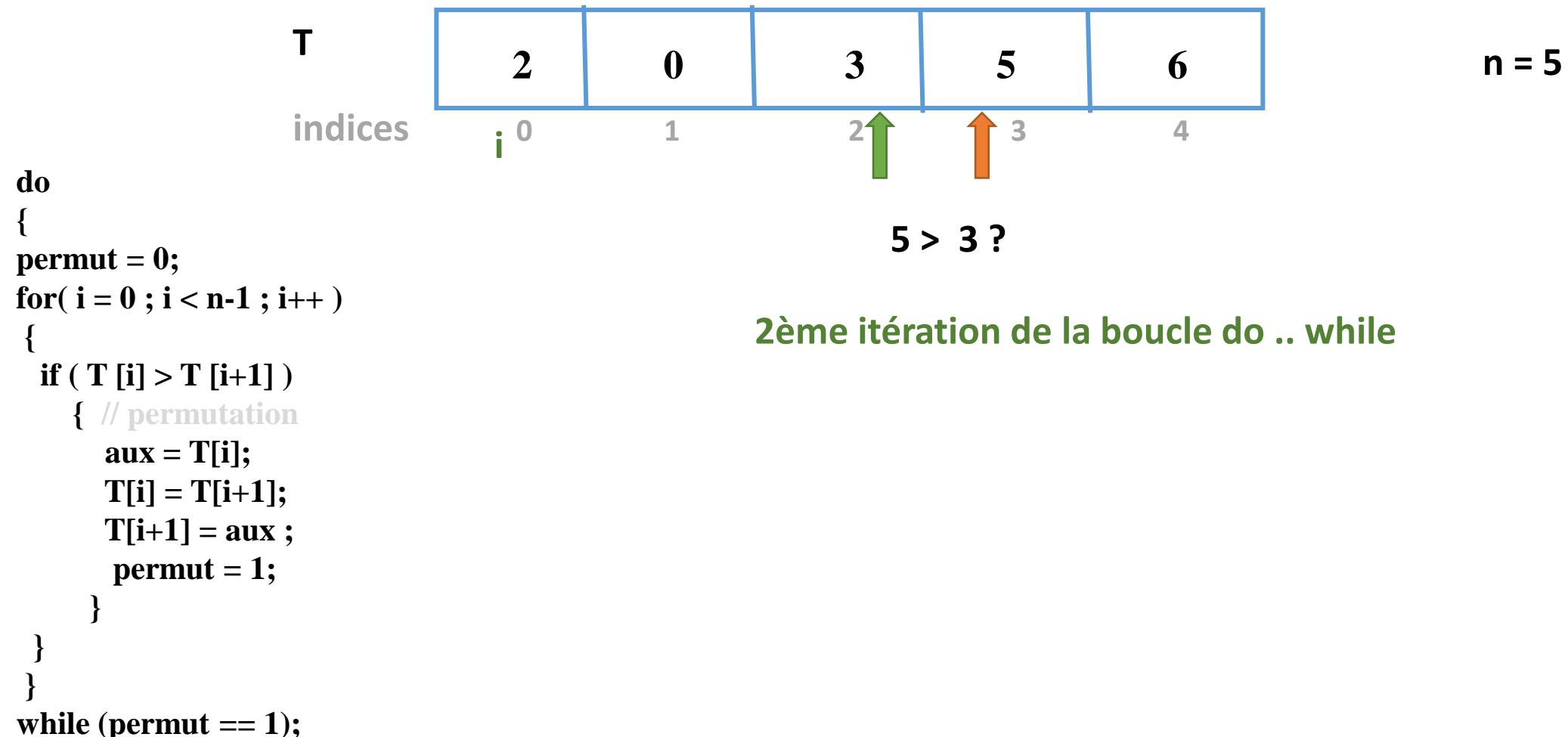


```
do
{
    permut = 0;
    for( i = 0 ; i < n-1 ; i++ )
    {
        if ( T [i] > T [i+1] )
            { // permutation
                aux = T[i];
                T[i] = T[i+1];
                T[i+1] = aux ;
                permut = 1;
            }
    }
    while (permut == 1);
```

2ème itération de la boucle do .. while

Exemple d'exécution

Soit le tableau T suivant composé de 5 entiers :



Exemple d'exécution

Soit le tableau T suivant composé de 5 entiers :

```
T          n = 5
indices      0   1   2   3   4
i           2   0   3   5   6
do
{
permut = 0;
for( i = 0 ; i < n-1 ; i++ )
{
    if ( T [i] > T [i+1] )
        { // permutation
            aux = T[i];
            T[i] = T[i+1];
            T[i+1] = aux ;
            permut = 1;
        }
}
while (permut == 1);
```

5 > 6 ?

2ème itération de la boucle do .. while

Exemple d'exécution

Soit le tableau T suivant composé de 5 entiers :

```
T          n = 5
indices
do
{
permut = 0;
for( i = 0 ; i < n-1 ; i++ )
{
    if ( T [i] > T [i+1] )
        { // permutation
            aux = T[i];
            T[i] = T[i+1];
            T[i+1] = aux ;
            permut = 1;
        }
}
while (permut == 1);
```

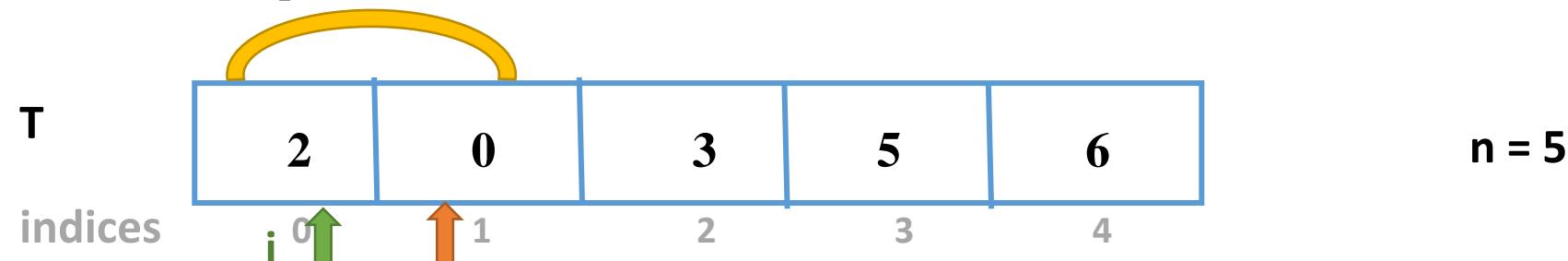
2 > 0 ?

3ème itération de la boucle do .. while

Exemple d'exécution

Permutation

Soit le tableau T suivant composé de 5 entiers :



```
do
{
    permut = 0;
    for( i = 0 ; i < n-1 ; i++ )
    {
        if ( T [i] > T [i+1] )
            { // permutation
                aux = T[i];
                T[i] = T[i+1];
                T[i+1] = aux ;
                permut = 1;
            }
    }
    while (permut == 1);
```

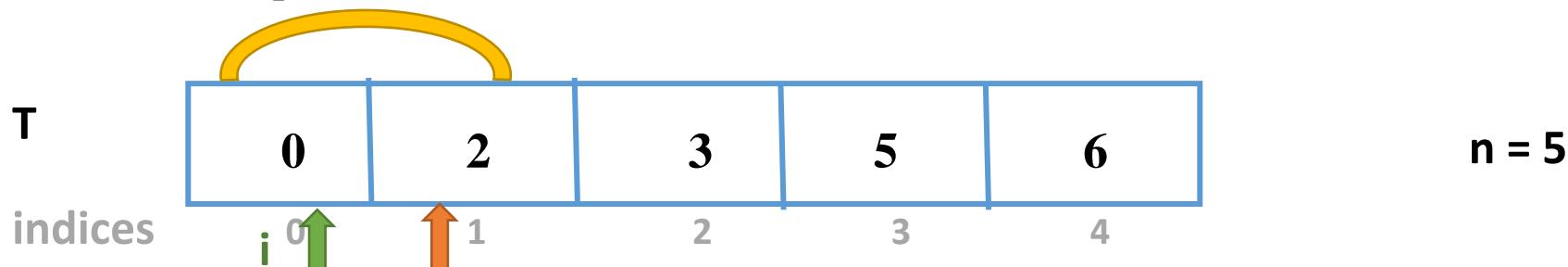
2 > 0 ?

3ème itération de la boucle do .. while

Exemple d'exécution

Permutation

Soit le tableau T suivant composé de 5 entiers :



```
do
{
    permut = 0;
    for( i = 0 ; i < n-1 ; i++ )
    {
        if ( T [i] > T [i+1] )
        { // permutation
            aux = T[i];
            T[i] = T[i+1];
            T[i+1] = aux ;
            permut = 1;
        }
    }
    while (permut == 1);
```

2 > 0 ?

3ème itération de la boucle do .. while

Exemple d'exécution

Soit le tableau T suivant composé de 5 entiers :

```
T          n = 5
indices    i 0           1   2           3   4
          0   2   3   5   6

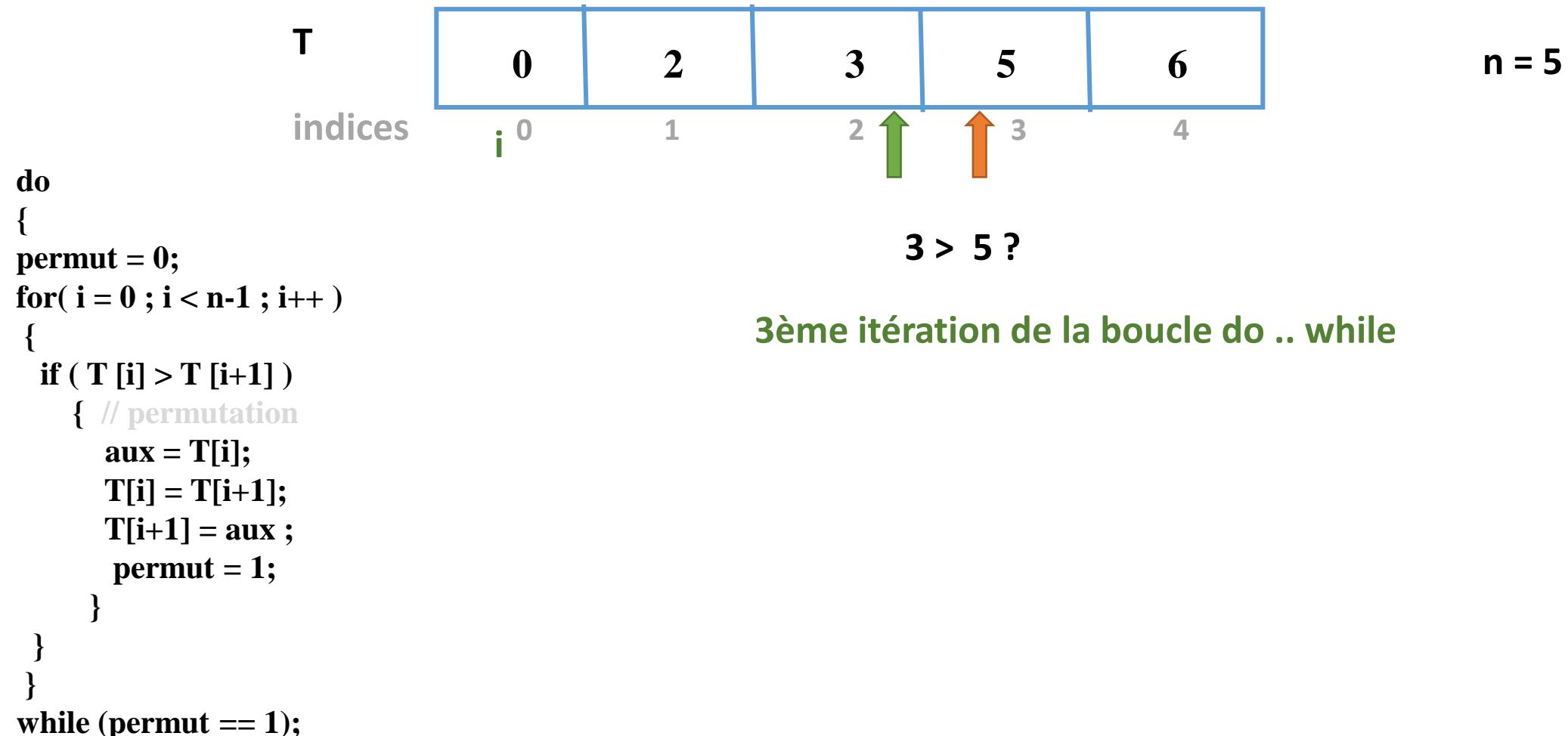
do
{
permut = 0;
for( i = 0 ; i < n-1 ; i++ )
{
  if ( T [i] > T [i+1] )
    { // permutation
      aux = T[i];
      T[i] = T[i+1];
      T[i+1] = aux ;
      permut = 1;
    }
}
while (permut == 1);
```

2 > 3 ?

3ème itération de la boucle do .. while

Exemple d'exécution

Soit le tableau T suivant composé de 5 entiers :



Exemple d'exécution

Soit le tableau T suivant composé de 5 entiers :

```
T          n = 5
indices
i 0      1      2      3      4
| | | | |
  0 2 3 5 6

do
{
permut = 0;
for( i = 0 ; i < n-1 ; i++ )
{
    if ( T [i] > T [i+1] )
        { // permutation
            aux = T[i];
            T[i] = T[i+1];
            T[i+1] = aux ;
            permut = 1;
        }
}
while (permut == 1);
```

5 > 6 ?

3ème itération de la boucle do .. while

Exemple d'exécution

Soit le tableau T suivant composé de 5 entiers :

```
T          n = 5
indices      0   2   3   5   6
              i  0   1   2   3   4
do
{
permut = 0;
for( i = 0 ; i < n-1 ; i++ )
{
    if ( T [i] > T [i+1] )
        { // permutation
            aux = T[i];
            T[i] = T[i+1];
            T[i+1] = aux ;
            permut = 1;
        }
}
while (permut == 1);
```

5 > 6 ?

4ème itération de la boucle do .. while

permut = 0

Exemple d'exécution

Soit le tableau T suivant composé de 5 entiers :

```
T          n = 5
indices
do
{
permut = 0;
for( i = 0 ; i < n-1 ; i++ )
{
    if ( T [i] > T [i+1] )
        { // permutation
            aux = T[i];
            T[i] = T[i+1];
            T[i+1] = aux ;
            permut = 1;
        }
}
while (permut == 1);
```

0 > 2 ?

4ème itération de la boucle do .. while

permut = 0

0	2	3	5	6
---	---	---	---	---

Exemple d'exécution

Soit le tableau T suivant composé de 5 entiers :

```
T          n = 5
indices      0   2   3   5   6
              i  0   1↑  2↑  2   3   4

do
{
permut = 0;
for( i = 0 ; i < n-1 ; i++ )
{
    if ( T [i] > T [i+1] )
        { // permutation
            aux = T[i];
            T[i] = T[i+1];
            T[i+1] = aux ;
            permut = 1;
        }
}
while (permut == 1);
```

2 > 3 ?

4ème itération de la boucle do .. while

permut = 0

Exemple d'exécution

Soit le tableau T suivant composé de 5 entiers :

```
T          n = 5
indices    i 0           1           2           3           4
          0   2   3   5   6
do
{
permut = 0;
for( i = 0 ; i < n-1 ; i++ )
{
  if ( T [i] > T [i+1] )
    { // permutation
      aux = T[i];
      T[i] = T[i+1];
      T[i+1] = aux ;
      permut = 1;
    }
}
while (permut == 1);
```

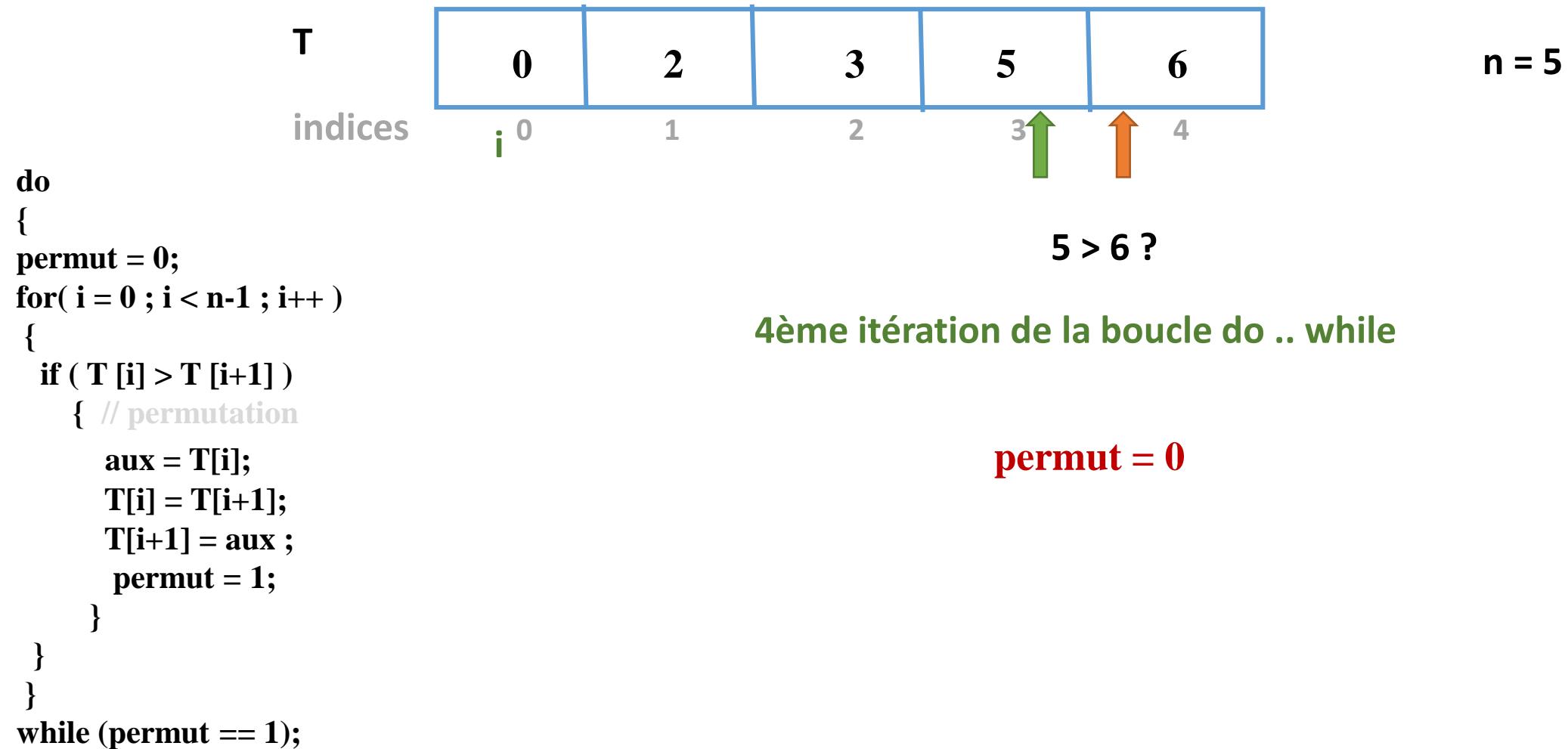
3 > 5 ?

4ème itération de la boucle do .. while

permut = 0

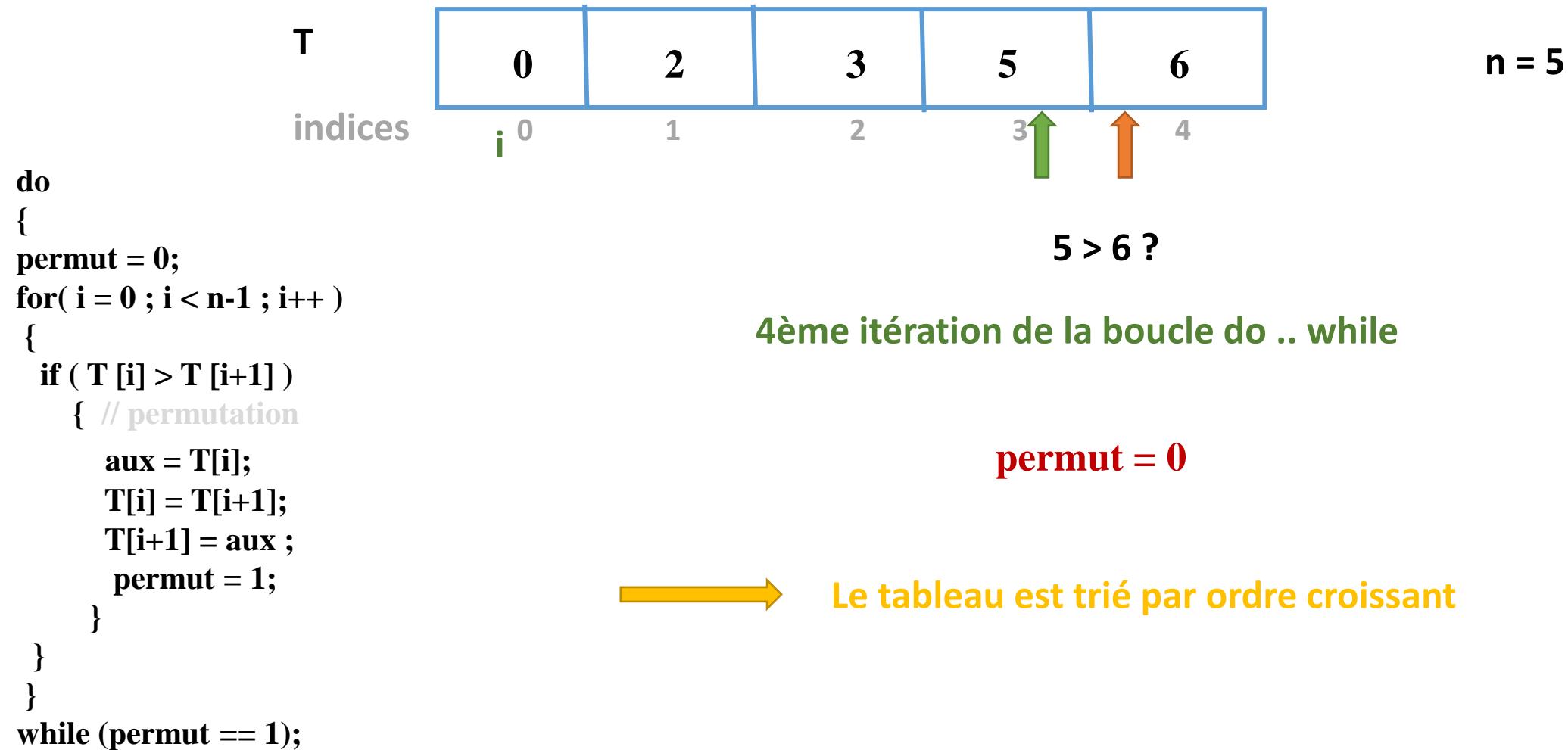
Exemple d'exécution

Soit le tableau T suivant composé de 5 entiers :



Exemple d'exécution

Soit le tableau T suivant composé de 5 entiers :



Optimisation de l'algorithme tri à bulles

```
taille = n ;                                // sauvegarder la taille initiale dans une variable nommée taille
do
{
    permut = 0;
    for(i=0; i<taille-1; i++)
    {
        if(T[i]>T[i+1])
        {
            aux=T[i];
            T[i]=T[i+1];
            T[i+1]=aux;
            permut = 1;
        }
    }
    taille--;
}
while(permut==1);
```

Autre version de l'algorithme tri à bulles

```
for (i=n-1; i>0; i--)
{
    for (j=1; j<=i; j++)
        if (T[j-1] > T[j])
    {
        tampon = T[j-1];
        T[j-1] = T[j];
        T[j] = tampon;
    }
}
```