

I. Définition d'un tri

- Pour trier un tableau, ses valeurs doivent pouvoir être ordonnées
- Un tri doit échanger des valeurs d'un tableau

procédure trier (E/S t : *Tableau*[1..MAX] d'*Element*, E $nbElements$: *Naturel*)

II. Tris courants

1. Tri à bulle

On parcourt le tableau tant qu'il n'est pas trié en inversant les deux premiers éléments consécutifs non ordonnée jusqu'à ce que le tableau soit trié.

2. Tri par sélection (ou par minimum successif)

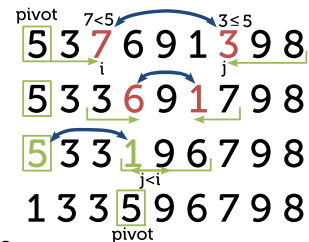
On parcourt le tableau. A chaque indice i , on parcourt le tableau de droite et on échange le plus petit élément qu'il contient avec la case suivant le tableau gauche.

3. Tri par insertion

On parcourt le tableau. A chaque indice i , on insère l'élément i où il faut dans le tableau gauche après avoir décalé la zone du tableau entre i et la nouvelle position de l'élément.

4. Tri rapide (récuratif, « intelligent » à la division)

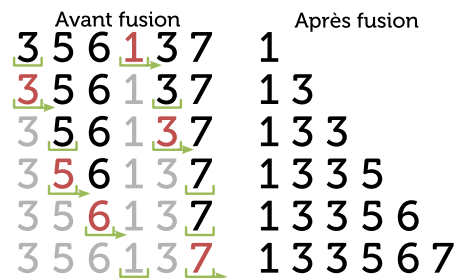
On partitionne le tableau pour que tous les éléments à gauche soient plus petits que tous les éléments à droite d'une valeur pivot. On trie récursivement de la même manière les tableaux de gauche et droite.



Pour partitionner, on prend la première valeur comme pivot, on incrémente i du début et on décrémente j de la fin jusqu'à une valeur respectivement plus grande et plus petite ou égale au pivot et on inverse les valeurs des indices i et j . On continue jusqu'à ce que j soit inférieur à i . Finalement, on inverse la valeur pivot et la valeur de la case j .

5. Tri fusion (récuratif, « intelligent » à la combinaison)

On coupe le tableau en deux, on trie les deux sous-tableaux, puis on les rassemble. Pour trier un sous-tableau, on répète l'opération. Pour fusionner deux tableaux triés, on prend à chaque fois la plus petite valeur entre la première valeur (pas encore prise) du premier tableau et celle du deuxième, jusqu'à avoir tout pris.



III. Pseudo-code des tris courants

1. Tri à bulle

procédure triABulles (E/S t : *Tbl*, E nb : *Naturel*)

Declaration i : *Naturel*
 $estTrie$: *Booleen*

début

répéter

$estTrie \leftarrow \text{vrai}$

pour $i \leftarrow 1$ **à** $nb - 1$ **faire**

si $t[i] > t[i + 1]$ **alors**

$\text{echanger}(t[i], t[i + 1])$

$estTrie \leftarrow \text{faux}$

fin

finpour

jusqu'à ce que $estTrie$

fin

2. Tri par sélection

```

fonction indiceDuMinimum (t : Tbl, borneInf, borneSup :
Naturel) : Naturel
  Déclaration i, resultat : Naturel
  début
    resultat ← borneInf
    pour i ← borneInf + 1 à borneSup faire
      si t[i] < t[resultat] alors
        resultat ← i
      finsi
    finpour
  retourner resultat
fin

procédure triParSelection (E/S t : Tbl, E nb : Naturel)
  Déclaration i : Naturel
  début
    pour i ← 1 à nb - 1 faire
      echanger(t[i], t[indiceDuMinimum(t,i,nb)])
    finpour
  fin

```

3. Tri par insertion

```

fonction obtenirIndiceDInsertion (t : Tbl, borneSup :
Naturel, lEntier : Entier) : Naturel
  Déclaration i : Naturel
  début
    i ← 1
    tant que t[i] ≤ lEntier et i < borneSup faire
      i ← i + 1
    fintantque
  retourner i
fin

procédure decaler (E/S t : Tbl, E borneInf, borneSup :
Naturel)
  Déclaration i : Naturel
  début
    pour i ← borneSup à borneInf - 1 pas de -1 faire
      t[i] ← t[i - 1]
    finpour
  fin

procédure triParInsertion (E/S t : Tbl, E nb : Naturel)
  Déclaration i, j : Naturel, temp : Entier
  début
    pour i ← 2 à nb faire
      j ← obtenirIndiceDInsertion(t, i, t[i])
      temp ← t[i]
      decaler(t, j, i)
      t[j] ← temp
    finpour
  fin

```

4. Tri rapide

```

procédure triRapide (E/S t : Tbl, E nb : Naturel)
  début
    triRapideRecuratif(t, 1, nb)
  fin

procédure triRapideRecuratif (E/S t : Tbl, E d, f : Naturel)
  Déclaration indicePivot : Naturel
  début
    si d < f alors
      partitionner(t, d, f, indicePivot)
      triRapideRecuratif(t, d, indicePivot - 1)
      triRapideRecuratif(t, indicePivot + 1, f)
    finsi
  fin

```

```

procédure partitionner (E/S t : Tbl, E debut, fin :
Naturel; S indicePivot : Naturel)
  Déclaration i, j, pivot : Naturel
  début
    pivot ← t[debut]
    i ← debut
    j ← fin
    tant que i ≤ j faire
      si t[i] ≤ pivot alors
        i ← i + 1
      sinon
        si t[j] > pivot alors
          j ← j - 1
        sinon
          echanger(t[i], t[j])
        finsi
      finsi
    fintantque
    indicePivot ← j
    echanger(t[debut], t[j])
  fin

```

5. Tri fusion

```

procédure triFusion (E/S t : Tbl, E nb : Naturel)
  début
    triFusionRecuratif(t, 1, nb)
  fin

procédure triFusionRecuratif (E/S t : Tbl, E d, f : Naturel)
  début
    si d < f alors
      triFusionRecuratif(t, d, (d + f) div 2)
      triFusionRecuratif(t, ((d + f) div 2) + 1, f)
      fusionner(t, d, (d + f) div 2, f)
    finsi
  fin

procédure fusionner (E/S t : Tbl; E debut, milieu, fin :
Naturel)
  Déclaration i, j, k : Naturel, temp : Tbl
  début
    i ← debut
    j ← milieu + 1
    pour k ← 1 à fin - debut + 1 faire
      si i ≤ milieu et j ≤ fin alors
        si t[i] ≤ t[j] alors
          temp[k] ← t[i]
          i ← i + 1
        sinon
          temp[k] ← t[j]
          j ← j + 1
        finsi
      sinon
        si i ≤ milieu alors
          temp[k] ← t[i]
          i ← i + 1
        sinon
          temp[k] ← t[j]
          j ← j + 1
        finsi
      finsi
    finpour
    pour k ← 1 à fin - debut + 1 faire
      t[debut + k - 1] ← temp[k]
    finpour
  fin

```