Université de Nantes
M2 Bioinformatique

Support pédagogique

Tri sur Vecteur

Christine Sinoquet

SOMMAIRE

Tri quick sort
Version en C

Fichier quick_sort.h :

```
 typedef int TvectInt[];
 void trier(TVectInt v, int g, int d);
 void separer(TVectInt v, int g, int d, int* adr_indice_pivot);
```

```
/*************************************

void separer(TVectInt v, int g, int d, int* adr_indice_pivot)
// PRECONDITION :
// g < d
//POSTCONDITION :
// v[g.. indice_pivot -1] ≤ v[indice_pivot] < v[indice_pivot +1..d]

 {
   int bas, haut; //indices de position dans le vecteur v
   int comp, pivot;  //comp pour comparateur
   bas    = g;
   haut  = d;
   pivot = v[bas];
   comp = v[haut];

   while (bas < haut)
   {
      if ( comp > pivot)
      {
        v[haut] = comp;
        haut --;
        comp = v[haut];
      }else{
        v[bas] = comp;
        bas ++;
        comp = v[bas];
      }
   }
   v[bas] = pivot;
   *adr_indice_pivot = bas; // ou indifferemment *adr_indice_pivot = haut
 } // fin separer

/*************************************
void trier(TVectInt v, int g, int d)
 {
   int indice_pivot;

   if (g < d)
   {
     separer(v, g, d, &indice_pivot);
     trier(v, g, indice_pivot-1);
     trier(v, indice_pivot+1, d);
   }
```

```
 } // fin trier
/*************************************
```
Fichier main.c :

```
#include <stdio.h>
#include <stdlib.h>
#include "quick_sort.h"

/*************************************

void afficher_vect(TVectInt v, int taille)
{
 for(i=0; i < taille; i++)
 {
   printf("%d ", v[i]);
  }
  printf("\n");
} // fin  afficher_vect

/*************************************

void main()
 {
   int i;
   TVecInt v                  = {2,6,4,1,9};
    int       borne_gauche = 0;
    int       borne_droite   = 4;

   afficher_vect(v, 5) ;
   trier(v, borne_gauche, borne_droite);
   afficher_vect(v, 5) ;
 } // fin main

/*************************************
/*************************************
/*************************************
```

Tri par baquets
Version en pseudo-code

L'algorithme suivant réalise un tri pour des nombres écrits en écriture décimale avec au plus k chiffres (ils sont donc inférieurs à 10 puissance k).

On utilise un tableau de « baquets » de taille 10, indicé par 0, 1, …, 9. Le tableau s'appelle baquets. On dispose donc de baquets[0], …, baquets[9]. Chacun des baquets est une structure contenant deux pointeurs destinés à contenir les adresses de début et de fin d'une liste chaînée d'entiers. Chaque baquets[i] « contient » donc une liste chaînée d'entiers.

On dispose des types suivants:

```
typedef struct {
 int info ;
 struct TCell* next ;
} TCell ;
typedef TCell* TPointer ;

typedef struct {
  Tpointer debut ;
  Tpointer fin ;
} TDebFin ;
```

```
PROCEDURE tri_par_baquets(es liste_a_trier : TPointer, e k : entier)
debut

 variables locales :
 baquets : tableau de 10 structures de type TDebFin
 i, elem, num : entier

  pour i allant de 1 à k
    // initialiser à NULL les deux pointeurs de chaque baquet dans le tableau baquets
    pour a allant de 0 à 9
     baquets[a].debut ← NULL ;  baquets[a].fin ← NULL
    fin pour

    pour chaque entier elem de liste_a_trier
        num ←  le chiffre numéro i en partant de la droite de l'entier elem
        ajouter_queue (baquets[num], elem)
    fin pour

    // mettre à jour liste_a_trier en concatenant entre eux baquets[0], baquets[1], …,  baquets[9]
    liste_a_trier  NULL
    pour a allant de 0 à 9
      liste_a_trier ← concatener (liste_a_trier, baquets[a])
    fin pour
  fin pour

fin
```

● exemple :
i = 2, elem = 473
num ← le chiffre numéro i en partant de la droite de l'entier elem
num = 7

● exemple d'exécution :

liste_a_trier = 67, 56, 3, 106, 97, 12, 3
k = 3

itération 1 : i = 1
liste_a_trier = **67**, 56, 3, 106, 97, 12, 3
baquets[0]
baquets[1]
baquets[2]
baquets[3]
baquets[4]
baquets[5]
baquets[6]
baquets[7] **67**
baquets[8]
baquets[9]

liste_a_trier = 67, **56**, 3, 106, 97, 12, 3
baquets[0]
baquets[1]
baquets[2]
baquets[3]
baquets[4]
baquets[5]
baquets[6] **56**
baquets[7] 67
baquets[8]
baquets[9]

liste_a_trier = 67, 56, **3**, 106, 97, 12, 3
baquets[0]
baquets[1]
baquets[2]
baquets[3] **3**
baquets[4]
baquets[5]
baquets[6] 56
baquets[7] 67
baquets[8]
baquets[9]

liste_a_trier = 67, 56, 3, **106**, 97, 12, 3
baquets[0]
baquets[1]
baquets[2]
baquets[3] 3
baquets[4]
baquets[5]
baquets[6] 56, **106**
baquets[7] 67
baquets[8]
baquets[9]

liste_a_trier = 67, 56, 3, 106, **97**, 12, 3
baquets[0]
baquets[1]
baquets[2]
baquets[3] 3
baquets[4]
baquets[5]
baquets[6] 56, 106
baquets[7] 67, **97**
baquets[8]
baquets[9]

liste_a_trier = 67, 56, 3, 106, 97, **12**, 3
baquets[0]
baquets[1]
baquets[2] **12**
baquets[3] 3
baquets[4]
baquets[5]
baquets[6] 56, 106
baquets[7] 67, 97
baquets[8]
baquets[9]

liste_a_trier = 67, 56, 3, 106, 97, 12, **3**
baquets[0]
baquets[1]
baquets[2] 12
baquets[3] 3, **3**
baquets[4]
baquets[5]
baquets[6] 56, 106
baquets[7] 67, 97
baquets[8]
baquets[9]

mise à jour : liste_a_trier = 12, 3, 3, 56, 106, 67, 97

itération 2 : i = 2
liste_a_trier = **12**, 3, 3, 56, 106, 67, 97
baquets[0]
baquets[1] **12**
baquets[2]
baquets[3]
baquets[4]
baquets[5]
baquets[6]
baquets[7]
baquets[8]
baquets[9]

liste_a_trier = 12, **3**, 3, 56, 106, 67, 97
baquets[0] **3**
baquets[1] 12
baquets[2]
baquets[3]
baquets[4]
baquets[5]
baquets[6]
baquets[7]
baquets[8]
baquets[9]

liste_a_trier = 12, 3, **3**, 56, 106, 67, 97
baquets[0] 3, **3**
baquets[1] 12
baquets[2]
baquets[3]
baquets[4]
baquets[5]
baquets[6]
baquets[7]
baquets[8]
baquets[9]

liste_a_trier = 12, 3, 3, **56**, 106, 67, 97
baquets[0] 3, 3
baquets[1] 12
baquets[2]
baquets[3]
baquets[4]
baquets[5] **56**
baquets[6]
baquets[7]
baquets[8]
baquets[9]

liste_a_trier = 12, 3, 3, 56, **106**, 67, 97
baquets[0] 3, 3, **106**
baquets[1] 12
baquets[2]
baquets[3]
baquets[4]
baquets[5] 56
baquets[6]
baquets[7]
baquets[8]
baquets[9]

liste_a_trier = 12, 3, 3, 56, 106, **67**, 97
baquets[0] 3, 3, 106
baquets[1] 12
baquets[2]
baquets[3]
baquets[4]
baquets[5] 56
baquets[6] **67**
baquets[7]
baquets[8]
baquets[9]

liste_a_trier = 12, 3, 3, 56, 106, 67, **97**
baquets[0] 3, 3, 106
baquets[1] 12
baquets[2]
baquets[3]
baquets[4]
baquets[5] 56
baquets[6] 67
baquets[7]
baquets[8]
baquets[9] **97**

mise à jour : liste_a_trier = 3, 3, 106, 12, 56, 67, 97

itération 3 : i = 3
liste_a_trier = **3**, 3, 106, 12, 56, 67, 97
baquets[0] **3**
baquets[1]
baquets[2]
baquets[3]
baquets[4]
baquets[5]
baquets[6]
baquets[7]
baquets[8]
baquets[9]

liste_a_trier = 3, **3**, 106, 12, 56, 67, 97
baquets[0] 3, **3**
baquets[1]
baquets[2]
baquets[3]
baquets[4]
baquets[5]
baquets[6]
baquets[7]
baquets[8]
baquets[9]

liste_a_trier = 3, 3, **106**, 12, 56, 67, 97
baquets[0] 3, 3
baquets[1] **106**
baquets[2]
baquets[3]
baquets[4]
baquets[5]
baquets[6]
baquets[7]
baquets[8]
baquets[9]

liste_a_trier = 3, 3, 106, **12**, 56, 67, 97
baquets[0] 3, 3, **12**
baquets[1] 106
baquets[2]
baquets[3]
baquets[4]
baquets[5]
baquets[6]
baquets[7]
baquets[8]
baquets[9]

liste_a_trier = 3, 3, 106, 12, **56**, 67, 97
baquets[0] 3, 3, 12, **56**
baquets[1] 106
baquets[2]
baquets[3]
baquets[4]
baquets[5]
baquets[6]
baquets[7]
baquets[8]
baquets[9]

liste_a_trier = 3, 3, 106, 12, 56, **67**, 97
baquets[0] 3, 3, 12, 56, **67**
baquets[1] 106
baquets[2]
baquets[3]
baquets[4]
baquets[5]
baquets[6]
baquets[7]
baquets[8]
baquets[9]

liste_a_trier = 3, 3, 106, 12, 56, 67, **97**
baquets[0] 3, 3, 12, 56, 67, **97**
baquets[1] 106
baquets[2]
baquets[3]
baquets[4]
baquets[5]
baquets[6]
baquets[7]
baquets[8]
baquets[9]

mise à jour liste_a_trier = 3, 3, 12, 56, 67, 97, 106