Feuille de TP n°1

Exercice 1. Programmer une recherche dichotomique dans un tableau trié, qui renvoie `-1` si l'élément cherché ne se trouve pas dans le tableau, et sa position (ou une de ses positions) sinon.

Exercice 2. Programmer le tri par tas décrit dans la feuille de TD n°1. On pourra utiliser le fichier squelette suivant, disponible à l'adresse `http://perso.univ-rennes1.fr/sandrine.caruso/ALBA/tas-squelette.c`. En plus des fonctions ci-dessous, n'hésitez pas à programmer toutes les fonctions intermédiaires que vous jugerez utiles.

1. Programmer une fonction `remonter` qui, étant donné un tableau de longueur n (que l'on imagine représenté sous forme d'arbre binaire), « remonte » le dernier élément en l'échangeant au besoin avec chacun de ses ancêtres, jusqu'à ce que son père soit plus grand que lui.
2. Programmer une fonction `construire_tas` qui transforme un tableau en tas. On pourra utiliser la fonction `remonter`.
3. Programmer une fonction `descendre` qui descend le premier élément du tableau donné en argument en l'échangeant au besoin avec le plus grand de ses fils, et ainsi de suite, jusqu'à ce que ses deux fils soient plus petits que lui.
4. Programmer la fonction `tri_par_tas` qui implémente le tri par tas. On pourra utiliser les fonctions `construire_tas` et `descendre`.

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

void remonter (int tab[], int n)
{
}

void construire_tas (int tab[], int n)
{
}

void descendre (int tab[], int n)
{
}

void tri_par_tas (int tab[], int n)
{
}

void printtab (int tab[], int n) /* fonction pour afficher
un tableau à l'écran */
```

```
{
  int i;
  for (i=0;i<n;i++) {
   printf("%d ",tab[i]);
  }
  printf("\n");
}

int bien_trie (int tab[], int n) /* renvoie 1 si le tableau est bien trié,
0 sinon */
{
  int i;
  for (i=1; i<n; i++) {
    if (tab[i] < tab[i-1]) {
      return 0;
    }
  }
  return 1;
}

int main() /* la fonction main teste la fonction tri_par_tas sur
un tableau de longueur 100 contenant des nombres tirés au hasard
entre 0 et 99 */
{
  srand(time(NULL));
  #define longueur_tableau 100

  int tab[longueur_tableau];

  int i;
  for (i=0;i<longueur_tableau;i++) {
   tab[i] = rand() % 100;
  } // initialisation du tableau

  printtab(tab,longueur_tableau); // affichage du tableau initial
  printf("----\n");

  tri_par_tas(tab,longueur_tableau); // tri du tableau

  printtab(tab,longueur_tableau); // affichage du tableau trié
  if (bien_trie(tab,longueur_tableau)) {
    printf("Le tableau est bien trie\n");
  }
  else {
    printf("Attention, le tableau n'est pas bien trie !\n");
  }
  return 0;
}
```