TP 2: Tri par tas (Heap Sort)

Objectif

Pour améliorer les algorithmes précédents de TP1 on propose de mettre en œuvre une structure de données facilitant la recherche du minimum d'un ensemble : un tas.

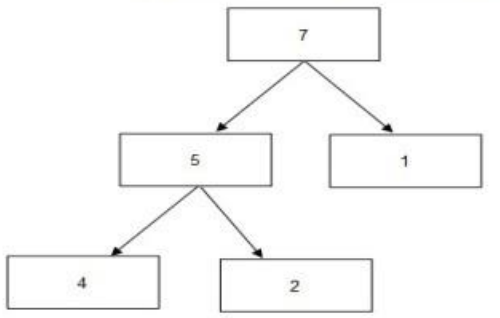
Définition de tas

Le tas (heap en anglais) peut être représenté comme un arbre binaire presque complet.
Pour rappel un arbre binaire est dit complet si toutes ses feuilles sont de même profondeur.
Deux sortes de tas binaires : les tas min et les tas max.
Tas-min : chaque élément est supérieur à son parent.
Tas-max : chaque élément est inférieur à son parent.
Le tas peut être stocké de façon très simple dans un tableau. Soit i l'indice d'un noeud. Les indices de son parent, enfant de gauche, et enfant de droite sont calculables de manière très simple.

- let parent i = (i - 1) / 2;
- let enfant_gauche i = 2 * i + 1 ;
- let enfant_droite i = 2 * i + 2;

Exemple :

Tas binaire réprésente sous forme d'arbre



Même tas sous forme de tableau

indice	0	1	2	3	4
valeur	7	5	1	4	2

Travail à faire :

1. Développer un programme qui détermine le tri_par_tas d'une liste d'entiers par ordre croissant.
2. Quelle est la complexité de cet algorithme de tri ?
3. Donner une comparaison de la complexité temporelle (rapidité) de l'algorithme de tri_par_tas avec les algorithmes précédents de TP1 (tri par sélection, tri par insertion, tri rapide et tri par fusion).

Bon courage