TP n°19 – Tas dans un tableau et tri par tas

Le but de ce TP est d'implémenter toutes les primitives d'un tas **max** stocké sous forme tabulaire, et d'en déduire l'algorithme de tri par tas.

1 Représentation d'un tas

On représente un tas dans un tableau par le type Ocaml suivant.

```
type 'a tasbleau = {
  donnees : 'a array;
  mutable taille : int
}
```

Comme d'habitude quand on représente par un tableau une structure de données qui grandit, on prend un tableau de grande taille n fixée (par exemple 40) et on stocke le nombre de cases du tableau effectivement utilisées dans `taille`.

Cet argument `taille` nous servira pour implémenter le tri par tas, puisqu'on pourra ignorer une partie des données.

- **Q1.** Implémenter des fonctions `gauche:int->int`, `droite:int->int`, `parent:int->int` qui donnent l'**indice** du fils gauche, du fils droit et du père du noeud d'indice i.
- **Q2.** Implémenter une fonction pour créer un tas vide. Elle pourra prendre en entrée la taille n désirée pour le tableau de stockage et un élément initialisateur (pour le type du tas).

On donne la fonction suivante qui permet d'échanger deux éléments dans le tas :

```
(* Échange les éléments d'indice i et j dans le tas *)
let tas_echanger tas i j =
  assert (0 <= i && i < tas.taille);
  assert (0 <= j && j < tas.taille);
  let tmp = tas.donnees.(i) in
  tas.donnees.(i) <- tas.donnees.(j);
  tas.donnees.(j) <- tmp;;
```

On rappelle ici qu'on veut faire un tas **max**.

- **Q3.** Implémenter une fonction `percole_haut : 'a tasbleau -> int -> unit` qui effectue la percolation vers le haut de la valeur x située initialement en position i. Par "percolation vers le haut" on entend bien l'opération en plusieurs étapes qui met une valeur à sa place, pas juste une étape d'échange.
- **Q4.** Implémenter une fonction `percole_bas : 'a tasbleau -> int -> unit` qui effectue la percolation vers le bas de la valeur x située initialement en position i. Même indication que précédemment.
- **Q5.** Implémenter l'ajout d'une valeur dans le tas. La signature sera `ajoute : 'a tasbleau -> 'a -> unit`.
- **Q6.** Implémenter le retrait de la racine. La signature sera `retire : 'a tasbleau -> 'a`. On rappelle que notre but est d'implémenter le tri par tas, donc il peut être intéressant de mettre la valeur retirée à un certain endroit du tableau.

2 Tri par tas

- **Q7.** Écrire une fonction `cree_tas:'a array -> 'a tasbleau` qui prend en entrée un tableau de valeurs et renvoie un tasbleau qui contient ces valeurs. C'est la méthode la plus efficace (linéaire) qui est attendue.
- **Q8.** Implémenter le tri par tas. La signature sera `tri_par_tas: 'a array -> unit`. Remarque : utiliser le n correspondant à la taille du tableau à trier.

3 Exemple d'application

Un secrétaire surchargé reçoit tous les jours des mails pour les contrats de son entreprise. Chaque contrat rapport plus ou moins à l'entreprise et le secrétaire voudrait traiter les mails dans l'ordre de ce qu'ils rapportent à l'entreprise.

On suppose que les contrats sont donnés dans une liste de couples. Chaque couple contient le nom du client et ce que le contrat rapporte.

Par exemple la liste peut ressembler à ça :

`[("supercar",10000);("minipme",500);("assurisque",30000);("arnakmax",2000)]`.

Chaque jour de nouveau mails arrivent, on modélise ça par la fonction `nouveau_mails:unit->(char*int) list` (qui les génère aléatoirement), disponible dans le fichier du TP.

On suppose que chaque jour le secrétaire peut gérer 5 contrats. On veut simuler le travail du secrétaire pendant 100 jours et compter le gain total de l'entreprise.

Évidemment le but est d'utiliser une file de priorité dans l'implémentation.

Écrire une fonction qui simule les 100 jours de travail, sachant que des nouveaux mails arrivent tous les jours et qu'avant le premier jour la boite mail du secrétaire était vide.

4 Si vous avez fini

Vous pouvez réimplémenter les primitives du tas quand il est représenté par un arbre (pas un tableau), ou écrire une fonction qui teste si un arbre est bien un tas.

Vous pouvez également écrire la fonction qui retire une étiquette **quelconque** du tasbleau (donnée en entrée), et pas juste la racine.