- TP 3 -
Boutique en ligne

On souhaite écrire un programme permettant de gérer les articles proposés à la vente dans une boutique. L'application doit permettre d'enregistrer des articles, de les enlever et de faire des recherches. On suppose qu'il n'existe pas deux articles avec le même prix.

On peut encore une fois utiliser l'unité qui implémente les primitives des arbres binaires et qui a été développée précédemment mais il faudra la compléter avec les opérations nécessaires pour les nouvelles fonctionnalités. Plus précisément, il faudra que les arbres construits soient des arbres binaires ordonnés permettant une recherche potentiellement plus efficace que pour les arbre binaires classiques. Un exemple d'exécution utilisant le programme `magasin-test.c` fourni sur la page du cours est disponible à la fin de l'énoncé.

1.Ajouter un article
Proposer un type permettant de décrire les articles mis en vente - ces articles sont caractérisés par un nom et un prix (voir header file `base.h`). Les articles sont stockés dans un arbre ordonné par rapport au prix de vente. Implémenter la procédure permettant d'insérer un article dans l'arbre de telle manière que l'arbre de recherche reste ordonné.

2.Chercher l'article le moins/plus cher
Implanter les fonctions permettant de trouver efficacement l'article le moins cher, respectivement le plus cher, proposé dans la boutique. Si aucun article n'est disponible dans la boutique alors un article avec un prix négatif doit être retourné dans les deux cas.

3.Chercher un article avec un prix donné
Implanter la fonction permettant de chercher efficacement un article avec un prix donné. Si un tel article n'existe pas dans la boutique alors un article avec un prix négatif doit être retourné.

4.Supprimer un article
Implanter la procédure permettant de supprimer un article. L'arbre binaire correspondant doit bien évidement rester ordonné à la fin de l'opération.

5.Améliorer la recherche
La boutique propose maintenant un nombre important d'articles et les recherches se révèlent inefficaces. Il faudra compléter l'implantation actuelle pour permettre l'utilisation des arbres binaires de recherche équilibrés - il faudra donc ajouter une procédure permettant d'insérer un article dans l'arbre de telle manière que l'arbre de recherche reste ordonné et équilibré.

6.Trouver un article abordable (si vous avez répondu aux questions précédentes)
Implanter une fonction permettant de trouver l'article le plus cher qui peut être acheté avec un montant donné. Si aucun article n'est disponible dans la boutique ou tous les articles coûtent plus cher que le montant dont on dispose alors un article avec un prix négatif doit être retourné dans les deux cas.

Exemple exécution

L'exécution du programme `magasin.c` produit l'affichage suivant:

```
---------- INSERT ----------

      @----MacMini (600 euros)
      |  |
      |  @----iPad (850 euros)
      |
   @----iPhone (900 euros)
   |  |
   |  @----iMac (1100 euros)
   |
@----MacBook (1300 euros)
   |
   @----MacPro (3000 euros)

--------------------------------------

---------- MIN/MAX ----------
MIN = MacMini (600 euros)
MAX = MacPro (3000 euros)
--------------------------------------

---------- SEARCH ----------
iPad (850 euros)
 (-1 euros)
--------------------------------------

---------- REMOVE ----------


--------------------------------------

---------- INSERT AVL ----------

   @----MacMini (600 euros)
   |  |
   |  @----iPad (850 euros)
   |
@----iPhone (900 euros)
   |
   |  @----iMac (1100 euros)
   |  |
   @----MacBook (1300 euros)
      |
      @----MacPro (3000 euros)

--------------------------------------

---------- AFFORDABLE ----------
iPhone (900 euros)
--------------------------------------
```