

INF251 Algo 1
 Devoir surveillé n° 3
 30mn
 Aucun document autorisé
 27/11/2008

Exercice 1 :

On considère un arbre binaire de recherche (ABR) contenant les entiers 1 à 500. Parmi les suites d'entiers suivantes, indiquez celles qui peuvent correspondre à une suite d'entiers rencontrés dans l'ABR lorsqu'on recherche la valeur 125. Justifier votre réponse.

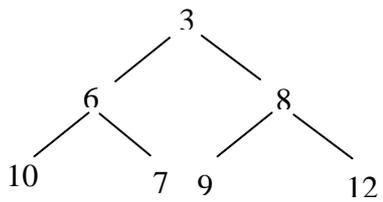
- a) 420 235 80 58 208 180 125
- b) 352 135 88 113 129 123 125
- c) 90 112 427 350 213 108 125

Exercice 2 :

- 1) Indiquer quelles sont les conditions nécessaires pour vérifier qu'un arbre binaire est un ABR.
- 2) Dessiner l'ABR obtenu en insérant successivement les valeurs suivantes :
 25 8 22 35 31 4 23 40 5 38 6 34 36
- 3) Rappeler où se trouve le sommet de valeur maximum dans un ABR.
- 4) Quelle est la particularité du sommet qui porte cette valeur ?
- 5) On appelle deuxième maximum la plus grande valeur présente dans l'ABR si on enlève le maximum.
 Ecrire une fonction `deuxième-max(ref A : ABR d'entier) :entier` qui retourne la valeur du deuxième maximum d'un ABR (sans enlever le maximum de l'ABR).

Exercice 3 : On considère des tas min d'entier.

- 1) Insérer la valeur 1 dans le Tas min ci-dessous. Détaillez les différentes étapes.



- 2) Ecrire une fonction qui renvoie la valeur du second plus petit élément dans le tas.
- 3) Sans écrire la fonction indiquez où peut-on trouver la valeur du second plus grand élément de T. Justifier votre réponse.

Rappel : Primitives des Arbres Binaires.

accès

```
fonction getValeur(val S:sommet):objet;  
/* vaut NIL si le sommet n'existe pas */  
fonction filsGauche(val S:sommet):sommet;  
/* vaut NIL si S n'a pas de fils gauche */  
fonction filsDroit(val S:sommet):sommet;  
/* vaut NIL si S n'a pas de fils droit */  
fonction pere(val S:sommet):sommet;  
/* vaut NIL si S est la racine de l'arbre */
```

modification

```
fonction setValeur(ref S:sommet;val x:objet):vide;  
/* affecte au sommet S la valeur x */  
fonction ajouterFilsGauche(ref S:sommet,val x:objet):vide;  
/* filsGauche(S)==NIL doit être vérifié */  
fonction ajouterFilsDroit(ref S:sommet,x:objet):vide;  
/* filsDroit(S)==NIL doit être vérifié */  
fonction supprimerFilsGauche(ref S:sommet):vide;  
/* filsGauche(S) est une feuille */  
fonction supprimerFilsDroit(ref S:sommet):vide;  
/* filsDroit(S) est une feuille */  
fonction detruireSommet(ref S:sommet):vide;  
/* S est une feuille */
```