

Université de Perpignan Via Domitia  
Mai 2024

Durée: 2h+ $\epsilon$

Contrôle: Arbres et Algorithmes

N.B: Ecrivez lisiblement.

Aucun document.

Portable entièrement éteint et rangé confortablement dans ton sac STP.

La réponse de votre voisin(e) est peut-être fausse!

Exercice 1: (1+1+1+1+1+1,5+1,5+1,5+1,5+2+1,5)

- a) Définir un arbre parfaitement équilibré et un arbre AVL.
- b) Définir un tas et un tournoi.
- c) Donner les cinq propriétés d'un algorithme et définir l'invariant de la boucle.
- d) Donner les définitions de grand oh  $O$  et grand  $\Omega$ .
- e) Dessiner les rotations GG(Gauche Gauche) et GD(Gauche Droite) pour un arbre AVL après l'insertion d'un noeud qui a déséquilibré AVL.
- f) Dessiner tous les arbres binaires de recherche possibles avec les noeuds 1, 2, 3. Effectuer les opérations pour que les arbres soient AVL.
- g) Construire un parse tree pour l'expression arithmétique postfixe

8 2 3 \* 4 9 + \* + 1 \*

à l'aide d'une pile. Montrer l'évaluation de la pile au cours de l'évaluation. Ensuite évaluer l'expression.

- h) Construire un binaire de recherche en lisant les entiers dans l'ordre : 5, 4, 7, 2. Ensuite insérer 1. Transformer en AVL. Ensuite, insérer 3. Transformer en AVL. Ensuite, insérer 6. Transformer en AVL. Montrer les

de trois sous-sections: s231, s232, s233. Tracer cet arbre qui représente ce livre hiérarchiquement. Dessiner cet arbre sous la forme d'Indentation. Donner les trois parcours. ( 2 pt)

Exercice 2: ( 2,5 points)

Écrire une fonction récursive retournant void permettant d'insérer un entier  $x$  dans un arbre binaire de recherche:  
void insertion(int x, noeud...) {...}

Exercice 2: (1,5 points)

Soit  $P(n) = a_0 + a_1n + a_2n^2 + \dots + a_kn^k$ , un polynôme de degré  $k$ . Est-il vrai que  $P(n) = \Omega(n^k)$ ? Justifier votre réponse. Prouve ou donne un contre-exemple.

Exercice 4: (2 points) Recherche d'un entier  $x$  dans un tableau  $a$  triés  $t[1..n]$  de  $n$  entiers (dichotomie):

```
i=1; j=n; //initialisation
do
{ k = (i+j)/2;
  if (x > a[k]) i = k + 1; else j = k - 1;
}
while (a[k] != x) && ( i <= j)
```

Trouver l'invariant de la boucle. A quel point cet invariant est valable.