

Travaux Dirigés : Arbres et Algorithmes

Exercice 1: Montrer que un arbre de n noeuds possède exactement $n - 1$ flèches (arcs/pointeurs).

Exercice 2: Considérons un arbre dont chaque noeud possède au plus (\leq) d fils. Si la hauteur de l'arbre est h , alors trouver le nombre maximum de noeuds l'arbre peut contenir.

Exercice 3: Écrire un algorithme (puis un programme en C) permettant de saisir un arbre avec un tableau de papa. Écrire un algorithme pour: Trouver toutes les feuilles de l'arbre et la hauteur de l'arbre.

Exercice 4: Écrire un algorithme (puis un programme en C) permettant de saisir un arbre avec les listes de ses fils. Votre programme écrira la liste de fils de chaque noeud.

Exercice 5:
Construire un arbre binaire de recherche avec les valeurs de noeuds:

8,9,11,15,19,20,21,7,3,2,1,5,6,4,13,14,10,12,17,16,18.

Exercice 6:
Construire un arbre binaire parfaitement équilibré avec les valeurs de noeuds:
21, 8,9,11,15,19,20,21,7,3,2,1,5,6,4,13,14,10,12,17,16,18. Le

premier entier $n(=21)$ est le nombre de noeuds dans l'arbre.

Un arbre binaire parfaitement équilibré: Un arbre binaire est parfaitement équilibré si pour chaque noeud le nombre de noeuds dans l'arbre gauche et l'arbre droit diffère par au plus 1.

Algorithme: Utiliser un noeud pour la racine. 2. Générer l'arbre gauche avec $ng = n\%2$ noeuds récursivement. 3. Générer l'arbre droit avec $nd = n-ng-1$ noeuds récursivement.

Exercice 7:

Écrire une fonction récursive permettant de créer un arbre binaire de recherche.

Exercice 8:

Écrire une fonction permettant d'imprimer un arbre binaire sous la forme "d'indentation."

Exercice 9:

Écrire une fonction récursive pour: 1. Parcours préfixe 2. Parcours infixé, 3. Parcours postfixé d'un arbre binaire.

Exercice 10: Écrire une fonction récursive en pseudo-code pour: 1. Parcours préfixe 2. Parcours infixé, 3. Parcours postfixé d'un arbre quelconque.