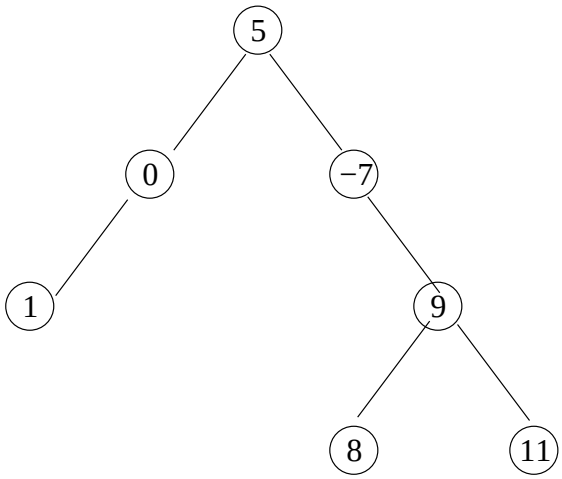
Travaux Pratiques d'informatique

Algorithmique

Codage d'un arbre strictement binaire et arbre binaire de recherche.

1. **Codage d'un arbre étiquetté quelconque**
   On considère des arbres binaires quelconques, etiquettés par des entiers relatifs. Toute valeur peut apparaitre dans l'arbre.
   Tout type de nœud peut apparaitre dans l'arbre: 0 fils, un seul fils droit, 1 seul fils gauche, deux fils. Pour décrire un tel arbre, on effectue un parcours en profondeur de l'arbre en indiquant pour chaque nœud rencontré son type et son étiquette. Pour le type du nœud on convient d'utiliser

   - 0 pour une feuille nœud interne,
   - 1 pour u nœud avec un seul fils droit,
   - 2 pour un nœud avec un seul fils gauche,
   - 3 pour un nœud avec 2 fils

   Ainsi l'arbre
   est codé par la suite 3 5 2 0 0 1 1 -7 3 9 0 8 0 11.

   (a) Écrire une fonction `int construitArbreQuelconque(Arbre *a,FILE *in)` qui construit un arbre à partir d'une telle suite lue dans le fichier manipulé grace à `in`.

   (b) Écrire une fonction `void ecritArbreQuelconque(Arbre a, FILE *out)` qui écrit dans le fichier manipulé par `out` la suite décrivanr l'arbre `a`

2. **Arbre binaire de recherche** Écrire les fonctions de manipulation d'arbre binaire de recherche sans doublon. Les étiquettes sont des entiers.

   (a) `Arbre Recherche(Arbre a, int n)` qui recherche un élément n dans l'arbre `a`. Elle renvoie l'adresse du nœud contenant l'élément s'il est présent, NULL sinon.

   (b) `int ajout(Arbre *a, int n)` qui effectue l'ajout de l'élément n dans l'arbre binaire de recherche `*a`.

   (c) `Arbre extraitMin(Arbre *a)` qui effectue l'extraction du nœud contenant la plus petite étiquette de l'arbre binaire de recherche `a`. Elle renvoie l'adresse du nœud correspondant et NULL en cas d'echec.

(d) `Arbre extraitMax(Arbre *a)`

(e) `Arbre extrait(Arbre a, int n)` qui effectue l'extraction du nœud contenant une étiquette donnée d'un arbre binaire de recherche. Elle renvoie l'adresse du nœud et NULL en cas d'échec.

3. **Vérification**
   Ecrire une fonction qui détermine si un arbre est un arbre binaire de recherche. Utiliser la première question pour tester différents arbres.