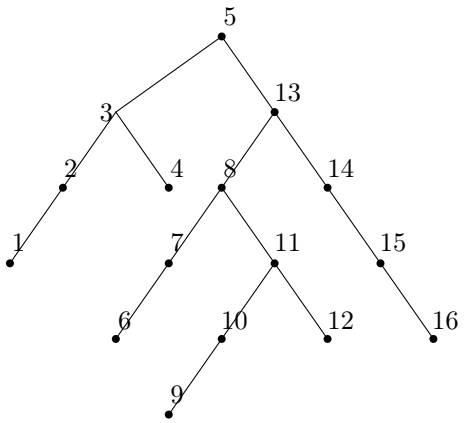
Analyse d'algorithmes (GLIN 401)

Une feuille manuscrite recto verso autorisée

1 Questions de cours

Question 1 On dispose de deux arborescences (généralisées, c'est à dire non binaires) A_1 et A_2, chacune de taille n, à clés entières, toutes distinctes pour une même arborescence. Donner un algorithme en $\mathcal{O}(n^2)$ qui calcule le nombre de clés communes aux deux arborescences (justifier la complexité).
Voyez vous un moyen d'améliorer la complexité de cet algorithme ? (justifier)

Question 2 On veut supprimer le sommet d'étiquette 8 dans l'ABR ci dessous.



- indiquez les différentes étapes de l'algorithme
- indiquer comment modifier le dessin pour représenter le résultat obtenu par l'algorithme

Question 3 On suppose que des entiers compris entre 1 et 1000 sont disposés dans un arbre binaire de recherche et que l'on souhaite trouver le nombre 363. Parmi les séquences suivantes, lesquelles **ne** pourraient **pas** être la suite des nœuds parcourus ? Justifiez chacune de vos réponses (ne pas justifier les réponses indiquant les séquences qui peuvent être la suite des nœuds parcourus).

1. 2, 252, 401, 398, 330, 344, 397, 363.
2. 924, 220, 911, 244, 898, 258, 362, 363.
3. 925, 202, 911, 240, 912, 245, 363.
4. 2, 399, 387, 219, 266, 382, 381, 278, 363.
5. 935, 278, 347, 621, 299, 392, 358, 363.

2 ABR

Dans l'arbre binaire de recherche que nous considérons dans ce problème, toutes les clés sont différentes.
Pour un sommet S, on notera $cle(S)$ la clé de ce sommet.

Analyse théorique pour établir l'algorithme

1. à quelle condition un sommet S est-il le sommet ayant la plus petite clé ? Si S n'est pas le sommet qui a la plus petite clé, et si toutes les clés sont différentes, on appelle $Pred(S)$ (son *prédécesseur*) le sommet qui a la clé imédiatement inférieure à celle de S.
2. Dans le premier cas que nous examinons, S a une sous arborescence gauche
   (a) où pensez vous que se trouve alors $Pred(S)$? appelez ce sommet $Max(S)$ dans les explications suivantes par lesquelles vous allez démontrer que votre intuition est juste.
   (b) expliquez pourquoi pour tout sommet T
      i. qui se trouve dans la sous arborescence gauche de S, $T \neq Max(S) \Rightarrow T$ ne peut pas être le prédécesseur de S
      ii. qui se trouve dans la sous arborescence droite de S, T ne peut pas être le prédécesseur de S
      iii. qui se trouve sur le chemin de S à la racine, T ne peut pas être le prédécesseur de S (il faudra examiner deux cas, suivant que S est dans la sous arborescence gauche ou droite de T)
      iv. qui se trouve ailleurs dans l'arbre, T ne peut pas être le prédécesseur de S (il faudra considérer U l'ancêtre commun à S et T le plus bas dans l'arbre, puis examiner deux cas)
3. Dans le deuxième cas que nous examinons, S n'a pas de sous arborescence gauche :
   (a) indiquez alors où vous pensez que se trouve $Pred(S)$ et appelez le $Bas(S)$ dans les explications ultèrieures.
   (b) prouvez le de la même manière que pour le premier cas

Algorithme et analyse complexité

Dans un arbre binaire de recherche dont toutes les clés sont différentes, pour lequel on dispose des fonctions (en $\Theta(1)$)

- $racine?(S)$ qui indique si un sommet S est la racine de l'arbre
- $pere(S)$, définie pour les autres sommets que la racine et qui renvoie le père du sommet S
- $val(S)$ qui renvoie l'étiquette du sommet S
- $sag(S)$ qui renvoie la racine de la sous arborescence gauche du sommet S
- $sad(S)$ qui renvoie la racine de la sous arborescence droite du sommet S
- $fd?(S)$, définie pour les autres sommets que la racine et qui indique si un sommet S est le fils droit de son père (si $fd?(S)$ vaut *faux*, S est le fils gauche de son père)
- $nonVide?(A)$ qui indique que l'arborescence A n'est pas vide

1. écrivez la fonction $pred(S)$ (l'utilisation de fonctions auxiliaires sera bien vue).
2. Donnez et justifiez la classe de complexité de votre fonction en fonction de la hauteur h de l'ABR.