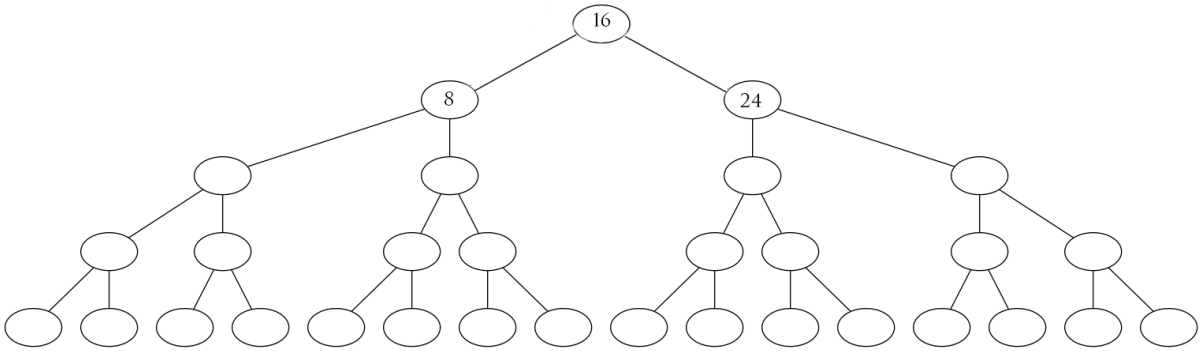
Arbres Binaires de Recherche (ABR)

0. Introduction.

Exercice 1 :

Alice choisit un nombre entier naturel compris entre 1 et 31. Bob cherche à trouver le plus rapidement ce nombre en posant des questions à Alice.

1. Quelle stratégie vue en première permettra à Bob de trouver le plus rapidement le nombre qu'a choisi Alice ?
2. Cette stratégie peut se représenter à l'aide d'un arbre binaire.Compléter cet arbre binaire.



3. Combien d'étapes au maximum sont nécéssaires pour trouver le nombre qu'a choisi Alice ?

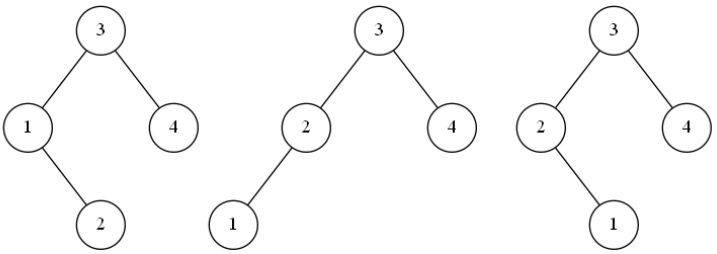
Réponse :

1. Définition

- Un arbre binaire de recherche (ABR par la suite) est un arbre binaire dont les noeuds contiennent des valeurs qui peuvent être comparées entre elles et tel que que pour tout noeud de l'arbre, toutes les valeurs situées dans le sous-arbre gauche (ou droit) sont plus petites (ou plus grandes) que la valeur située dans le noeud.
- Vocabulaire : On emploie aussi le terme "clé" à la place de "valeur".

Exercice 2 :

Parmi ces trois arbres binaires, lesquels sont des arbres binaires de recherche ?



Réponse:

Exercice 3 :

Dans quel ordre seront visités les noeuds de ces arbres en parcours infixe ? Que remarque-t-on ?

Réponse :

Remarques :

- Un ABR est un arbre binaire, le vocabulaire déjà abordé reste valable (taille , hauteur, sous-arbres gauche et droite, parcours préfixe, postfixe et surtout infixe).
- Le parcours en profondeur infixe d'un ABR visite les noeuds de l'arbre dans l'ordre croissant de leurs étiquettes. En effet, il affiche de façon récursive d'abord le sous-arbre gauche (valeurs plus petites), la racine, puis le sous-arbre droit(valeurs plus grandes).
- Les ABR se révèlent très efficaces dans la recherche d'informations.

Exercice 4 :

Donner tous les ABR formés de trois noeuds et contenant les valeurs 1, 2 et 3.

Réponse :

2. Recherche dans un ABR

- Pour chercher une valeur dans un ABR, il suffit de la comparer à la valeur à la racine puis, si elle est différente, de se diriger vers un seul des deux sous-arbres. On élimine ainsi complètement la recherche dans l'autre sous-arbre.
- Cette recherche de valeur se décrit aisément de façon récursive :
  - Si l'arbre est vide, la réponse est évidente, la valeur n'est pas dans l'arbre.
  - Dans le cas contraire, l'arbre contient au moins un noeud. on peut donc comparer la valeur avec celle de la racine :
    - Si la valeur est plus petite, on continue la recherche dans le sous-arbre gauche, ce que l'on fait récursivement.
    - Si la valeur est plus grande, on continue la recherche dans le sous-arbre droit, ce que l'on fait récursivement.
    - Sinon, la valeur est égale à celle de la racine.

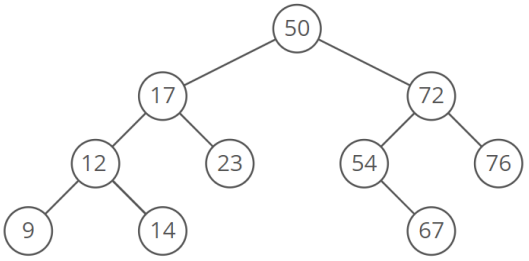
- Autrement écrit :

$recherche(x, arbre)$:
    $si\ arbre\ est\ vide$
        $renvoyer\ False$
    $sinon$
        $si\ x < arbre.racine.valeur$
            $recherche(x, arbre.gauche)$
        $si\ x > arbre.racine.valeur$
            $recherche(x, arbre.droit)$
        $sinon$
            $renvoyer\ True$

Exercice 5 :

On considère l'ABR ci-contre.

1. Ecrire la liste des noeuds affichés par un parcours infixe.
2. Décrire ce qui se passe lors de la recherche :
   - De la valeur 23
   - De la valeur 9
   - De la valeur 15



Réponse:

Efficacité

- Lorsque les éléments sont répartis à peu près équitablement entre les deux sous-arbres, la recherche élimine la moitié des éléments à chaque étape. C'est le même principe que pour la recherche dichotomique dans un tableau trié (voir cours de première).
- Dans des cas particuliers ou il faut parcourir l'arbre dans sa totalité, pour éventuellement arriver à une recherche infructueuse, cette recherche est alors moins efficace.
- D'une manière générale , le nombre d'étapes ne peut pas dépasser la hauteur de l'arbre.
- Or, la hauteur de l'arbre dépend justement de la façon dont l'arbre est construit. Pour une recherche efficace, la construction d'un ABR de hauteur minimale est donc intéréssante.

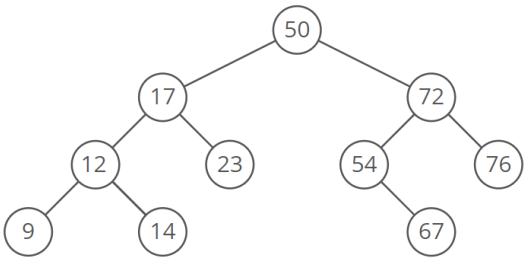
3. Ajout

- En principe, ajouter un nouvel élément dans un ABR n'est pas plus compliqué que de le chercher :
  - S'il est plus petit, on va à gauche.
  - S'il est plus grand, on va à droite.
  - Quand on arrive à un arbre vide, on ajoute un nouveau noeud.
- Autrement écrit :

$ajoute(x, arbre)$:
    $si\ arbre\ est\ vide$
        $créer\ noeud(x)$
    $sinon$
        $si\ x < arbre.racine.valeur$
            $ajoute(x, arbre.gauche)$
        $sinon$
            $ajoute(x, arbre.droit)$

Exercice 6 :

On considère le même arbre que précédemment.



1. Ou seront ajoutées les valeurs :

- 51
- 2
- 81
- 23

2. Plus généralement :

- Dans un ABR, ou se trouvent :
  - La valeur la plus petite ?
  - La valeur la plus grande ?

Réponse :

Remarques :

- Tel que l'algorithme est décrit , on peut ajouter une valeur déjà présente dans l'arbre. Plus précisément, un doublon sera ajouté dans le sous arbre droit du noeud dont la valeur est déjà présente dans l'arbre.
- On pourrait modifier cet algorithme pour que les doublons ne soient pas possible ou encore qu'ils soient ajoutés dans un sous-arbre gauche.
- En pratique, nous verrons que l'implémentation en python est un peu plus compliquée, car l'ajout dans un arbre vide pose problème.
- La suppresion d'un noeud dans un ABR suit également une démarche récursive similaire mais elle est plus complexe et dépasse le cadre du programme de terminale.

4. Arbre équilibré

Exercice 7 :

On considère un ABR de 4 noeuds contenant les clés 1 , 2 , 3 et 4.

1. Quelle est la hauteur minimale de cet ABR ? Dessiner un tel arbre.
2. Quelle est sa hauteur maximale ? Dessiner un tel arbre.

Réponse:

Remarques :

- Dans des cas particuliers ou il faut parcourir l'arbre dans sa totalité, pour éventuellement arriver à une recherche infructueuse, cette recherche est moins efficace.
- D'une manière générale , le nombre d'étapes ne peut pas dépasser la hauteur de l'arbre.
- Or, la hauteur de l'arbre dépend justement de la façon dont l'arbre est construit.Pour une recherche efficace, la construction d'un ABR de hauteur minimale est donc intéréssante. :
  - Dans le pire des cas un ABR de N noeuds a pour hauteur N.
  - Dans le meilleur des cas, la hauteur de l'arbre est dite logarithmique(de base 2).
- S'assurer de l'équilibrage d'un ABR repose sur des algorithmes qui dépassent largement le programme de terminale.

Exercice 8 :

1. Dessiner un arbre binaire de recherche de taille minimale contenant les 26 lettres de l'alphabet.
2. Quelle est sa hauteur ?
3. Quels noeuds seront visités lors de la recherche :
   - De la lettre H ?
   - De la lettre E ?