Exercices Type Bac

Pile, File et arbre binaire (théorie)

Exercice 1 : arbre binaire (partie 1)

On rappelle qu'un arbre binaire est composé de nœuds, chacun des nœuds possédant éventuellement un sous-arbre gauche et éventuellement un sous-arbre droit. Un nœud sans sous-arbre est appelé feuille. La taille d'un arbre est le nombre de nœuds qu'il contient ; sa hauteur est le nombre de nœuds du plus long chemin qui joint le nœud racine à l'une des feuilles. Ainsi la hauteur d'un arbre réduit à un nœud, c'est-à-dire la racine, est 1.

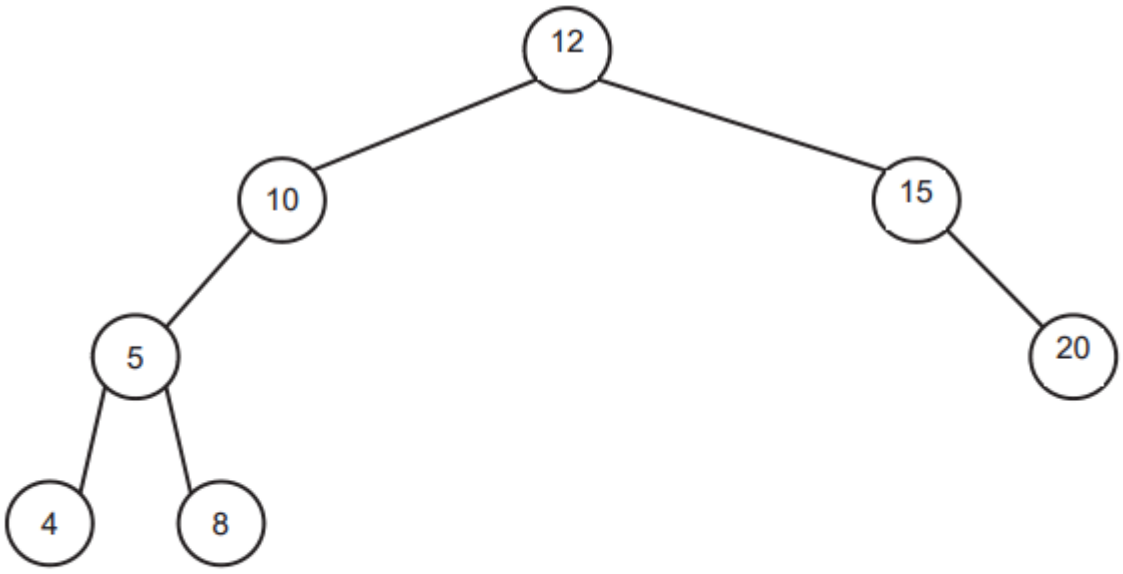
Dans un arbre binaire de recherche, chaque nœud contient une clé, ici un nombre entier, qui est :

- strictement supérieure à toutes les clés des nœuds du sous-arbre gauche ;
- strictement inférieure à toutes les clés des nœuds du sous-arbre droit.

Ainsi les clés de cet arbre sont toutes distinctes.

Un arbre binaire de recherche est dit « bien construit » s'il n'existe pas d'arbre de hauteur inférieure qui pourrait contenir tous ses nœuds.

On considère l'arbre binaire de recherche ci-dessous.



1. **a**. Quelle est la taille de l'arbre ci-dessus ?
   b. Quelle est la hauteur de l'arbre ci-dessus ?

2. Cet arbre binaire de recherche n'est pas « bien construit ». Proposer un arbre binaire de recherche contenant les mêmes clés et dont la hauteur est plus petite que celle de l'arbre initial.

3. Donner le parcours préfixe et infixe de cet arbre.

Exercice 2 : Pile (écriture polonaise)

La notation polonaise inverse (NPI) permet d'écrire des expressions de calculs numériques sans utiliser de parenthèse. Cette manière de présenter les calculs a été utilisée dans des calculatrices de bureau dès la fin des années 1960. La NPI est une forme d'écriture d'expressions algébriques qui se distingue par la position relative que prennent les nombres et leurs opérations.

Par exemple :

Notation classique	Notation NPI
3+9	3 9 +
8×(3+5)	8 3 5 + ×
(17+5)×4	17 5 + 4 ×

L'expression est lue et évaluée de la gauche vers la droite en mettant à jour une pile.

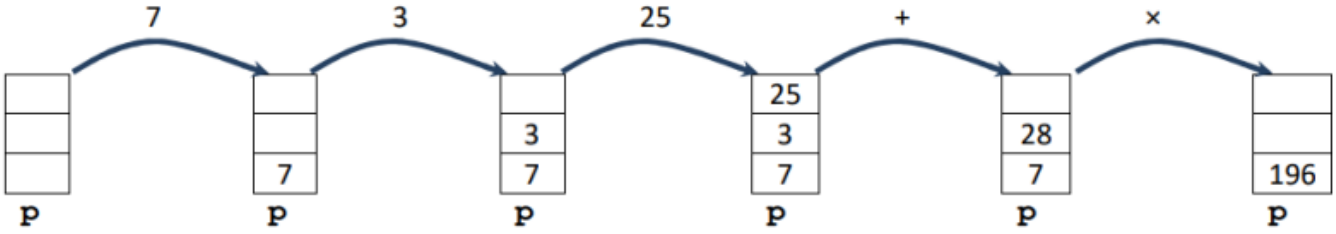
- Les nombres sont empilés dans l'ordre de la lecture.
- Dès la lecture d'un opérateur (+, -, ×, /), les deux nombres au sommet de la pile sont dépilés et remplacés par le résultat de l'opération effectuée avec ces deux nombres. Ce résultat est ensuite empilé au sommet de la pile.

A la fin de la lecture, la valeur au sommet est renvoyée.

Exemple : l'expression `7 3 25 + ×` qui correspond au calcul `7×(3+25)` s'évalue à 196 comme le montrent les états successifs de la pile créée, nommée `p` :

- On empile la valeur 7.
- On empile la valeur 3.
- On empile la valeur 25.
- On remplace les deux nombres du sommet de la pile (25 et 3) par leur somme 28.
- On remplace les deux nombres du sommet de la pile (28 et 7) par leur produit 196.

Schéma descriptif des différentes étapes d'exécution.



1. En vous inspirant de l'exemple ci-dessus, dessiner le schéma descriptif de ce que donne l'évaluation par la NPI de l'expression `12 4 5 × +`.

2. On dispose de la pile suivante nommée `p1` :

On rappelle ci-dessous les primitives de la structure de pile (LIFO : Last In First out) :

Fonction	Description
`pile_vide()`	Créé et renvoie une nouvelle pile vide
`empiler(p, e)`	Place l'élément **`e`** au sommet de la pile **`p`**.
`depiler(p)`	Supprime et renvoie l'élément se trouvant au sommet de **`p`**.
`est_vide(p)`	Renvoie un booléen indiquant si **`p`** est vide ou non.

On dispose aussi de la fonction suivante, qui prend en paramètre une pile `p` :

```
def top(p) :
    x = depiler(p)
    empiler(p, x)
    return x
```

On exécute la ligne suivante `temp = top(p1)` :

a. Quelle valeur contient la variable `temp` après cette exécution ?

b. Représenter la pile `p1` après cette exécution.

3. En utilisant uniquement les 4 primitives d'une pile, écrire en langage Python la fonction `addition(p)` qui prend en paramètre une pile p d'au moins deux éléments et qui remplace les deux nombres du sommet d'une pile p par leur somme. Remarque : cette fonction ne renvoie rien, mais la pile p est modifiée.

4. On considère que l'on dispose également d'une fonction `multiplication(p)` qui remplace les deux nombres du sommet d'une pile p par leur produit (on ne demande pas d'écrire cette fonction). Recopier et compléter, en n'utilisant que les primitives d'une pile et les deux fonctions `addition` et `multiplication`, la suite d'instructions (ci-dessous) qui réalise le calcul (3+5)×7 dont l'écriture en NPI est : 3 5 + 7 ×

```
p=pile_vide()
empiler(p,3)
```

Corrigé

Exercice 1 :

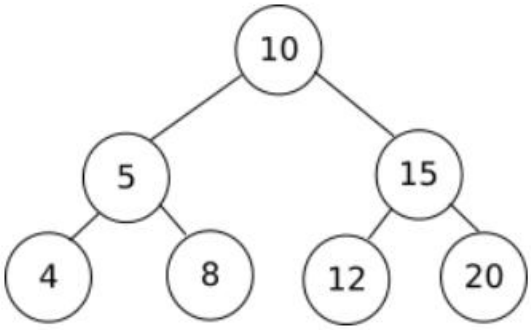
1a

taille de l'arbre = 7

1b

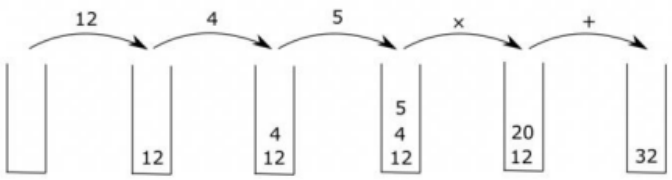
hauteur de l'arbre = 4

2



Exercice 2 :

1.



2.

a. la variable *temp* contient la valeur 25

b.

3.

```
def addition(p):
    v1 = depiler(p)
    v2 = depiler(p)
    empiler(p, v1+v2)
```

4.

```
p = pile_vide()
empiler(p,3)
empiler(p,5)
addition(p)
empiler(p,7)
multiplication(p)
```