

Proposition de correction

Exercice 1

Q1.a

chaque nœud de l'arbre a au plus deux fils

Q1.b

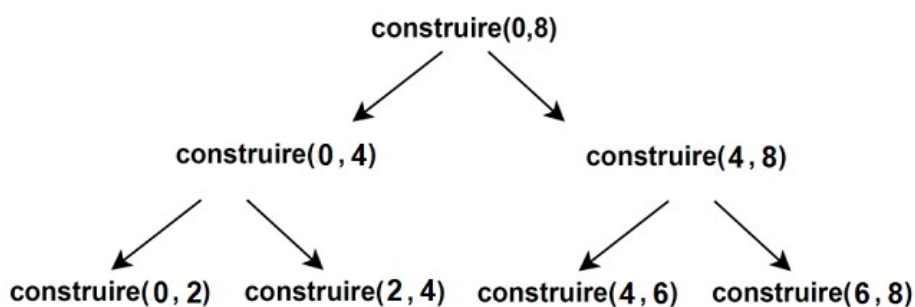
Les clés des nœuds du sous-arbre gauche ne sont pas inférieures aux clés du sous-arbre droit.

Ce n'est pas un ABR

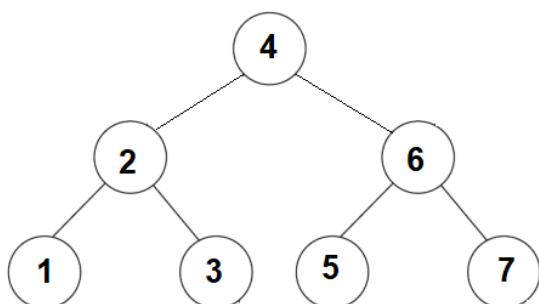
Q2.a

```
assert isinstance(mini, int) and isinstance(maxi, int) and mini <= maxi
```

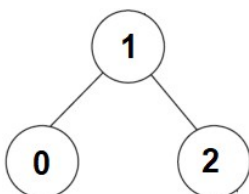
Q2.b



Q2.c



Q2.d



Q2.e

1, 2, 3, 4, 5, 6, 7

parcours infixe → valeurs triées

Q2.f

```
def maximum(abr):  
    if abr is None:  
        return None  
    elif abr.droit is None:  
        return abr.valeur  
    else :  
        return maximum(abr.droit)
```

Q3.a

assert mystere(abr_7_noeuds, 5, []) == [6, 4, 5]

assert mystere(abr_7_noeuds, 6, []) == [6]

assert mystere(abr_7_noeuds, 2, []) == []

Q3.b

Parcours de l'arbre jusqu'à la valeur cherchée

Exercice 2

Q1.a

- Mohamed, Ali
- Fernando, Alonso

Q1.b

Harry Kane

Q2.a

Dupont, Antoine

Kane, Harry

Q2.b

SELECT tarif

FROM Objet

WHERE description = "Scie circulaire"

LIMIT 1

Q2.c

```
UPDATE Objet  
SET tarif = 15  
WHERE id_objet = 1
```

Q2.d

```
INSERT INTO Membre  
VALUES(6, 'Renard', 'Wendie', '69100')
```

Q3.a

Un membre ne pourrait pas réserver une nouvelle fois le même objet.

Q3.b

La clé primaire sur Mohamed Ali est une clé étrangère de la table Possede

Q3.c

```
DELETE FROM Possede WHERE id_membre = 1  
DELETE FROM Objet WHERE id_objet = 6  
DELETE FROM Membre WHERE id_membre = 1
```

Q4.a

```
SELECT COUNT(*)  
FROM Reservation, Membre  
WHERE Membre.prenom = 'Fernando' AND Membre.nom = 'Alonso'  
AND Reservation.id_membre = Membre.id_membre
```

Q4.b

```
SELECT Membre.nom, Membre.prenom  
FROM Membre, Possede, Objet  
WHERE Objet.description = 'Appareil à raclette'  
AND Possede.id_objet = Objet.id_objet  
AND Membre.id_membre = Possede.id_membre
```

Exercice 3

Q1.a

```
6217
|---- 8887
      |---- 9617
            |---- 9657
            |---- 9697
            |---- 9750
            |---- 9794
            |---- 9795
```

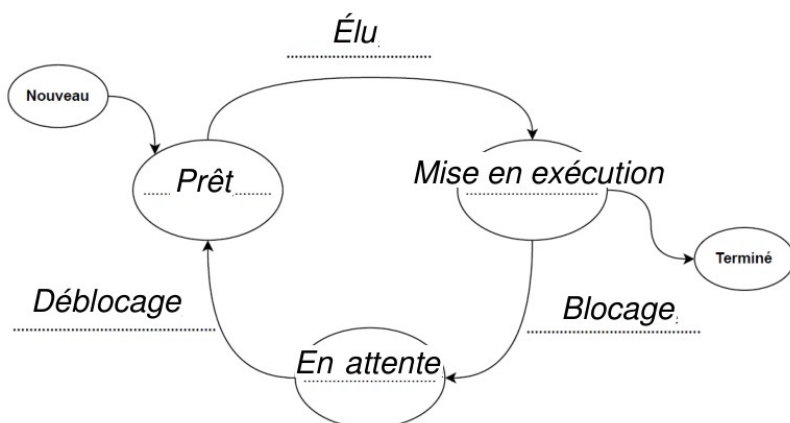
Q1.b

bash

Q1.c

kill | cut -f1 | ps -aef | grep firefox (ou kill 9617 si PID connu)

Q2a



Q2.b

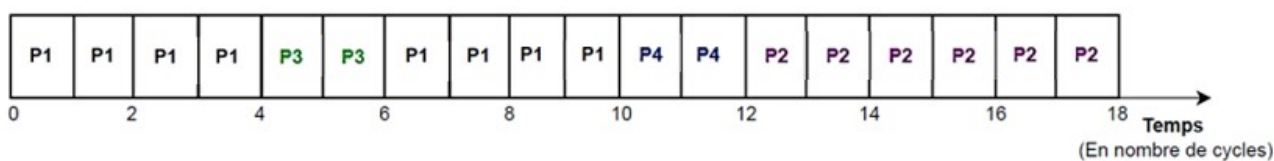
$$\bar{t}_{p1} = 8 / (12 - 0) = 0,75$$

$$\bar{t}_{p2} = 6 / (18 - 12) = 1$$

$$\bar{t}_{p3} = 2 / (5 - 3) = 1$$

$$\bar{t}_{p4} = 2 / (9 - 7) = 1$$

Q2.c



$$\bar{t}_{P1} = 8 / 10 = 0,8$$

$$\bar{t}_{P2} = 6 / 6 = 1$$

$$\bar{t}_{P3} = 2 / 2 = 1$$

$$\bar{t}_{P4} = 2 / 2 = 1$$

gain pour P1

Q3.a

```
def choix_processus(liste_attente):
    """Renvoie l'indice du processus le plus court parmi
    ceux présents en liste d'attente liste_attente"""
    if liste_attente != []:
        mini = len(liste_attente[0])
        indice = 0
        for i in range(1, len(liste_attente)):
            if mini > (len(liste_attente[i]) - liste_attente[i].count(")):
                mini = len(liste_attente[i]) - liste_attente[i].count("")
                indice = i
        return indice
```

Q3.b

```
def ordonnancement(liste_proc):
    """Exécute l'algorithme d'ordonnancement
    liste_proc -- liste des processus
    Renvoie la liste d'exécution des processus"""
    execution = []
    attente = scrutation(liste_proc, [])
    while attente != []:
        indice = choix_processus(attente)
        execution.append(attente[indice].pop())
        attente = scrutation(liste_proc, attente)
    return execution
```