Proposition de correction

Exercice 1

Q1

Idgrp : INT

nom : CHAR

style : CHAR

nb_pers : INT

Q2

Un même musicien peut jouer de plusieurs instruments et il peut exister des homonymes.

Q3

Weather Report

Return to Forever

Q4

UPDATE concerts

SET heure_fin = ‘22h30’

WHERE idconc = 36 ;

Q5

SELECT groupes.nom

FROM groupes, concerts

WHERE concert.scene = 1 AND concert.idgrp = groupes.idgrp

ORDER BY groupes.nom

Q6

INSERT INTO groupes

VALUES(15, 'Smooth Jazz Fourplay', ‘Free Jazz’, 4)

Q7

```
def recherche_nom (musiciens : dict) -> list :
    noms = []
    for m in musiciens:
            if m['nb_concerts'] >= 4:
                    noms.append(m['nom'])
```

```
    return noms
```

Exercice 2

Q1

@IP 172.16.2.3/24 → @réseau 172.16.2.0 (LAN2), donc Alice

Q2

$$coût = \frac{débit\ maximal\ de\ référence}{débit\ du\ réseau\ concerné} = \frac{10000}{1000} = 10$$

Q3

Routeur R6		
Destination	**Pass.**	**Coût**
LAN1	R5	21
LAN2	-	-
WAN1	R5	11
WAN2	R5	21
WAN3	R5	11
WAN4	R5	20
WAN5	R5	10
WAN6	-	-
WAN7	-	-
WAN8	R5	10

Q4

R1 → R2 → R5 → R6

Q5

R5

Exercice 3

Q1

File (FIFO) : premier arrivé, premier servi

Q2.a

Taille

Q2.b

racine

Q2.c

feuille

Q3.a

tache, indice, gauche, droite

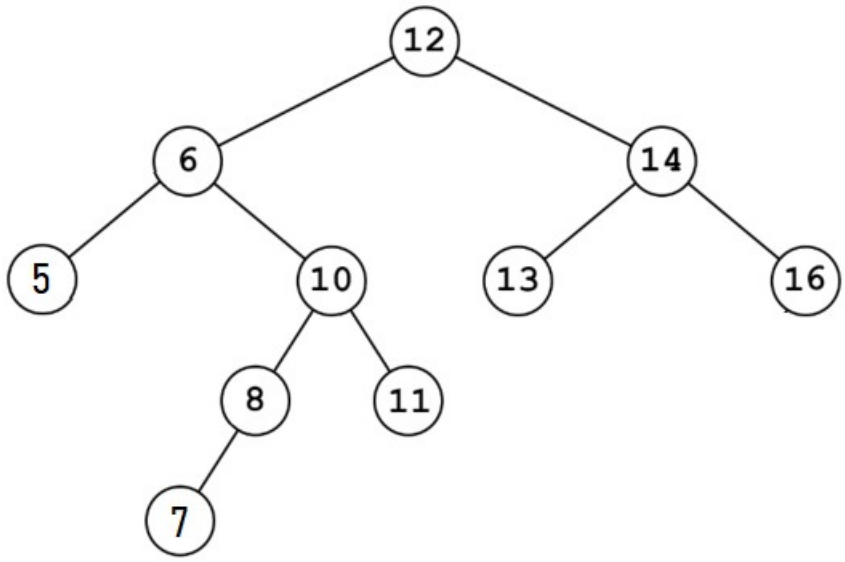
Q3.b

la méthode s'appelle elle-même (se déplace dans l'arbre jusqu'à la racine et termine)

Q3.c

>

Q3.d



Q4

```
def est_present(self, indice_recherche) :
    """renvoie True si l'indice de priorité indice_recherche
    (int) passé en paramètre est déjà l'indice d'un nœud de l'arbre, False sinon"""
    if self.est_vide():
            return False
    elif self.racine.indice < indice_recherche :
            return self.racine.gauche.est_present(indice_recherche) :
    elif self.racine.indice > indice_recherche :
            return self.racine.droite.est_present(indice_recherche) :
    else :
            return True
```

Q5.a

6, 8, 10, 12, 13, 14

Q5.b

Les tâches sont triées par ordre prioritaire de façon ascendante

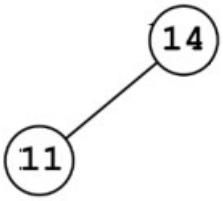
Q6

```
def tache_prioritaire(self):
    """renvoie la tache du noeud situé le plus
    à gauche de l'ABR supposé non vide"""
    if self.racine.gauche.est_vide():          #pas de nœud plus à gauche
            return self.racine.tache
    else:
            return self.racine.gauche.tache_prioritaire()
```

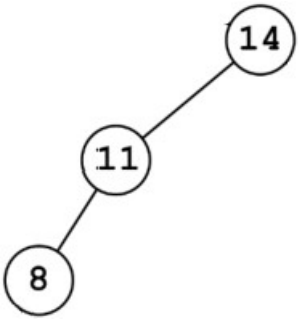
Q7

Étape 1

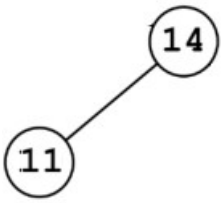
Étape 2



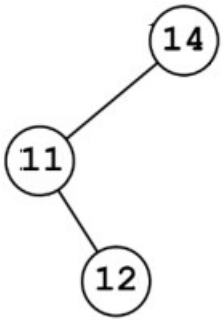
Étape 3



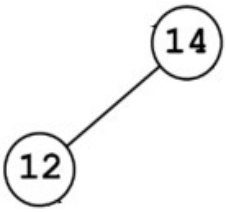
Étape 4



Étape 5

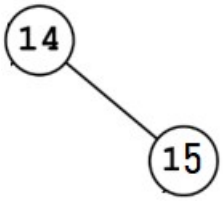


Étape 6

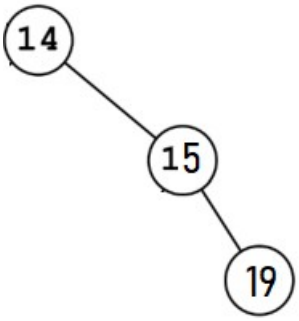


Étape 7

Étape 8



Étape 9



Étape 10