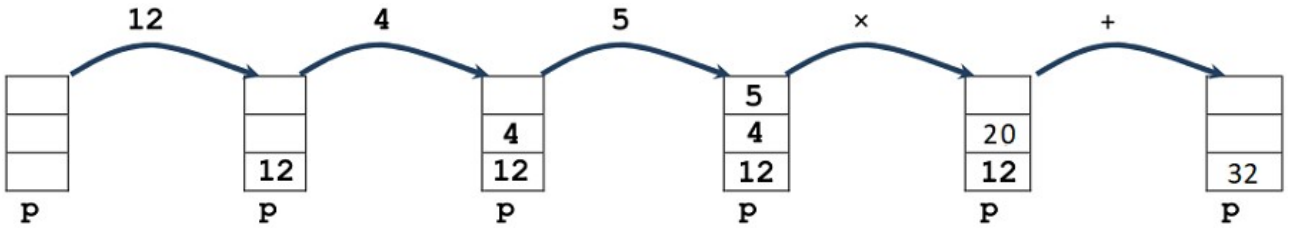
Proposition de correction

Exercice 1

Q1



Q2.a

25

Q2.b

Q3

```
def addition(p : object) :
    if len(p) > 1 :
            x = depiler(p) + depiler(p)
            empiler(p, x)
```

Q4

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

Exercice 2

Q1

Un client peut réserver plusieurs chambres (eg : agence de voyage, entreprise, ...)

Q2.a

SELECT Nom, Prenom

FROM Clients

ORDER BY Nom, Prenom

NB : on ne gère pas les homonymes

Q2.b

SELECT Telephone

FROM Clients

WHERE Prenom = "Grace" AND Nom = "Hopper"

Q3

SELECT NumChambre

FROM Reservations

WHERE date(DateArr) >= date('2024-12-28') AND date(DateDep) <= date('2024-12-28')

ORDER BY NumChambre

Q4.a

UPDATE Chambres

SET Prix = 75

WHERE NumChambre = 404

Q4.b

SELECT DISTINCT Reservations.NumChambre

FROM Reservations, Clients

WHERE Clients.Prenom = "Edgar" AND Clients.Nom = "Codd"

AND Clients.NumClient = Reservations.NumClient

ORDER BY Reservations.NumChambre

Exercice 3

Q1.a

8

Q1.b

$2^8 = 256$

Q1.c

[0 – 255]

Q2.a

$65 = 64 + 1 = 0100\ 0001_2$

Q2.b

$0011\ 1010_2 = 32 + 16 + 8 + 2 = 58$

Q2.c

$+58 = 0011\ 1010_2$

$-59 = 1100\ 0101_2$

$-58 = 1100\ 0110_2$

Q2.d

$0100\ 0001_2$ = 65

$+\ 1100\ 0110_2$ = -58

$= 0000\ 0111_2$ = 7

Q3.a

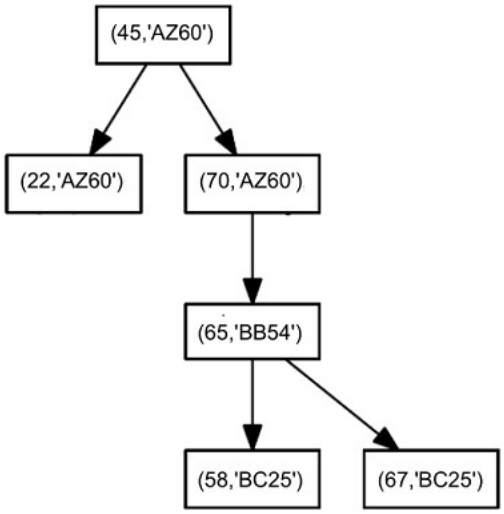
mv ./pierre/documents/saxo.mp3 ./pierre/musiques/saxo.mp3

Q3.b

mv ./pierre/bazarre ./pierre/videos

Exercice 4

Q1.a



Q1.b

Parcours infixe

Q2

```
fonction taille(a)
    si a est null
            alors renvoyer 0
    sinon
            renvoyer 1 + taille(filsgauche(a)) + taille(filsdroit(a))
```

Q3.a

NB : on suppose que l'appel est renvoyer « mystere(filsgauche(a), n) OU mystere(filsdroit(a), n) »

La fonction renvoie Vrai si le n° de billet n est trouvé dans l'arbre ou Faux sinon

Q3.b

```
fonction mystereABR(a, n)
    si a est null alors
            renvoyer Faux
    sinon si billet(a) vaut n alors
            renvoyer Vrai
    sinon si billet(a) < n alors
            renvoyer mystereABR(filsgauche(a), n)
    sinon
            renvoyer mystereABR(filsdroit(a), n)
```

Exercice 5

Q1

```
def autre(x : int) -> bool :
    return not x      # par défaut renvoie False pour tout x ≠ 0
```

Q2.a

```
def nbValeurs(li : int, v : int) -> int :
    if len(grille) > li :
            return grille[li].count(v)
    return None
```

Q2.b

```
def regle1(li : int)
    zero = nbValeurs(li, 0)
    un   = nbValeurs(li, 1)
    if zero is not None and un is not None :
            # S'il n'y a pas assez de 0 ou de 1, la fonction ne modifie rien.
            if max(zero, un) * 2 <= len(grille[li]) :
                    valeur = 0 if zero < un else 1
                    mini, maxi = min(zero, un), max(zero, un)
                    for i in range(mini, maxi) :
                            vide = grille[li].index(-1)
                            grille[li][vide] = valeur
```

Q3

```
def regle3(li)
    for col in range(...):
            if grille[li][col] == grille[li][col+2] and grille[li][col+1] == -1:
                    grille[li][col+1] = autre(grille[li][col])
```

Q4

```
def convert(L : list) -> int :
    decimal = 0
    puissance = 2 ** (len(L) - 1)
    for i in L :
            decimal += i * (2 ** puissance)
            puissance //= 2
    return decimal
```

Q5

```
fonction doublon(V : liste d'entiers) : booléen
{ renvoie Vrai s'il existe un doublon dans V, sinon renvoie Faux }
début
    L : liste d'entiers := []
    pour chaque elt de V faire
            si elt dans L alors
                    renvoyer Vrai
            sinon
                    L := L + [elt]
    renvoyer Faux
fin
```