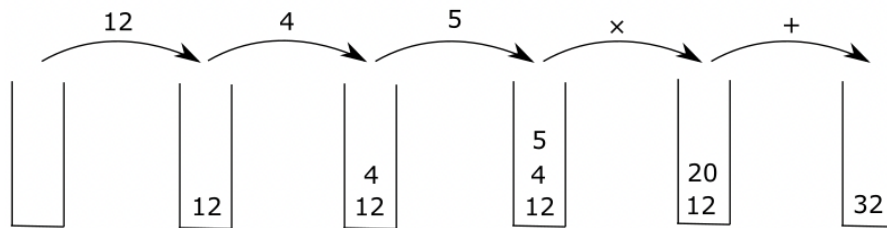


Éléments de correction sujet 11 (2022)

Exercice 1

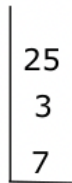
1.



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)
```

Exercice 2

1. Le couple (NumClient, NumChambre) ne peut pas jouer le rôle de clé primaire, car un client peut réserver plusieurs fois la même chambre (à des dates différentes).

2.

a.

```
SELECT Nom, Prenon  
FROM Clients
```

b.

```
SELECT Telephone  
FROM Clients  
WHERE Nom = 'Hopper' AND Prenom = 'Grace'
```

3.

```
SELECT NumChambre  
FROM Reservations  
WHERE date(DateArr) <= date('2024-12-28') AND date(DateDep) >  
date('2024-12-28')
```

4.

a.

```
UPDATE Chambres  
SET Prix = 75  
WHERE NumChambre = 404
```

b.

```
SELECT Reservations.NumChambre  
FROM Reservations  
JOIN Clients ON Reservations.NumClient = Clients.NumClient  
WHERE Prenom = 'Edgar' AND Nom = 'Codd'
```

Exercice 3

1.

- a. un octet correspond à 8 bits
- b. il est possible de coder 2^8 valeurs soit 256 valeurs
- c. $0 \leq \text{valeur} \leq 255$

2.

- a. 01000001
- b. $0 \times 2^0 + 1 \times 2^1 + 0 \times 2^2 + 1 \times 2^3 + 1 \times 2^4 + 1 \times 2^5 + 0 \times 2^6 + 0 \times 2^7 = 58$
- c. 11000110
- d.

$$\begin{array}{r} 1 \\ 01000001 \\ + 11000110 \\ \hline 100000111 \end{array}$$

Les valeurs étant codés sur un octet on "abandonne" le 9e bit, le résultat est donc 00000111.

3.

a.

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

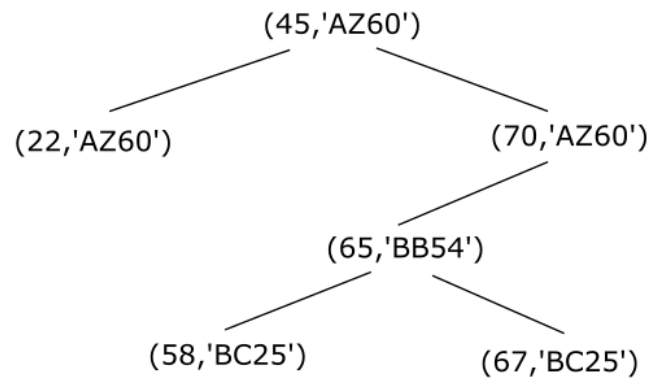
b.

```
mv ./pierre/bizarre ./pierre/videos
```

Exercice 4

1.

a.



b.

Pour obtenir la liste triée, il faut effectuer un parcours en profondeur infixe.

2.

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

3.

a. Cette fonction permet de rechercher un billet de numéro n dans un arbre a . Si le billet recherché est présent dans l'arbre, la fonction renvoie Vrai, dans le cas contraire, la fonction renvoie Faux.

b.

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

Exercice 5

1.

```
def autre(x):  
    if x == 0:  
        return 1  
    if x == 1:  
        return 0
```

2.

a.

```
def nbValeurs(li, v):  
    nb_colonne = 10  
    cpt = 0  
    for i in range(nb_colonne):  
        if grille[li][i] == v:  
            cpt = cpt + 1  
    return cpt
```

b.

```
def regle1(li):  
    nb_colonne = 10  
    v = -1  
    if nbValeurs(li, 0) == 5:  
        v = 1  
    if nbValeurs(li, 1) == 5:  
        v = 0  
    if v != -1:  
        for i in range(nb_colonne):  
            if grille[li][i] == -1:  
                grille[li][i] = v
```

3.

```
def regle3(li):  
    for col in range(8):  
        if grille[li][col] == grille[li][col+2] and grille[li][col+1] != -1:  
            grille[li][col+1] = autre(grille[li][col])
```

4.

```
def convert(L):  
    numbit = len(L)-1  
    s = 0  
    for n in L:  
        s = s + n*2**(numbit)  
        numbit = numbit - 1  
    return s
```

5.

```
fonction doublon(L) :  
  pour tous les éléments v1 de L :  
    compteur ← 0  
    pour tous les éléments v2 de L :  
      si v1 == v2 :  
        compteur ← compteur + 1  
    si compteur > 1:  
      renvoyer Vrai  
renvoyer Faux
```