Proposition de correction

Exercice 1

Q1.a

- Utilisateur : gestion
- ordinateur : capNSI-ordinateur_central

Q1.b

ls -d Contrats

Q2.a

mkdir ~/Contrats/TURING_Alan

Q2.b

chmod 774 ~/Contrats/TURING_Alan

Q3

```
def formatage(tab : list) -> list:
    clients = []
    for nom, prenom in tab:
        clients.append(nom+"_"+prenom)
    return clients
```

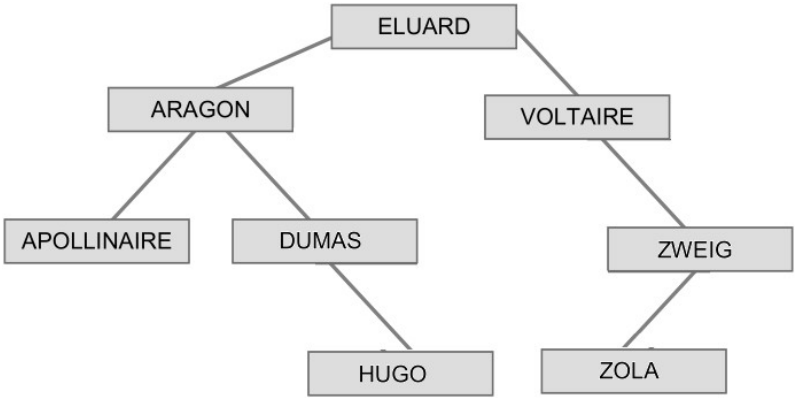
Q4

```
import os

def creation_dossiers(tab):
    """ à lancer dans le home directory """
    for client in tab:
        os.mkdir("Contrats/"+client)
        os.chmod("Contrats/"+client, 774)
```

Exercice 2

Q1.a



Q1.b

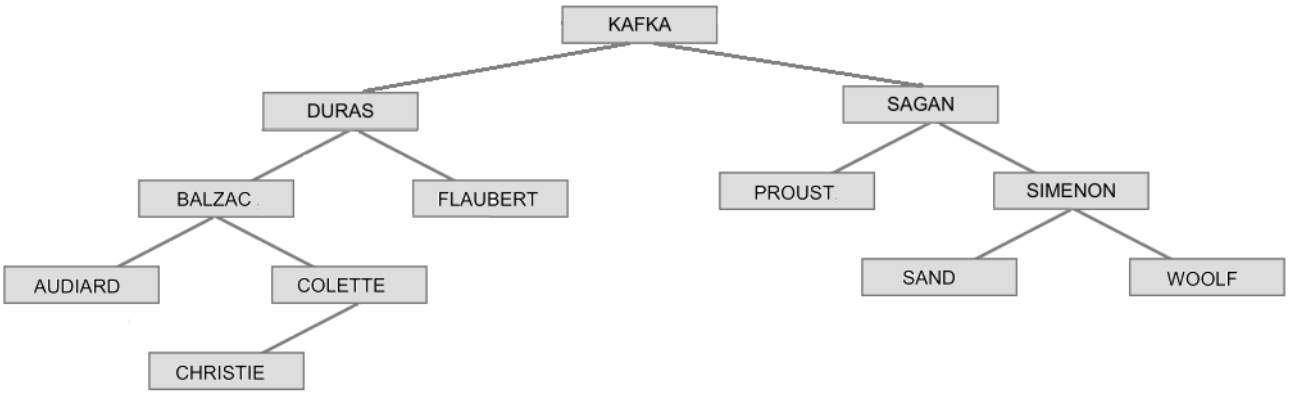
taille : 8

hauteur : 4, convention h(racine) = 1

Q1.c

2^h -1

Q2



Q3

VRAI

La fonction mystere() parcourt l'arbre ABR récursivement pour y chercher la valeur t.

Q4

```
fonction hauteur(ABR) : entier
début
```

```
    si ( ABR = Null ) alors
            renvoyer 0
    sinon
            renvoyer 1 + MAX(hauteur(fils_gauche(ABR)), hauteur(fils_droit(ABR)))
fin
```

Exercice 3

Q1.a

Choix 2 : il faut créer un nouvel objet à chaque itération (sinon pb d'effet de bord)

Q1.b

```
jeu[5][2] = 1
```

Q2.a

```
import random

def remplissage(n : int, jeu: list):
    """ 0 < n <= taille(jeu) """
    cellule = 0
    taille = len(jeu) * len(jeu[0])
    while cellule < (n % taille):          # pré condition pour sortie de boucle
        if jeu[random.randint(0,7)][random.randint(0,7)] == 0:
            jeu[random.randint(0,7)][random.randint(0,7)] = 1
            cellule += 1
```

Q2.b

0 < n ≤ taille(jeu)

Q3

```
def nombre_de_vivants(i : int, j : int, jeu : list) -> int:
    nb = 0
    voisins = [(i-1,j-1), (i-1,j), (i-1,j+1), (i,j+1), (i+1,j+1), (i+1,j), (i+1,j-1), (i,j-1)]
    for e in voisins :
        if 0 <= e[0] < 8 and 0 <= e[1] < 8 :
            nb = nb + jeu[e[0]][e[1]]
    return nb
```

Q4

```
def transfo_cellule(i : int, j : int, jeu : list):
```

```
    voisins = nombre_de_vivants(i, j, jeu)
    if jeu[i][j] == 0:
        return int(voisins == 3)
    else:
        return int(2 <= voisins <=3)
```

Exercice 4

Q1.a

id_match

Q1.b

id_creneau, id_terrain, id_joueur1 et id_joueur2

Q2.a

Le 2020-08-01 de 10h-11h

Q2.b

Dupont Alice et Durand Belina

Q3.a

SELECT prenom_joueur

FROM joueurs

WHERE nom_joueur = "Dupont"

ORDER BY prenom_joueur

Q3.b

UPDATE joueurs

SET mdp = 1976

WHERE id_joueur = 4

Q4

INSERT INTO joueurs

VALUES(5, "MAGID", "Zora", "zora", 2021)

Q5

SELECT DISTINCT matchs.date

```
FROM matchs, joueurs
WHERE joueurs.prenom_joueur = "Alice"
AND matchs.id_joueur1 = joueurs.id_joueur OR  matchs.id_joueur2 = joueurs.id_joueur
ORDER BY matchs.date
```

Exercice 5

Q1

```
def somme(n) :
    total = 0
    for i in range(1, n+1) :
        total = total + 1/i
    return total
```

Q2.a

```
while indice < len(L) :
```

Q2.b

```
maximum = L[0] if len(L) else None
```

traite également le cas du tableau vide…

Q3

Comme indiqué par le message d'erreur, le langage ne peut appliquer l'opérateur de concaténation entre une variable de type string et de type integer

```
L.append('Joueur '+str(i))
```

Q4.a

21

Q4.b

la condition d'arrêt de l'appel récursif ne sera pas vérifié car n ne sera jamais égal à 0 (dépassement de pile d'exécution)

Q5.a

(5, [10])

Q5.b

4 [10]