

Éléments de correction sujet 9 (2024)

Exercice 1

1. Aucun site ne pointe vers site2, sa liste de prédécesseur est donc vide
2.

```
s4.predecesseurs = [(s1, 1), (s2,2)]  
s5.predecesseurs = [(s1, 2), (s3,3), (s4,6)]
```
3.

```
s2.successeurs[1][1]
```

 a pour valeur 5 : le premier [1] correspond à (s3,5) et le 2e [1] correspond à 5 dans (s3,5).
4.
La valeur de popularité du site1 est 6 (4+2)
5.

```
def calculPopularite(self):  
    self.popularite = 0  
    for s in self.predecesseurs :  
        self.popularite = self.popularite + s[1]  
    return self.popularite
```
6.
Il s'agit d'une file (premier arrivé, premier sortie)
7.
Avec une file, on effectue un parcours en largeur
8.
On obtient une liste d'objet de type Site : [s1, s3, s4, s5]
9.

```
def lePlusPopulaire(listeSites):  
    maxPopularite = 0  
    siteLePlusPopulaire = listeSites[0]  
    for site in listeSites:  
        if site.popularite > maxPopularite:  
            maxPopularite = site.popularite  
            siteLePlusPopulaire = site  
    return siteLePlusPopulaire
```
10.
La fonction renvoie : 'site3' (le site3 est le plus populaire avec 12)
11.
Le code n'est pas adapté, car pour un grand nombre de sites, le temps d'exécution serait beaucoup trop long.

Exercice 2

Erreur énoncé : nom_croisiere n'est pas de type INT

Partie A

1. un SGBD permet de gérer la redondance des données (gestion des pannes) et la sécurisation des accès.
2.
route : B -> E -> A
3. routes :
C -> I -> G -> F -> D -> A ou C -> I -> H -> F -> D -> A avec un coût de 2,3

Partie B

4. Parce que l'attribut id_client est unique pour chaque entrée de la table clients.
5.
Une clé étrangère est un attribut qui permet d'effectuer la jointure entre 2 tables. La clé étrangère d'une table A pointe vers la clé primaire d'une table B ce qui permet d'établir une jointure entre A et B.
6.
Les attributs escale_1, escale_2... de la table croisiere sont des clés étrangères qui doivent pointer vers des entrées de la table villes. Les villes 'Puerto sebo', 'Puerto kifecho', ... n'existent pas dans la table villes, d'où l'erreur. Il est donc nécessaire de compléter la table villes (avec les villes 'Puerto sebo', 'Puerto kifecho', ...) avant de renseigner la table croisiere.
7.
Erreur énoncé : il s'agit d'id_client et pas d'id
La première requête permet d'obtenir l'identifiant de M Jean Barc
La deuxième requête permet d'obtenir l'identifiant de la réservation du client ayant pour identifiant 1243 (c'est à dire M Jean Barc)
8.

```
SELECT id_reservation
FROM reservations
JOIN clients ON reservations.id_client = clients.id_client
WHERE nom = 'Barc' AND prenom = 'Jean' AND date_naissance =
'1972/06/29' AND pays = 'Allemagne'
```
9.

```
UPDATE reservation
SET nom_croisiere = 'Croisière Puerto'
WHERE id_client = 20456
```
10.

```
SELECT nom, prenom, date_naissance
FROM clients
JOIN reservations ON clients.id_client = reservations.id_client
WHERE nom_croisiere = 'Croisiere Piano'
OR nom_croisiere = 'Croisiere Puerto'
```

Exercice 3

Partie A

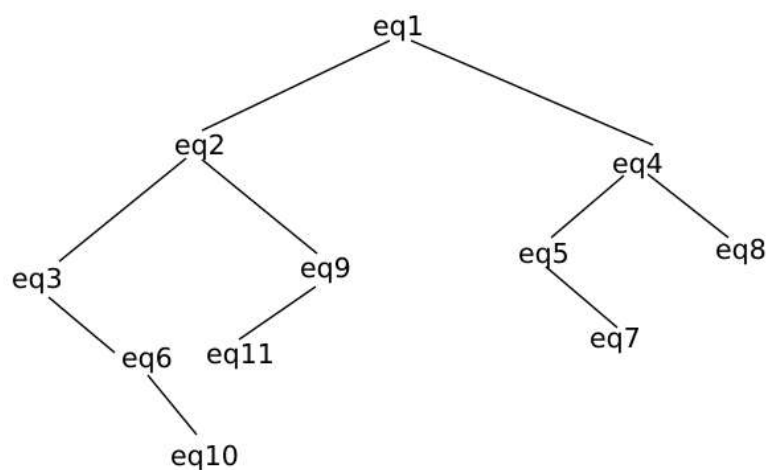
1.
`chien40 = Chien(40, 'Duke', 'wheel dog', 10)`
2.
`def changer_role(self, nouveau_role):`
 `self.role = nouveau_role`
3.
`chien40.changer_role('leader')`

Partie B

4.
`def retirer_chien(self, numero):`
 `new_liste = []`
 `for c in self.liste_chiens:`
 `if c.id_chien != numero:`
 `new_liste.append(c)`
 `self.liste_chiens = new_liste`
5.
`eq11.retirer_chien(46)`
6.
la fonction renvoie le résultat de l'addition $4 + 36/60$ soit 4.6
7.
`def temps_course(equipe):`
 `total = 0`
 `for t in equipe.liste_temps :`
 `total = total + convert(t)`
 `return total`

Partie C

- 8.



9.

Il s'agit d'un parcours infixe.

10.

La fonction est une fonction récursive car elle s'appelle elle-même (ligne 8)

11.

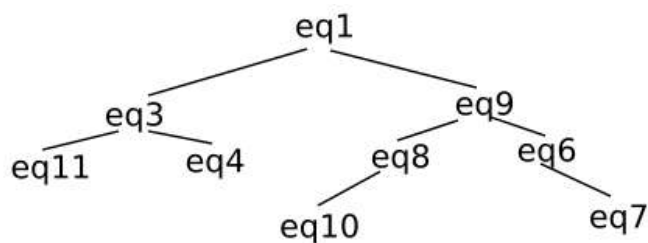
```
def inserer(arb, eq):
    if convert(eq.temps_etape) < convert(arb.racine.temps_etape):
        if arb.gauche is None:
            arb.gauche = Noeud(eq)
        else :
            inserer(arb.gauche, eq)
    else :
        if arb.droit is None:
            arb.droit = Noeud(eq)
        else :
            inserer(arb.droit, eq)
```

12.

```
def est_gagnante(arbre):
    if arbre.gauche == None:
        return arbre.racine.nom_equipe
    else:
        return est_gagnante(arbre.gauche)
```

Partie D

13.



14.

```
def rechercher(arbre, equipe):
    if arbre == None:
        return False
    if arbre.racine == equipe:
        return True
    if convert(equipe.temps_etape) < convert(arbre.racine.temps_etape):
        return rechercher(arbre.gauche, equipe)
    else :
        return rechercher(arbre.droit, equipe)
```