Éléments de correction sujet 06 (2022)

Exercice 1

1.
   a. On obtient lundi avec `jours[1]`
   b. `jours[18%7]` correspond à `jours[4]` c'est-à-dire jeudi
2.
   `numero_jour =(jours.index(j) + n )% 7`
   ATTENTION, il y a une erreur dans l'énoncé, on doit écrire *jours.index(...)* et pas *jours.index[...]*
3.
   a. Pour obtenir le nombre de jours au mois de mars, on doit écrire :
      `mois[3][1]`
   b.
      `mois[(numero_mois + x) % 12][0]`
4.
   a. mois[date[2]][1] renvoie 31
   b.

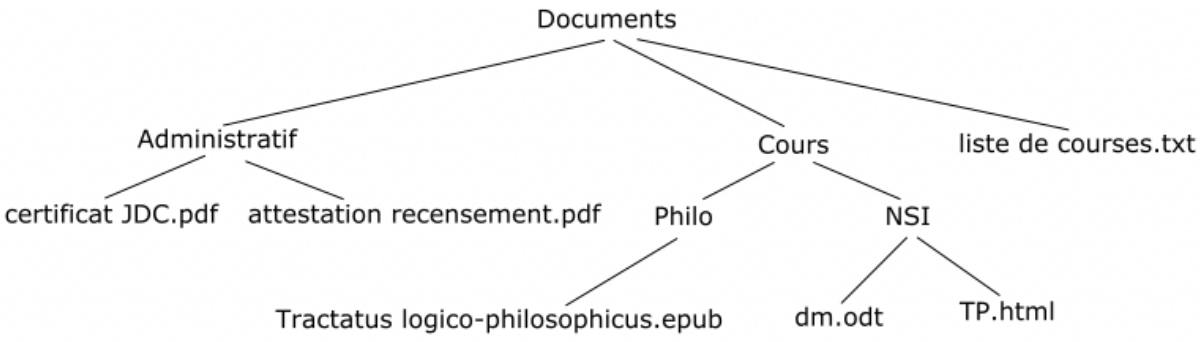
```
def jour_suivant(date):
    nom_j = date[0]
    j = date[1]
    m = date[2]
    a = date[3]
    nom_jour_plus = jours[(jours.index(nom_j) + 1 )% 7]
    nbe_jour_mois = mois[date[2]][1]
    jour_plus = (j + 1) % nbe_jour_mois
    if nbe_jour_mois == j:
        mois_plus = (m + 1)%12
    else :
        mois_plus = m
    if m == 12 and j == 31:
        annee_plus = a + 1
    else :
        annee_plus = a
    return (nom_jour_plus, jour_plus, mois_plus, annee_plus)
```

Exercice 2

1.
```
panier1.enfiler((31002, "café noir", 1.50, 50525))
```
2.
```
def remplir(self, panier_temp):
        while not panier_temp.est_vide() :
            article = panier_temp.defiler()
            self.enfiler(article)
```
3.
```
def prix_total(self):
        p_temp = Panier()
        montant = 0
        while not self.est_vide() :
            article = self.defiler()
            montant = montant + article[2]
            p_temp.enfiler(article)
        while not p_temp.est_vide() :
            article = p_temp.defiler()
            self.enfiler(article)
        return montant
```
4.
```
def horaire_scan(self):
        if self.est_vide():
            return 0
        premier_article = self.defiler()[3]
        dernier_article = premier_article
        while not self.est_vide() :
            dernier_article = self.defiler()[3]
        return dernier_article - premier_article
```

Exercice 3

1.



2.

a.

```
def Parcourir(racine, adr):
    dossier = racine
    for nom_dossier in adr:
        dossier = dossier[nom_dossier]
    return dossier
```

b.

L'instruction `Afficher(Documents, ["Cours", "NSI"], "TP.html")` permet d'afficher la taille en Ko du fichier *TP.html*, c'est-à-dire 60

3.

a.

L'erreur est à la ligne 3, il faut écrire `dossier[nom_fichier] = taille` à la place de `taille = dossier[nom_fichier]`

b.

```
def Ajouter_dossier(racine, adr, nom_dossier):
    dossier = Parcourir(racine, adr)
    dossier[nom_dossier]={}
```

4.

```
def taille(dossier):
    s = 0
    for k in dossier.keys():
        s = s + dossier[k]
    return s
```

Exercice 4

1.
    a. L'attribut *id_mesure* peut jouer le rôle de clé primaire. En effet, chaque valeur de cet attribut est unique (ce qui n'est pas le cas des autres attributs de cette relation, par exemple, il est possible d'avoir plusieurs fois la même température).
    b. Les attributs *Mesures.id_centre* et *Centres.id_centre* vont permettre de faire une jointure entre la relation *Centres* et la relation *Mesures* (*Mesures.id_centre* est une clé étrangère)
2.
    a. Cette requête permet d'afficher tous les centres qui ont une altitude supérieure à 500 m. Plus exactement, on affichera l'id_centre, le nom, la latitude, la longitude et l'altitude de tous les centres qui ont une altitude supérieure à 500 m.
    b.

```
SELECT nom_ville
FROM Centres
WHERE altitude > 700 AND altitude < 1200
```

    c.

```
SELECT longitude, nom_ville
FROM Centres
WHERE longitude > 5 ORDER BY nom_ville
```

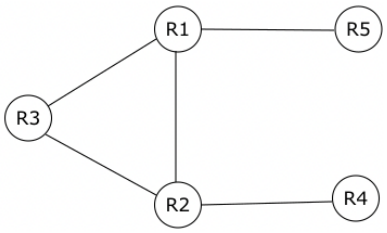
3.
    a. Cette requête permet d'afficher l'ensemble des données (id_mesure, id_centre, date, température, pression et pluviométrie) des mesures effectuées le 30 octobre 2021.
    b.

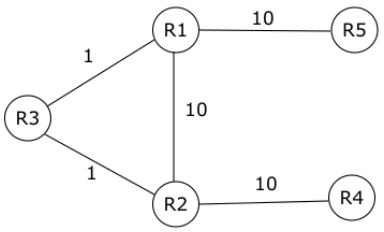
```
INSERT INTO Mesures
VALUES (3650, 138, 2021-11-8, 11, 1013, 0)
```

4.
    a. Cette requête permet d'afficher l'ensemble des données (id_centre, nom de la ville, latitude, longitude et altitude) du centre qui a la latitude minimum (le centre situé le plus au sud).
    b.

```
SELECT DISTINCT nom_ville
FROM Centres
JOIN Mesures ON Mesures.id_centre = Centres.id_centre
WHERE temperature < 10 AND date > 2021-09-30 AND date < 2021-11-1
```

Exercice 5

1.
   a. Dans un SoC (System on Chip) les différents composants (CPU, RAM, GPU, circuit Wifi…) sont tous intégrés sur la même puce (le SoC). Dans un ordinateur “classique” tous ces composants sont séparés.
   b. Seul le Broadcom BCM271 possède une interface réseau de type Ethernet, donc, seul ce SoC peut être connecté à un réseau filaire.
   c. On peut comparer le nombre de cœurs (4 contre 1) et la fréquence de base (1,5 GHz contre 700 MHz). Le Broadcom BCM271 est donc plus “puissant” que le Broadcom BCM2835.
2.
   a. L'adresse MAC est liée au matériel, chaque carte réseau (Ethernet ou Wifi) possède sa propre adresse MAC, il n'existe pas dans le monde, 2 cartes réseau (Ethernet ou Wifi) qui possèdent la même adresse MAC. On retrouve l’adresse MAC au niveau de la couche *accès réseau* du modèle TCP/IP, elle permet d’identifier les machines sur un réseau (au niveau de la couche *accès réseau*).
   b. "10.0.2.15" correspond à une adresse IP (qui permet d’identifier les machines au niveau de la couche IP du modèle TCP/IP).
   c. L’adresse 10.0.2.2 correspond à une passerelle (gateway)
3.
   a. 

   b. R4 → R2 → R1 → R5
4. 

   Le chemin doit minimiser le coût total, on a donc la route :
   R4 → R2 → R3 → R1 → R5 (coût = 22)