Sujet 1 - Centres étrangers - 11 mai 2022 - Éléments de correction

Exercice 1 ***

```
jours=["dimanche", "lundi", "mardi", "mercredi", "jeudi",
       "vendredi", "samedi"]

mois={1 :("janvier",31) , 2 :("février",28) , 3 :("mars",31),
      4 :("avril",30) , 5 :("mai",31) , 6 :("juin",30) ,
      7 :("juillet",31) , 8 :("aout",31) , 9 :("septembre",30),
      10 :("octobre",31) , 11 :("novembre",30) , 12 :("décembre",31)}
```

1. a.

jours[1]

b.

18 % 7 renvoie 4 car 18 = 2 x 7 + 4 reste = 4
donc >>> jours[18 % 7] renvoie "jeudi"

2.

numero_jour = (jours.index(j)+n) % 7

Attention, il y a une erreur dans le sujet : ce sont des parenthèses (pas des crochets).

3. a.

```
>>> mois[3][1]
```

b.

```
>>> numero_mois = 4
>>> x = 5
>>> mois[(numero_mois + x) % 12][0]
```

4. a.

```
>>> date = ("samedi", 21, 10, 1995)
>>> mois[date[2]][1]
31
```

b. **(difficile !)**

```
def jour_suivant(date):
    jour = date[0]
    num_next_jour = (jours.index(jour) + 1) % 7
    nom_jour_suivant = jours[num_next_jour]
    num_jour = date[1] # 1 ... 28 ou 30 ou 31 suivant le mois
    num_mois = date[2] # 1 ... 12
    nb_total_jours_du_mois = mois[num_mois][1]
    num_jour_suivant = (num_jour + 1) % nb_total_jours_du_mois
    num_mois_suivant = (num_mois + ((num_jour + 1) // nb_total_jours_du_mois)) % 12
    num_annee = date[3]
    num_annee_suivante = num_annee + (num_mois + ((num_jour + 1) // nb_total_jours_du_mois)) // 12
    return (nom_jour_suivant, num_jour_suivant, num_mois_suivant, num_annee_suivante)

print(jour_suivant( ("samedi", 21, 10, 1995) ))
print(jour_suivant( ("mardi", 31, 10, 1995) ))
print(jour_suivant(("vendredi", 31, 12, 2021)))
print(jour_suivant(("dimanche", 28, 2, 2022)))
```

```
('dimanche', 22, 10, 1995)
('mercredi', 1, 11, 1995)
('samedi', 1, 1, 2022)
('lundi', 1, 3, 2022)
```

Exercice 2

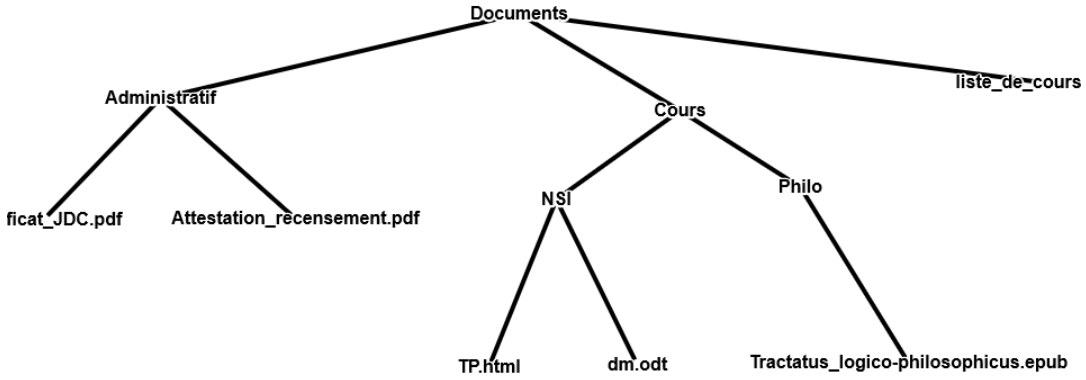
```
class Panier():
    def __init__(self):
        self.contenu = []

    def est_vide(self):
        return self.contenu == []

    def enfiler(self, e):
        self.contenu.append(e)

    def defiler(self):
        return self.contenu.pop(0)

    def remplir(self, panier_temp):
        while not panier_temp.est_vide():
            article = panier_temp.defiler()
            self.enfiler(article)

    def prix_total(self):
        total = 0
        for article in self.contenu:
            total += article[2]
        return total

    def duree_courses(self):
        if self.est_vide():
            return 0
        article_debut = self.defiler()
        while not self.est_vide():
            article_fin = self.defiler()
        duree = article_fin[3] - article_debut[3]
        return duree

if __name__ == '__main__':
    panier1 = Panier()
    print(panier1.est_vide())
    panier1.enfiler((31002, "café noir", 1.50, 50525))
    print(panier1.est_vide())
    print(panier1.contenu)
    panier_temp = Panier()
    panier_temp.enfiler((31004, "limonade", 1.70, 50600))
    panier_temp.enfiler((31010, "bierre", 2.30, 50620))
    print(panier_temp.contenu)
    panier1.remplir(panier_temp)
    print(panier1.contenu)
    print(panier1.prix_total())
    print(panier1.duree_courses())
```

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):
        total = 0
        for article in self.contenu:
            total += article[2]
        return total
```
4.
```
    def duree_courses(self):
        if self.est_vide():
            return 0
        article_debut = self.defiler()
        while not self.est_vide():
            article_fin = self.defiler()
        duree = article_fin[3] - article_debut[3]
        return duree
```

Exercice 3 *

1.

http://graphonline.ru/fr/home?graph=gsjTSpgjunzjyZZoZZcst



2. a.

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

```
Documents = { "Administratif": {
    "certificat JDC.pdf ": 1500,
    "attestation recensement.pdf ": 850},
              "Cours":  {
    "NSI": {
        "TP.html": 60,
        "dm.odt": 345 },
    "Philo": {
         "Tractatus logico-philosophicus.epub": 2600 }},
              "liste de courses.txt ": 24 }
```

b.

```
Afficher(Documents, ["Cours", "NSI"], "TP.html") # renvoie 60
```

3. a.

```
"""
def Ajouter(racine, adr, nom_fichier, taille):
    dossier = Parcourir(racine, adr)
    taille = dossier[nom_fichier] <<< ERREUR ICI !
"""
def Ajouter(racine, adr, nom_fichier, taille):
    dossier = Parcourir(racine, adr)
    dossier[nom_fichier] = taille
```

b.

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

4.

```
def taille(dossier):
    T = 0
    for taille_fic in dossier.values():
        T = T + taille_fic
    return T
```

Exercice 4

```
Centres(id_centre: INT, nom_ville: VARCHAR, latitude: FLOAT, longitude: FLOAT, altitude: FLOAT)
Mesures(id_mesure: INT, id_centre: INT, date: DATE, temperature: FLOAT, pression: INT, pluviometrie: FLOAT)
```

Relation Centres

id_centre	nom_ville	latitude	longitude	altitude
213	Amiens	49.894	2.293	60
138	Grenoble	45.185	5.723	550
263	Brest	48.388	-4.49	52
185	Tignes	45.469	6.909	2594
459	Nice	43.706	7.262	260
126	Le Puy-en-Velay	45.042	3.888	744
317	Gérardmer	48.073	6.879	855

Relation Mesures

id_mesure	id_centre	date	temperature	pression	pluviometrie
1566	138	2021-10-29	8.0	1015	3
1568	213	2021-10-29	15.1	1011	0
2174	126	2021-10-30	18.2	1023	0
2200	185	2021-10-30	5.6	989	20
2232	459	2021-10-31	25.0	1035	0
2514	213	2021-10-31	17.4	1020	0
2563	126	2021-11-01	10.1	1005	15
2592	459	2021-11-01	23.3	1028	2
3425	317	2021-11-02	9.0	1012	13
3430	138	2021-11-02	7.5	996	16
3611	263	2021-11-03	13.9	1005	8
3625	126	2021-11-03	10.8	1008	8

1. a.

L'attribut **`id_mesure`** de la relation **`Mesures`** peut jouer le rôle de **clé primaire**. En effet, cet attribut permettra d'identifier de manière unique un enregistrement.

b.

Entre la relation **`Centres`** et la relation **`Mesures`**, il est possible de faire une jointure à l'aide de l'attribut **`id_centre`** qui est une clé primaire de la relation **`Centres`** et une clé étrangère de la relation **`Mesures`**.

2. a. La requête suivante…

```
SELECT * FROM Centres WHERE altitude > 500;
```

… permet d'afficher tous les attributs (l'identifiant du centre, le nom de la ville, la latitude, la longitude et l'altitude) des centres météorologiques de la table Centres dont l'altitude dépasse les 500 mètres.

On obtient donc l'affichage suivant :

138	Grenoble	45.185	5.723	550
185	Tignes	45.469	6.909	2594
126	Le Puy-en-Velay	45.042	3.888	744
317	Gérardmer	48.073	6.879	855

b.

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

~~`SELECT nom_ville FROM Centres WHERE 700 <= altitude <= 1200 ;`~~ `<<< NON !`

Le Puy-en-Velay
Gérardmer

c.

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

6.879	Gérardmer
5.723	Grenoble
7.262	Nice
6.909	Tignes

3. a.

La requête `SELECT * FROM Mesures WHERE date="2021-10-30";` permet d'obtenir tous les attributs des mesures (enregistrements) de la table `Mesures` qui ont été effectuées le 30 octobre 2021.

On obtient donc l'affichage suivant :

2174	126	2021-10-30	18.2	1023	0
2200	185	2021-10-30	5.6	989	20

b. Il s'agit d'un nouvel enregistrement dans la table Mesures dont le schéma est (rappel) :

`Mesures(id_mesure: INT, id_centre: INT, date: DATE, temperature: FLOAT, pression: INT, pluviometrie: FLOAT)`

```
INSERT INTO Mesures
VALUES (3650, 138, "2021-11-08", 11, 1013, 0) ;
```

id_centre	nom_ville	latitude	longitude	altitude
459	Nice	43.706	7.262	260.0

4. a.

```
SELECT * FROM Centres WHERE latitude = (SELECT MIN(latitude) FROM Centres);
```

Cette requête va permettre d'afficher tous les attributs du (des) centre(s) météorologique(s) dont la latitude est la plus petite. On obtient donc l'affichage suivant :

459	Nice	43.706	7.262	260.0

b.

```
SELECT DISTINCT Centres.nom_ville
FROM Centres, Mesures
WHERE Centres.id_centre = Mesures.id_centre
AND Mesures.date LIKE "2021-10-%"
AND Mesures.temperature < 10 ;
```

nom_ville
Grenoble
Tignes

```
SELECT DISTINCT Centres.nom_ville
FROM Centres
JOIN Mesures ON Centres.id_centre = Mesures.id_centre
WHERE Mesures.date LIKE "2021-10-%"
AND Mesures.temperature < 10 ;
```

nom_ville
Grenoble
Tignes

```
SELECT DISTINCT Centres.nom_ville
FROM Centres, Mesures
WHERE Centres.id_centre = Mesures.id_centre
AND Mesures.date >= "2021-10-01" AND Mesures.date <= "2021-10-31"
AND Mesures.temperature < 10 ;
```

nom_ville
Grenoble
Tignes

Exercice 5

1. a. Expliquer ce qui différencie un SOC d'un nano ordinateur d'un microprocesseur classique ?

A la différence d'un microprocesseur classique, le SoC d'un nano ordinateur contient en plus un GPU (processeur graphique), des périphériques d'interface, de la mémoire RAM.

b. Lequel de ces SOC peut être connecté à un réseau filaire. Justifier la réponse.

	SOC de 2 nano ordinateurs	
Processeur	Broadcom BCM271	Broadcom BCM2835
Architecture	ARMv8-A (64-bit)	ARMv6Z (32-bit)
Microarchitecture	Cortex-A72	ARM11
Famille du processeur0	BCM	BCM
Cœur	4	1
Fréquence de base	1,5 GHz	700 MHz
Fréquence turbo	-	1,0 GHz
Mémoire cache	1 MB	128 KB
Capacité mémoire maxi	8 GB	512 MB
Types de mémoire	LPDDR4-3200 SDRAM	SDRAM
GPU (processeur graphique) intégré	Broadcom VideoCore VI	Aucun
GPU, unités d'exécution	4	-
GPU, unités shader	64	-
GPU, cadence	500 MHz	-
GPU, flottant FP32	32 GFLOPS	-
Drystone MIPS	22 740 DMIPS	1 190 DMIPS
Résol. affichage max	4K@60fps	1080p@30fps
Décodage vidéo	H.265 4K@60fps, H.264 1080p@60fps	H.264 1080p@30fps
Encodage vidéo	H.264 1080p@30fps	H.264 1080p@30fps
Interface réseau	10/100/1000M Gigabit Ethernet	-
Connectivité	USB 2.0, USB 3.0, HDMI 2.0	USB 2.0, HDMI 1.3
Wifi	2.4GHz/5GHz 802.11 b/g/n/ac	-
Bluetooth	Bluetooth 4.2	-
Audio	I2S	I2S

Le 1e SoC, celui dont le processeur est le Broadcom BCM271 peut être connecté à un réseau filaire car il possède une interface réseau (une carte réseau).

c. Citer 2 caractéristiques permettant de comparer la puissance de calcul de ces deux SOC.

Le 1e Soc est plus puissant que le second. En effet, il possède un processeur plus rapide (fréquence de base meilleure : 1500 MHz contre 700 MHz), de plus le processeur possède 4 cœurs (multitâches), il a une mémoire cache supérieure. Le 1e Soc est plus puissance que l'autre car il contient un GPU, un processeur dédié à l'affichage vidéo.

Afin l'architecture du 1e SoC est de 64 bits (contre seulement 32 bits pour le second).

2.

```
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 10.0.2.15  netmask 255.255.255.0  broadcast 10.0.2.255
        inet6 fe80::761a:3e85:cc97:6491  prefixlen 64  scopeid 0x20<link>
        ether 08:00:27:8b:c3:91  txqueuelen 1000  (Ethernet)
        RX packets 136  bytes 13703 (13.3 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 180  bytes 17472 (17.0 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
        device interrupt 9  base 0xd020
```

a. L'indication `ether 08:00:27:8b:c3:91` correspond à l'adresse MAC de sa carte réseau (son interface réseau) ; c'est un numéro qui permet d'identifier de manière unique la carte réseau.

b. `inet 10.0.2.15`

Cette indication représente l'adresse IP qui lui a été attribuée de manière dynamique lorsqu'on l'a branché sur le réseau local.

c. `1 _gateway (10.0.2.2)  0.328 ms  0.275 ms  0.267 ms`

L'adre sse 10.0.2.2 correspond à un routeur (passerelle), en l'occurrence, la box de la maison.

3. Protocole **RIP**

Routeur1 (R1)

Destination	Direction	Saut	Débit (Mbits/s)
R2	R2	1	10
R3	R3	1	100
R4	R2	2	
R5	R5	1	10

Routeur2 (R2)

Destination	Direction	Saut	Débit (Mbits/s)
R1	R1	1	10
R3	R3	1	100
R4	R4	1	10
R5	R1	2	

Routeur3 (R3)

Destination	Direction	Saut	Débit (Mbits/s)
R1	R1	1	100
R2	R2	1	100
R4	R2	2	
R5	R1	2	

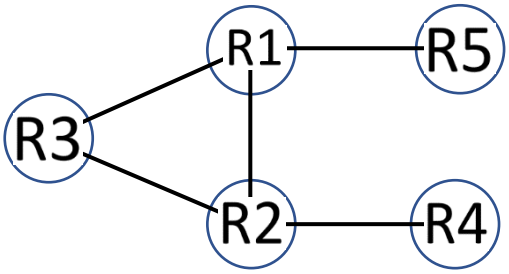
Routeur4 (R4)

Destination	Direction	Saut	Débit (Mbits/s)
R1	R2	2	
R2	R2	1	10
R3	R2	2	
R5	R2	3	

Routeur5(R5)

Destination	Direction	Saut	Débit (Mbits/s)
R1	R1	1	10
R2	R1	2	
R3	R1	2	
R4	R1	3	

a.



b. Pour aller de R4 à R5, il faut suivre la route suivante :

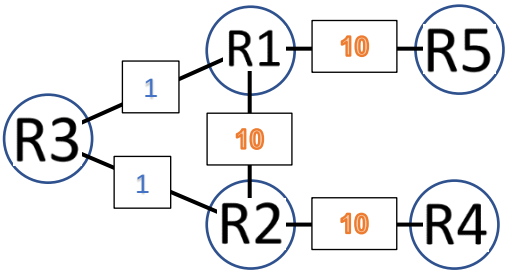
• d'après la table de routage de R4, pour atteindre la destination R5, il faut passer par R2

• d'après la table de routage de R2, pour aller vers R5, il faut passer par R1

• d'après la table de routage de R1, R5 est directement connecté à R1.

Le chemin est donc : **`R4 - R2 - R1 - R5`**

4. Protocole OSPF.



$C = \frac{100}{10} = 10$

$C = \frac{100}{100} = 1$

Avec le protocole OSPF, la route dont le coût total de transmission est le plus petit, pour aller de R4 à R5 est :

`R4 --(10)-- R2 --(1)-- R3 --(1)-- R1 --(10)-- R5 pour un coût total de 22.`

`R4 - R2 - R3 - R1 - R5`