2025 métropole jour 1

Exercice 1

Partie A

1. Le numéro de série n'est pas unique. Deux guitares de marques différentes peuvent avoir le même numéro de série comme la 4 et la 8 de l'extrait.

2.

marque	modele
Gibson	Les Paul Goldtop
Fender	Stratocaster

3.
```sql
SELECT annee FROM inventaire WHERE modele = 'Les Paul Standard';
```

4.

```sql
SELECT modele
FROM inventaire
WHERE marque = 'Gibson'
ORDER BY annee;
```

5.
```sql
UPDATE inventaire SET annee = 1957 WHERE id = 1;
```

Partie B

6. `guitare` doit être crée après `modele` car une clé étrangère lui fait référence. `modele` doit être crée après `marque` car une clé étrangère lui fait référence. Donc l'ordre de création doit être `marque`, `modele` puis `guitare`

7.

```sql
SELECT num_ser, annee
FROM guitare
JOIN modele
ON guitare.id_modele = modele.id
WHERE nom = 'Les Paul Standard';
```

8.
```sql
DELETE FROM guitare WHERE id = 3;
```

9.

```sql
INSERT INTO marque VALUES (3, 'BC Rich')
INSERT INTO modele VALUES (5, 'Mockingbird', 3)
INSERT INTO guitare VALUES (9, 5, 1992, '92R', 5000)
```

10.

```sql
SELECT SUM(prix)
FROM guitare
JOIN modele
ON guitare.id_modele = modele.id
WHERE nom = 'Stratocaster';
```

Exercice 2

1.

```
tache1 = Tache(1, "Répondre aux e-mails", 45)
tache2 = Tache(2, "Ranger ma chambre", 60)
```

2.

```
def avancer(self, n):
    self.duree_restante = self.duree_restante - n
```

3.

```
def est_terminee(self):
    return self.duree_restante <= 0
```

4.

```
[début] (<t3>, 4) (<t7>, 4) (<t1>, 3) (<t2>, 3) (<t6>, 2) (<t4>, 1)(<t5>, 1) [fin]
```

5.

```
<t3>
[début] (<t1>, 3) (<t2>, 3) (<t4>, 1) (<t5>, 1)[fin]
```

6.

```
4
[début] (<t3>, 4) (<t1>, 3) (<t2>, 3) (<t4>, 1) (<t5>, 1)[fin]
```

7.

```
def ajouter_file_prio(f, t, p):
    f_aux = File()
    while not f.est_vide() and f.examiner()[1] >= p:
        f_aux.enfiler(f.defiler())
    f_aux.enfiler((t, p))
    while not f.est_vide():
        f_aux.enfiler(f.defiler())
    while not f_aux.est_vide():
        f.enfiler(f_aux.defiler())
```

8. On defiler deux fois une file de taille m. Le coût est donc linéaire : $O(m)$

9. `3 7 3 3 3 1 2 1 2 2 6 6 6 4 5 4 5`

10.

```
def planning(f):
    tab = []
    while not f.est_vide():
        t, p = f.depiler()
        # On peut aussi ne pas mettre le __repr__()
        tab.append(t.__repr__())
        t.avancer(25)
        if not t.est_terminee():
            ajouter_file_prio(f, t, p)
    return tab
```

Exercice 3

Partie A

1.
Les deux valides sont (a) et (b) car (d) est sur un sous-réseau différent et (c) n'existe pas car 261 > 255.

2. 192.168.20.255

3. Les valeurs vont de 0 à 255, il y en a donc 256. Il faut retrancher l'adresse du réseau (0), l'adresse de broadcast (255), l'adresse du reouteur et les trois autres. Il reste donc 250 possibilités.

4. 8 adresses nécessitent 3 bits. Le masque peut donc utiliser $32 - 3 = 29$ bits.

Partie B

5.

Réseau destination	Interface de sortie	Prochain routeur	Nombre de sauts
192.168.30.0	172.16.4.1	172.16.4.2	1
172.16.1.0	172.16.3.1	172.6.3.2	1

6. Celui qui est attendu semble être `192.168.10.0` :

Réseau destination	Interface de sortie	Prochain routeur	Nombre de sauts
192.168.10.0	172.16.4.1	172.16.4.2	2

Mais il y a aussi le `172.16.0.0` :

Réseau destination	Interface de sortie	Prochain routeur	Nombre de sauts
172.16.0.0	172.16.3.1	172.16.3.2	1

7.

Réseau destination	Interface de sortie	Prochain routeur	Nombre de sauts
autre	172.16.3.1	172.16.3.2	

Partie C

8.
- Fast Ethernet : 10
- Fibre optique : 1

9. Le chemin est 1-2-3-4 avec un coût de 21.

Partie D

10. `'11000000.10101000.00010100.00001100'`

11. Quand les deux adresses sont identiques.

12.

```
def precede(ip_1, ip_2):
    for i in range(35):
        if ip_1[i] < ip_2[i]:
            return True
        elif ip_1[i] > ip_2[i]:
            return False
    return False
```

13.
- attribut : `adresse_ip`
- méthode : `est_vide`

14. `return self.adresse_ip == ''`

15. L'accès aux éléments d'un ABR est rapide, il se fait en $O(log(n))$. Alors que dans un tableau la recherche est linéaire en $O(n)$.

16.

```
def modifie(self, adresse_ip, interface, passerelle, cout):
    if self.est_vide():
        self.gauche = Abr('','','',0)
        self.droite = Abr('','','',0)
    self.adresse_ip = adresse_ip
    self.interface = interface
    self.passerelle = passerelle
    self.cout = cout
```

17.

```
elif precede(ip_bin(adresse_ip), ip_bin(self.adresse_ip)):
```