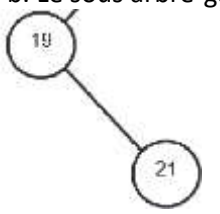
Correction

NSI - 2021 Métropole Jour 1 (--)

Exercice 1 - Arbres binaires de recherche

1. a. Cet arbre contient 4 feuilles qui ont respectivement pour valeur 12, val, 21 et 32.

1. b. Le sous arbre-gauche du nœud 23 est :



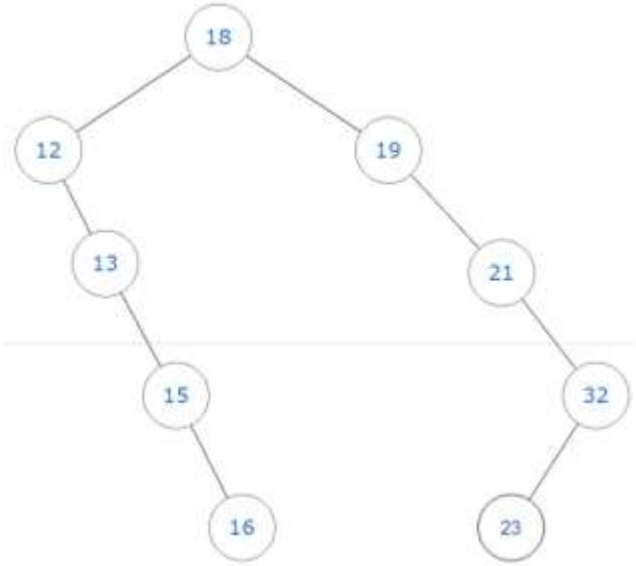
1. c. La hauteur de cet arbre est de 4 et sa taille (nombre de nœud) est de 9.
Remarque : L'énoncé explique qu'un arbre limité à un nœud a une hauteur de 1 mais ici nous devrions parler de niveau plutôt que de hauteur. Cet arbre possède 4 niveaux et a une hauteur de 3.

1. d. Les valeurs possibles pour val sont 16 et 17.

2. a. Parcours infixe : 12, 13, 15, val = 16, 18, 19, 21, 23 et 32.

2. b. Parcours suffixe : 12, 13, val = 16, 15, 21, 19, 32, 23 et 18.

3. a.



3. b. Les instructions sont :

```
racine = Noeud(18)
racine.insere_tout([15, 13, 12, 16, 23, 19, 21, 32])
```

Remarque : Il y a plusieurs possibilités pour l'ordre des valeurs à insérer :
Autre possibilité : `[23, 15, 32, 19, 16, 13, 21, 12]`
Impossible : `[13, 15, 12, 16, 23, 19, 21, 32]` car la valeur 15 doit être insérée avant la valeur 13.

Correction

NSI - 2021 Métropole Jour 1 (--)

3. c. L'ordre d'exécution est : bloc 3 (car 19 > 18), bloc 2 (car 19 < 23) et bloc 1 (car 19 est déjà dans l'arbre).

4.

Solution itérative :

```
def recherche2(self, v):
    n = self
    while n :
        if v == n.v:
            return True
        elif v < n.v:
            n = n.ag
        else:
            n = n.ad
    return False
```

Solution récursive :

```
def recherche(self, v):
    if v == self.v:
        return True
    elif v < self.v and self.ag:
        return self.ag.recherche(v)
    elif self.ad:
        return self.ad.recherche(v)
    return False
```

Correction

NSI - 2021 Métropole Jour 1 (--)

Exercice 2 - Gestion des processus par les systèmes d'exploitation et chiffrement avec XOR

Partie A

1. b

2. c

3. b

4. d

Partie B

1.

P3	P3	P2	P1	P1	P1	P2	P2	P3	P3
0	1	2	3	4	5	6	7	8	9	10

Explications : À l'instant 2, le processus P2 arrive et il est plus prioritaire que P3. À L'instant 3, le processus P1 arrive et est plus prioritaire que P2. Ce processus P1 s'exécute jusqu'à son terme puis c'est le processus P2 et le processus P3 qui s'achèvent.

2. Le scénario 2 provoque un interblocage car :
- le processus P1 occupe la ressource R1 et attend la ressource R2
- le processus P3 occupe la ressource R2 et attend la ressource R1

Partie C

1. a. $\mathrm{m} = 0\mathrm{b}\ 0110\ 0011\ 0100\ 0110 = 0\mathrm{x}\ 63\ 46 \rightarrow \mathrm{cF}$

1. b.
$$\begin{array}{rcl} m & = & 0b\ 0110\ 0011\ 0100\ 0110 \\ k & = & 0b\ 1110\ 1110\ 1111\ 0000 \\ m\ XOR\ k & = & 0b\ 1000\ 1101\ 1011\ 0110 \end{array}$$

2.a

a	b	a XOR b	(a XOR b) XOR b
0	0	0	0
0	1	1	0
1	0	1	1
1	1	0	1

2. b. Bob doit de nouveau appliquer l'opération XOR bit à bit avec la même clé sur le message chiffré.

Correction

NSI - 2021 Métropole Jour 1 (--)

Exercice 3 - Bases de données de trains

1. Un tel logiciel est un SGBDR pour « Système de Gestion des Bases de Données Relationnel ».

2. a. La première requête demande d'effacer dans la table `Train` le train n°1241. Or si dans la table `Reservation` une clé étrangère fait référence à ce train, la contrainte de référence va empêcher sa suppression (sauf si la clé étrangère numT de la table `Reservation` a été définie avec l'option `ON DELETE CASCADE`, dans ce cas, le train n°1241 sera supprimé ainsi que toutes les réservations associées et la deuxième requête ne sert à rien).

Remarques : Si l'ordre de ces deux requêtes est inversé, il n'y a pas d'erreur.

2. b. Si une réservation fait référence à un n° de train qui n'existe pas, la contrainte de référence bloquera l'enregistrement.

3.a.

```
SELECT numT FROM Train WHERE Destination = 'Lyon';
```

ou

```
SELECT numT FROM Train WHERE Destination = 'Lyon' ORDER BY numT;
```

pour obtenir les numéro dans l'ordre croissant mais ce n'est pas demandé.

3. b.

```
INSERT INTO Reservation VALUES (1307, 'Turing', 'Alan', 33, 654);
```

3. c.

```
UPDATE Train SET horaireArrivee = '08:11' WHERE numT = 7869;
```

4. Cette requête permet de dénombrer les réservations faites au nom de Grace Hopper (informaticienne américaine conceptrice du premier compilateur en 1951).

5.

```
SELECT Train.destination, Reservation.prix FROM Train
JOIN Reservation ON Train.numT = Reservation.numT
WHERE Reservation.nomClient = 'Hopper' AND Reservation.prenomClient = 'Grace';
```

Correction

NSI - 2021 Métropole Jour 1 (--)

Exercice 4 - Algorithme du tri fusion

1. a. L'ordre de grandeur du coût, en nombre de comparaisons, de l'algorithme de tri fusion pour une liste de longueur n est $n.log(n)$. Sa complexité est linéarithmique $O(n.log(n))$

1. b. L'algorithme du tri à bulles (ou tri par propagation) ou le tri par sélection ou le tri par insertion qui ont un ordre de grandeur du coût, en nombre de comparaisons de n^2. Leur complexité est quadratique $O(n^2)$.
Ils sont moins efficaces que le tri fusion qui utilise la méthode « diviser pour régner ».

2.
```
[7, 4, 2, 1, 8, 5, 6, 3]
[7, 4, 2, 1]
[7, 4]
[2, 1]
[8, 5, 6, 3]
[8, 5]
[6, 3]
```

3.

Solution 1

```
def moitie_droite(L) :
    tab = []
    for i in range(len(L)//2, len(L)) :
        tab.append(L[i])
    return tab
```

Solution 2

```
def moitie_droite(L) :
    return L[n//2:]
```

4.
```
def fusion(L1, L2):
    L = []
    n1 = len(L1)
    n2 = len(L2)
    i1 = 0
    i2 = 0
    while i1 < n1 or i2 < n2 :
        if i1 >= n1:
            L.append(L2[i2])
            i2 = i2 + 1
        elif i2 >= n2:
            L.append(L1[i1])
            i1 = i1 + 1
        else:
            e1 = L1[i1]
            e2 = L2[i2]
            if e1 > e2 :
                L.append(e2)
                i2 = i2 + 1
            else :
                L.append(e1)
                i1 = i1 + 1
    return L
```

Correction

NSI - 2021 Métropole Jour 1 (--)

Exercice 5 - Réseaux et protocoles de routage

1. a. Grâce à l'adresse IP de la passerelle 86.154.10.1 on en déduit que $R1$ envoie le paquet à $R2$.

1. b. Les routeurs traversés par ce paquet lorsqu'il va du réseau $L1$ au réseau $L2$ sont $R1$, $R2$, $R6$.

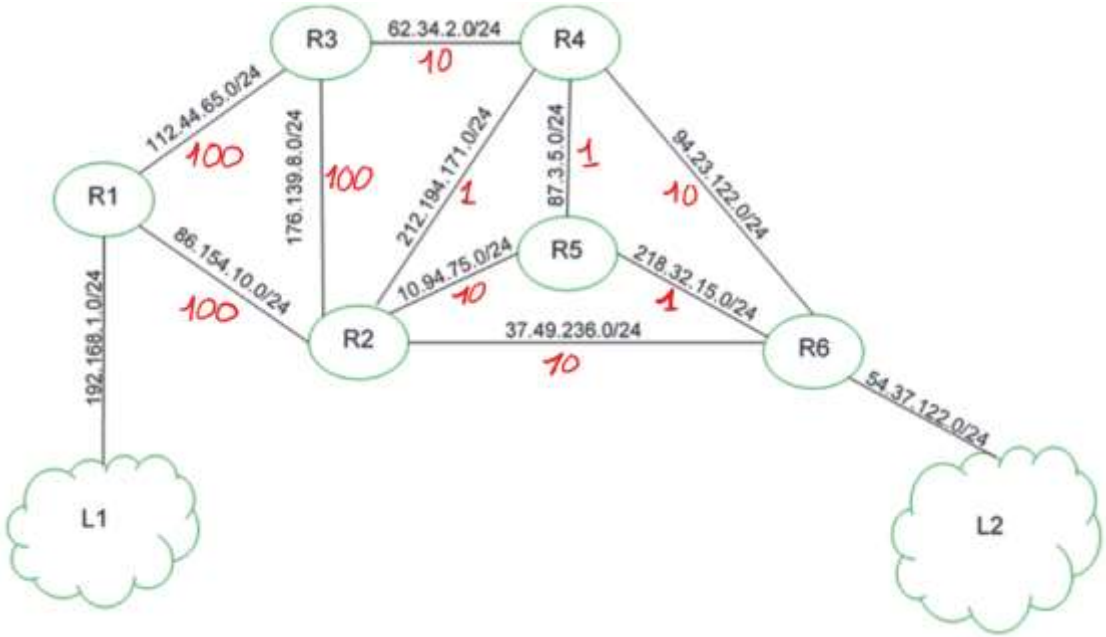
2. a. Un chemin que pourra suivre ce paquet est $R1 \rightarrow R3 \rightarrow R4 \rightarrow R6$
(L'autre chemin possible en utilisant le protocole Rip est $R1 \rightarrow R3 \rightarrow R2 \rightarrow R6$.)

2. b. À la suite de la rupture de la liaison entre les routeurs $R1$ et $R2$, seule la ligne de $R1$ sera modifiée. Sa passerelle deviendra $R3$ en remplacement de $R2$.

3. a. Le coût de la liaison $R2 \leftrightarrow R3$ est :

$$C = \frac{10^9}{BP} = \frac{10^9}{10 \times 10^6} = 100$$

3. b.



Le chemin parcouru en appliquant le protocole OSPF est $R1 \rightarrow R2 \rightarrow R4 \rightarrow R5 \rightarrow R6$ pour un coût total de 103.

3. c. Avec la métrique OSPF, l'extrait de la table de routage est modifié pour le routeur $R2$ dont la passerelle deviendra $R4$ et pour le routeur $R4$ dont la passerelle deviendra $R5$.