Proposition de correction

Exercice 1

Q1.a

Oui : id_mère est un identifiant unique

Q1.b

Oui : le n-uplet donne un identifiant unique

Q1.c

Non : le n-uplet ne peut pas assurer de donner un identifiant unique

Q2

il y a une vérification de contrainte d'intégrité entre la table Naissances et la table Patientes, un enfant a obligatoirement une mère.

Q3

```
INSERT INTO Patientes
VALUES(13862, 'Bélanger', 'Ninette', 'La Rochelle')
```

Q4

```
UPDATE Naissances
SET prenom = 'Laurette'
WHERE idmere = 13860
```

Q5

```
SELECT nom, prenom FROM Patientes
WHERE commune = 'Aigrefeuille d\'Aunis'
ORDER BY nom, prenom
```

Q6

```
SELECT AVG(Naissances.poids)
FROM Naissances, TypesAccouchement
WHERE TypesAccouchement.libelleAcc = 'césarienne' AND Naissances.acc = TypesAccouchement.idAcc
```

Q7

Les nom et prénom des mères qui ont accouché par voie naturelle

Exercice 2

Q1

attente.append((50,4))

Q2.a

tri par insertion

Q2.b

quadratique

Q3.a

```
def quitte(attente) :
    return [(patient, prio) for (patient, prio) in attente if prio == 1]
```

Q3.b

```
def maj(attente : list) -> list :
    return [(patient, prio-1) for (patient, prio) in attente]
```

Q4.a

```
def priorite(attente : list, p : int) -> int :
    return [prio for (patient, prio) in attente if patient == p][0]
```

Q4.b

```
def revise(attente, p) :
    nouvelle = []
    n = priorite(attente, p)
    for (patient, prio) in attente :
        if patient == p :
            nouvelle.append((patient, 1))
        elif prio < n :
            nouvelle.append((patient, prio + 1))
        else :
            nouvelle.append((patient, prio))
    return nouvelle
```

Exercice 3

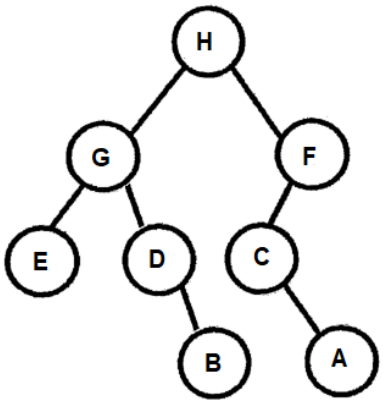
Q1

5

Q2.a

arbre 2

Q2.b



Q3.a

Parcours postfixe : dbfga

Q3.b

```
def parcours_maladies(arb : dict):
    if arb == {}:
            return None
    if arb['sag'] == {} and arb['sad'] = {} :      # parcours préfixe
            print(arb['etiquette'])
    parcours(arb['sag'])
    parcours(arb['sad'])
```

Q4

```
def symptomes(arb, mal):
   if arb['sag'] != {} :
      symptomes(arb['sag'], mal)

   if arb['sad'] != {} :
      symptomes(arb['sad'], mal)
```

```
if arb['sag'] == {} and arb['sad'] == {} :
    arb['surChemin'] = True
    print('symptômes de', arb['etiquette'], ':')
else :
    if arb['sad'] != {} and arb['sad']['surChemin'] :
        print(arb['etiquette'])
        arb['surChemin'] = True

    if arb['sag'] != {} and arb['sag']['surChemin'] :
        print(arb['etiquette'])
        arb['surChemin'] = True
```

Exercice 4

Partie A

Q1

P1	P1	P2	P3	P3	P3	P2	P2	P2	P4	P4	P4	P4	P1	
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14

Q2

Processus	Temps d'exécution	Instant d'arrivée	Temps de séjour	Temps d'attente
P1	3	0	14 – 0 = 14	14 – 3 = 11
P2	4	2	9 – 2 = 7	7 – 4 = 3
P3	3	3	6 – 3 = 3	3 – 3 = 0
P4	4	5	13 – 5 = 8	8 – 4 = 4

Q3

Le processus est en top priorité

Partie B

Q1

Il y a un circuit fermé : D1D4 → D4D5 → D5D1

Q2

interblocage

Q3

Tableur
SBGD
Traitement de texte
Analyseur échantillon

Exercice 5

Partie A

Q1

192.168.1.0
255.255.255.0

Q2

175.89.50.0
44.197.5.0
192.168.5.0

Q3.a

192.168.1.1
192.168.1.254

Q3.b

254

Partie B

Q1

R5-R1-R0

Q2

R5-R4-R2-R0

Partie C

Q1

$C = \frac{B_{ref}}{BP} = \frac{10^9}{400 \times 10^6} = 2{,}5$ arrondi à 3

Q2

$BP = \frac{B_{ref}}{C} = \frac{10^9}{5} = 200\,Mb/s$

Q3

R5-R4-R2-R1-R0

C = 1 + 4 + 1 + 2 = 8

Q4

R5-R4-R2-R3-R0