2021 centre étrangers sujet 1

Exercice 1

1.

```
D
A
```

2.

```
def cryptage(self, texte):
    s = ""
    for ci in texte:
        co = self.decale(ci)
        s = s + co
    return s
```

3.

```
cle = int(input("Quelle est la clé de chiffrement ? "))
code = CodeCesar(cle)
texte = input("Quel est le texte à chiffrer ? ")
print(code.cryptage(texte))
```

4. La méthode `transforme` déchiffre un message chiffré. Pour cela elle remplace temporairement la clé par son opposé. Le code va afficher :

```
FIN
```

Exercice 2

1.a. `{"type" : "classique", "etat" : 1, "station" : "Coliseum"}`

1.b. `0`

1.c. Erreur car la clé n'existe pas.

2.a. "electrique" ou "classique".

2.b. Elle renvoie une station contenant un vélo disponible correspondant au choix.

3.a.

```
for v in flotte:
    if flotte[v]["station"] == "Citadelle" and flotte[v]["etat"] != -1:
        print(v)
```

3.b.

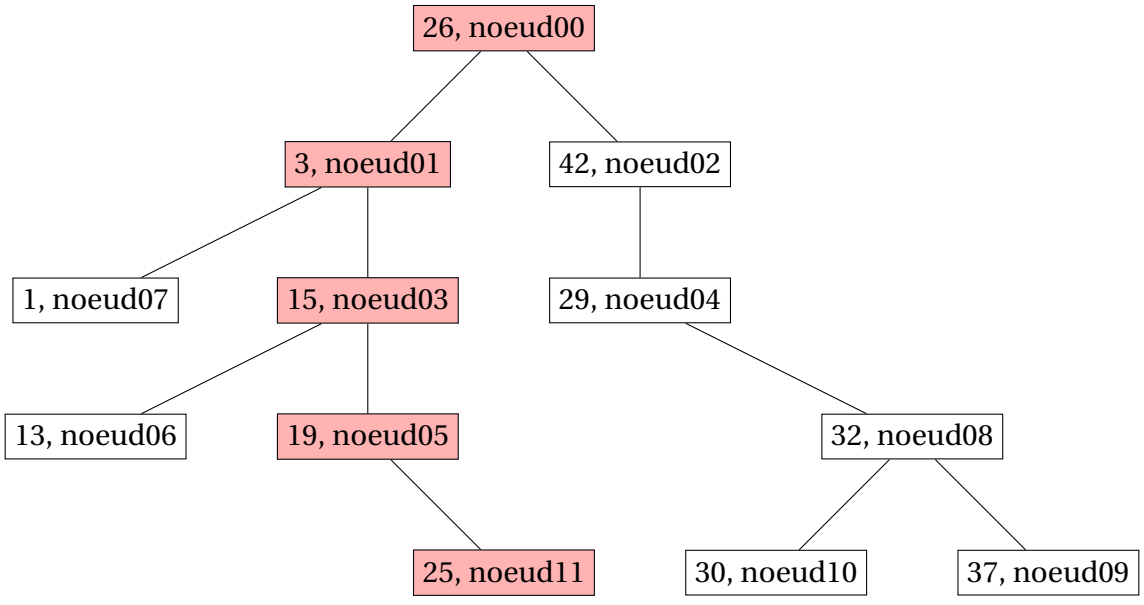
```
for v in flotte:
    if flotte[v]["type"] == "electrique" and flotte[v]["etat"] != -1:
        print(v, flotte[v]["station"])
```

4. On retournera un dictionnaire avec les nome des stations comme clés.

```
def proche(pu):
    dico = {}
    for s, ps in stations.items():
        d = distance(pu, ps)
        if d <= 800: # Si la station est proche
            liste_velos = []
            for v in flotte:
                if flotte[v]["station"] == s and flotte[v]["etat"] == 1:
                    liste_velos.append(v)
            if liste_velos != []: # S'il y a des vélos disponibles
                dico[s] = [d, liste_velos]
    return dico
```

Exercice 3

1.



La valeur 25 sera insérée sous le noeud 05 en fils droit. On commence à la racine, 25 < 26, donc on va à gauche, 25 > 3 donc on va à droite, 25 > 15 donc on va à droite, 25 > 19 donc on va à droite.

2. On peut stocker les valeurs 26, 27 et 28. Elles doivent être supérieures ou égales à 26 et strictement inférieures à 29.

3.a. 26-3-1-15-13-19-(25)-42-29-32-30-37

3.b. Préfixe.

4. Il faut faire un parcours infixe :

```
Parcours2(A)
    Parcours2(A.fils_gauche)
    Afficher(A.valeur)
    Parcours2(A.fils_droit)
```

Exercice 4

Partie A

1. Il y a 4 sous-réseaux :
- 192.168.20.0 / 24
- 172.16.0.0 / 16
- 192.168.0.0 / 24
- 192.168.100.0 / 24

2.a. 4 octets.

2.b. $168 = 128 + 32 + 8$ et $10 = 8 + 2$
Ligne 2 : 11000000.10101000.00010100.00001010

2.c. Ligne 3 : 11111111.11111111.11111111.00000000

2.d. Ligne 4 : 255.255.255.0

2.e. Ligne 5 : 11000000.10101000.00010100.00000000

2.f. Ligne 6 : 192.168.20.0

3.
- ☐ 192.168.20.0
- ☐ 192.256.20.11
- ☑ 192.168.20.30
- ☑ 192.168.20.230
- ☐ 192.168.20.260
- ☐ 192.168.27.11

Partie B

```
def IP_bin(liste):
    s = []
    for n in liste:
        s.append(dec_bin(n))
    return s
```

Exercice 5

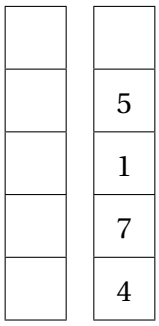
1.

None

8

False

2.



3.

```
def maximum(P) :
    maxi = 0 # On peut mettre 0 car les entiers sont positifs
    while not est_vide(P) :
        v = depile(P)
        if v > maxi:
            maxi = v
    return maxi
```

4.a.

Il faut dépiler P dans une autre pile Q en comptant le nombre d'éléments. Puis on empile à nouveau les éléments de Q dans P.

4.b.

```
def taille(P) :
    t = 0
    Q = creer_pile()
    while not est_vide(P) :
        v = depile(P)
        empile(Q,v)
        t = t + 1
    while not est_vide(Q) :
        v = depile(Q)
        empile(P,v)
    return t
```