

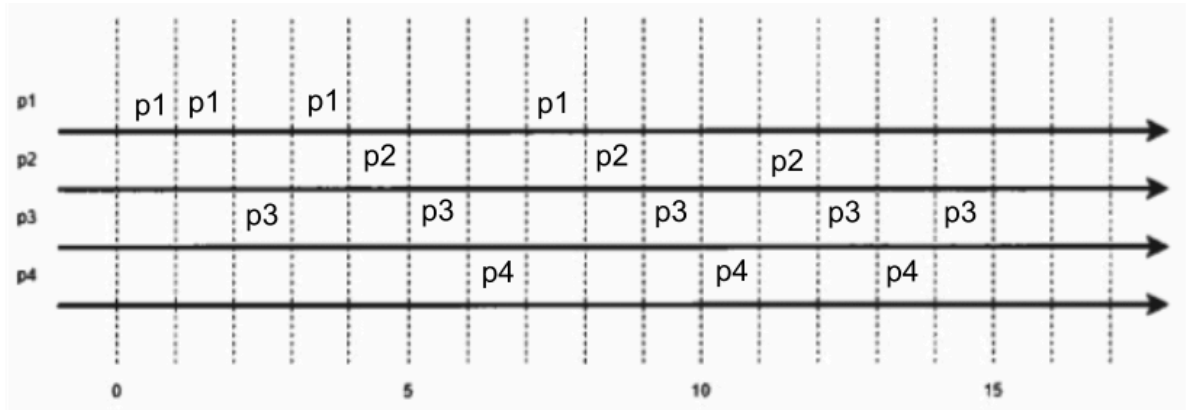
Éléments de correction sujet 01 (2024)

Exercice 1

1. prêt, élu et bloqué
2. pas de ressource à récupérer donc pas d'état "bloqué"
- 3.

```
def defile(self):  
    if len(self.contenu) == 0:  
        return None  
    else :  
        return self.contenu.pop(0)
```

4.



5.

```
def ajoute_nouveau_processus(self, proc):  
    self.file.enqueue(proc)  
def tourniquet(self):  
    self.temps += 1  
    if not self.file.est_vide():  
        proc = self.file.dequeue()  
        proc.execute_un_cycle()  
        if not proc.est_fini():  
            self.file.enqueue(proc)  
    return proc.nom  
else:  
    return None
```

6.

```
ord = Ordonnanceur()  
fin = False  
while not fin:  
    if ord.temps in depart_proc:  
        ord.ajoute_nouveau_processus(depart_proc[ord.temps])  
    elu = ord.tourniquet()  
    if elu == None:  
        fin = True  
    else :  
        print(elu)
```

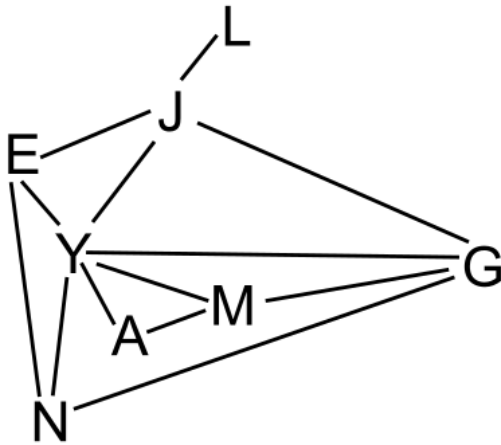
7.

B possède le clavier, D possède le fichier ; B attend le fichier avant de pouvoir libérer le clavier et D attend le clavier avant de pouvoir libérer le fichier : nous sommes donc en situation d'interblocage entre B et D

Exercice 2

Partie A

1.



2.

```
[[0, 1, 1, 0, 1, 1, 0, 0],  
 [1, 0, 1, 1, 0, 0, 0, 1],  
 [1, 1, 0, 1, 1, 1, 1, 0],  
 [0, 1, 1, 0, 1, 0, 0, 0],  
 [1, 0, 1, 1, 0, 0, 0, 0],  
 [1, 0, 1, 0, 0, 0, 1, 0],  
 [0, 0, 1, 0, 0, 1, 0, 0],  
 [0, 1, 0, 0, 0, 0, 0, 0]]
```

3.

```
>>> position(sommets, 'G')  
0  
>>> position(sommets, 'Z')  
None
```

4.

```
def nb_amis(L, m, s):  
    pos_s = position(L, s)  
    if pos_s == None:  
        return None  
    amis=0  
    for i in range(len(m)):  
        amis += m[pos_s][i]  
    return amis
```

5.

```
>>> nb_amis(sommets, matrice_adj, 'G')  
4
```

Partie B

6. c représente la clé et v représente la valeur

7.

```
graphe = {'G':['J', 'Y', 'N', 'M'],
          'J':['E', 'Y', 'L', 'G'],
          'Y':['E', 'J', 'N', 'A', 'M', 'G'],
          'E':['N', 'J', 'Y'],
          'N':['E', 'Y', 'G'],
          'M':['Y', 'A', 'G'],
          'A':['Y', 'M'],
          'L':['J']}
}
```

8.

```
def nb_amis(d, s):
    return len(d[s])
```

9.

cercle d'amis de Lou : J, G, Y, E, N

10.

```
def parcours_en_profondeur(d, s, visites = []):
    visites.append(s)
    for v in d[s]:
        if v not in visites:
            parcours_en_profondeur(d, v)
    return visites
```

Passer un tableau vide en paramètre par défaut d'une fonction récursive est une mauvaise pratique. On préférera :

```
def parcours_en_profondeur(d, s):
    visites = []
    def parcours(d, s):
        visites.append(s)
        for v in d[s]:
            if v not in visites:
                parcours(d, v)
    parcours(d,s)
    return visites
```

Exercice 3

Partie A

1. séparateur : ; (point virgule)
2. Parce que la virgule est déjà utilisée dans les données (exemple : "Allemagne, Italie, Japon")
3.

```
def charger(nom_fichier):  
    with open(nom_fichier, 'r') as f:  
        donnees = list(csv.DictReader(f, delimiter=';'))  
    return donnees
```
4. La méthode sleep (ligne 37 : `time.sleep(5)`)
5. `donnees[i]` est un dictionnaire
6.

```
flashcard = charger('flashcards.csv')  
d = choix_discipline(flashcard)  
c = choix_chapitre(flashcard, d)  
entrainement(flashcard, d, c)
```

Partie B

7.

```
INSERT INTO boite  
VALUES (5, 'tous les quinze jours', 15)
```
8.

```
UPDATE flashcard  
SET reponse = '7 décembre 1941'  
WHERE id = 5
```
9.

```
SELECT lib  
FROM discipline
```
10.

```
SELECT chapitre.lib  
FROM chapitre  
JOIN discipline ON chapitre.id_disc = discipline.id  
WHERE discipline.lib = 'histoire'
```
11.

```
SELECT flashcard.id  
FROM chapitre  
JOIN flashcard ON flashcard.id_ch = chapitre.id  
JOIN discipline ON discipline.id = chapitre.id_disc  
WHERE discipline.lib = 'histoire'
```
12.

```
DELETE FROM flashcard  
WHERE id_boite = 3
```