Exercice 1 *6 points*

1. Un processus peut se trouver dans l'état Prêt, Élu ou Bloqué.
2. S'il n'y a pas de conflit de ressources, il ne reste que les états Prêt et Élu.
3.

```
def defile(self):
    """ Renvoie le premier élément de la file et l'enlève de la file """
    if self.est_vide():
        return None
    return self.contenu.pop(0)
```

4.

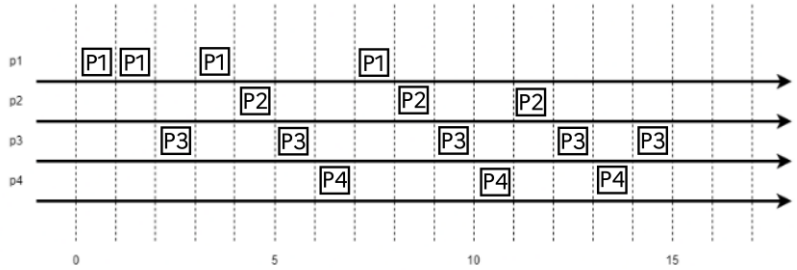


Figure 2. Chronogramme pour les processus p1, p2, p3 et p4

5.

```
class Ordonnanceur:

    def __init__(self):
        self.temps = 0
        self.file = File()

    def ajoute_nouveau_processus(self, proc):
        '''Ajoute un nouveau processus dans la file de l'ordonnanceur. '''
        self.file.enfile(proc)

    def tourniquet(self):
        '''Effectue une étape d'ordonnancement et renvoie le nom du processus élu.'''
        self.temps += 1
        if not self.file.est_vide():
            proc = self.file.defile()
            proc.execute_un_cycle()
            if not proc.est_fini():
                self.file.enfile(proc)
            return proc.nom
        else:
            return None
```

6.

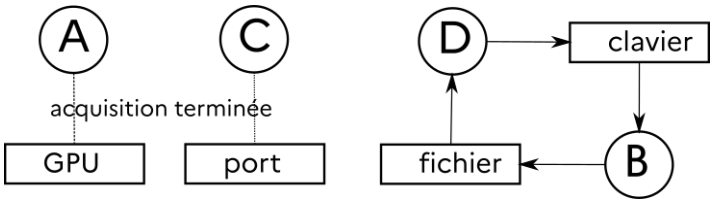
```

ord = Ordonnanceur()
fini = False
while not fini:
        if ord.temps in depart_proc:
                ord.ajoute_nouveau_processus(depart_proc[ord.temps])
```

```
        elu = ord.tourniquet()
        if elu == None:
                fini = True
        else:
                print(elu)

```

7.



On s'aperçoit qu'il y a un cycle d'interdépendance entre les processeurs B et D : il y a donc un risque d'interblocage.

Exercice 2 *6 points*

Partie A

1. Les effectifs possibles sont 6, 7, 8, 9, et 10.

2.

```

def effectif(val, lst):
        compteur = 0
        for elt in lst:
                if elt == val:
                        compteur += 1
        return compteur

```

3. La liste passée en paramètre comporte 9 éléments : il y aura 9 comparaisons.

4.

```

def majo_abs1(lst):
        for elt in lst:
                if effectif(elt, val) > len(lst)//2:
                        return elt
        return None

```

5. La liste passée en paramètre comporte 9 éléments. Le nombre 1, premier élément, se retrouve 5 fois dans la liste : il est donc majoritaire. La fonction `majo_abs1` va donc s'arrêter dès le premier élément : il y aura eu 9 comparaisons.

Partie B

6.

```
def eff_dico(lst):
    dico_sortie = {}
    for elt in lst :
        if elt in dico_sortie:
            dico_sortie[elt] += 1
        else:
            dico_sortie[elt] = 1
    return dico_sortie
```

7.

```
1
2  def majo_abs2(lst):
3          dico = eff_dico(lst)
4          for elt in dico:
5                  if dico[elt] > len(lst)//2:
6                          return elt
7          return None
8
```

Partie C

8. Il n'y a qu'un seul élément : il est donc forcément absolument majoritaire.

9. Supposons qu'un élément `elt` est majoritaire dans une liste de taille `n`. Appelons `p` l'effectif de `n` dans `lst`. On a donc `p > n//2`.

Séparons la liste `lst` en deux listes `lst1` et `lst2`, de sorte que `lst = lst1 + lst2`.

Appelons `k1` et `k2` l'effectif de `elt` dans `lst1` et `lst2`.

On a `p = k1 + k2`

Il est clair que `k1` et `k2` ne peuvent être simultanément inférieurs à `n//4`, sinon `p = k1 + k2` ne peut pas être supérieur à `n//2`.

Donc soit `k1`, soit `k2` (soit les deux...) est supérieur à `n//4`.

Donc `elt` est absolument majoritaire dans `lst1`, ou dans `lst2` (ou aussi dans les 2).

Si on suppose donc que ni `lst1` ni `lst2` n'admet d'élément absolument majoritaire, il ne peut pas y avoir d'élément majoritaire dans `lst`. (car s'il y avait un élément majoritaire, la démonstration précédente nous prouve qu'il y en aurait dans `lst1` ou `lst2`).

10. Il suffit de vérifier si l'effectif de `maj1` dans `lst` est supérieur à `len(lst)/2`.

11.

```
1  def majo_abs3(lst):
2      n = len(lst)
3      if n == 1:
4          return lst[0]
5      else:
6          lst_g = lst[:n//2]
7          lst_d = lst[n//2:]
8          maj_g = majo_abs3(lst_g)
9          maj_d = majo_abs3(lst_d)
10         if maj_g is not None:
11             eff = effectif(maj_g, lst)
12             if eff > n/2:
13                 return maj_g
14         if maj_d is not None:
15             eff = effectif(maj_d, lst)
16             if eff > n/2:
17                 return maj_d
```

Exercice 3 *8 points*

Partie A

1. Une adresse IPv4 est composée de 4 octets.

2.
- `Serveur_web` : `172.16.0.1`
- `Serveur_BDD` : `172.16.0.2`

3. La commande `ping` permet de tester si la machine pingée est accessible.

4. L'adresse de la passerelle n'est pas bonne : il faut mettre `192.168.1.254`.

Partie B

5. `PC_A1 -> routeur A -> routeur B -> routeur C -> routeur D -> Serveur_impression`
6. `PC_A1 -> routeur A -> routeur B -> routeur C -> ?`. Les paquets sont perdus une fois arrivés au routeur C. Ils n'arrivent pas à destination.
7.

Routeur c		
Destination	Prochain saut	Métrique
172.16.0.0	10.0.2.2	2
192.168.0.0	10.0.2.2	2
192.168.1.0	10.0.2.2	2
192.168.2.0	10.0.3.2	1
192.168.3.0	10.0.4.2	?
10.0.0.0	10.0.2.2	1
10.0.1.0	10.0.2.2	1
10.0.2.0	-	-
10.0.3.0	-	-
10.0.4.0	-	-
10.0.5.0	10.0.3.2 ou 10.0.4.2	1
0.0.0.0	10.0.2.2	2

8. `PC_A1 -> routeur A -> routeur B -> routeur C -> routeur D -> Serveur_impression`
9. La liaison entre le routeur C et D possède un débit de 10 Mb/s. Alors que la liaison entre C et E puis E et D possède un débit de 1 Gb/s, ce qui est 100 fois plus rapide. Il faut donc préférer ce chemin.
10. Sur la table de routage de C, la ligne de destination vers `192.168.2.0` aura comme prochain saut `10.0.4.2` et comme métrique 2. La ligne de destination vers `10.0.5.0` aura comme prochain saut `10.0.4.2` et comme métrique 1.

Partie C

11.

```
SELECT titre_parution
FROM parution
```

12. Cette requête renvoie le numéro de parution et le numéro de la page de toutes les pages où la police employée est du Arial 12, classés par numéro de parution.
13.

```
SELECT num_image, titre_image, poids
FROM image
WHERE poids > 1000
```

14. Cette requête renvoie le numéro de parution de tous les magazines qui comportent une image dont le titre contient le mot "Appolo".
15. Cette requête va insérer dans la table `image` une image de numéro 2923, intitulée "Volcans du massif central", sans descriptif, de taille 400 sur 400 et de poids 1430 Ko.
16.

```
INSERT INTO texte
VALUES (2754, "Vulcania", "Parc d'attraction", 250);
```

17. Cette requête va supprimer de la relation `texte` le texte ayant pour numéro 2034.
18.

```
DELETE FROM comporte_texte
WHERE num_texte = 2034;
```