

# POLYNÉSIE – 2023 – sujet1

## Exercice 2 (3 points)

Cet exercice porte sur la gestion des processus et la programmation orientée objet.

On rappelle qu'un processus est l'instance d'un programme en cours d'exécution. Il est identifié par un numéro unique appelé PID. L'ordonnanceur est la composante du système d'exploitation qui gère l'allocation du processeur entre les différents processus. Nous allons nous intéresser à l'algorithme d'ordonnement du tourniquet dont le fonctionnement est résumé ci-dessous :

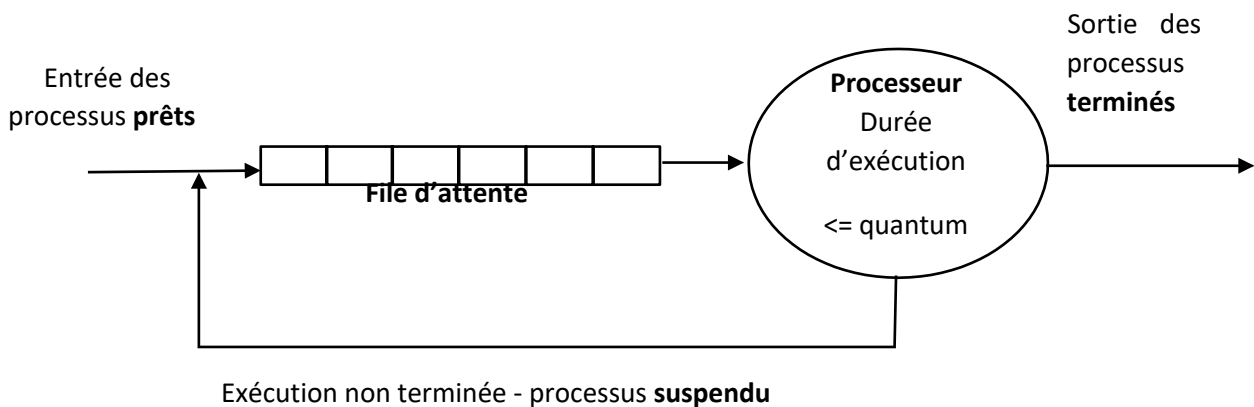


Schéma d'ordonnement du tourniquet

- Les processus prêts à être exécutés sont placés dans une file d'attente selon leur ordre d'arrivée ;
- L'ordonnanceur alloue le processeur à chaque processus de la file d'attente un même nombre de cycles CPU, appelé **quantum** ;
- Si le processus n'est pas terminé au bout de ce temps, son exécution est suspendue et il est mis à la fin de la file d'attente ;
- Si le processus est terminé, il sort définitivement de la file d'attente.

1. On considère trois processus soumis à l'ordonnanceur **au même instant** pour lesquels on donne les informations ci-dessous :

PID	Durée (en cycles CPU)	Ordre d'arrivée
11	4	1
20	2	2
32	3	3

- a. Si le quantum du tourniquet est d'un cycle CPU, recopier et compléter la suite des PID des processus dans l'ordre de leur exécution :  
11, 20, 32, 11, .....



```
def execute_un_cycle(self):
    """Met à jour le reste à faire après l'exécution d'un
    cycle."""
    .....

def change_etat(self, nouvel_etat):
    """Change l'état du processus avec la valeur passée en
    paramètre."""
    .....

def est_termine(self):
    """Renvoie True si le processus est terminé, False sinon,
    en se basant sur le reste à faire."""
    .....
```

- c. La fonction `tourniquet` ci-dessous implémente l'algorithme décrit dans l'exercice. Elle prend en paramètres une liste d'objets `Processus` donnés par ordre d'arrivée et un nombre entier positif correspondant au quantum. La fonction renvoie la liste des PID dans l'ordre de leur exécution par le processeur. Recopier et compléter sur la copie le code manquant.

```
1 def tourniquet(liste_attente, quantum):
2     ordre_execution = []
3     while liste_attente != []:
4         # On extrait le premier processus
5         processus = liste_attente.pop(0)
6         processus.change_etat("En cours d'exécution")
7         compteur_tourniquet = 0
8         while ..... and .....:
9             ordre_execution.append(. .... )
10            processus.execute_un_cycle()
11            compteur_tourniquet = compteur_tourniquet + 1
12            if ..... :
13                processus.change_etat("Suspendu")
14                liste_attente.append(processus)
15            else:
16                processus.change_etat(. .... )
17    return ordre_execution
```