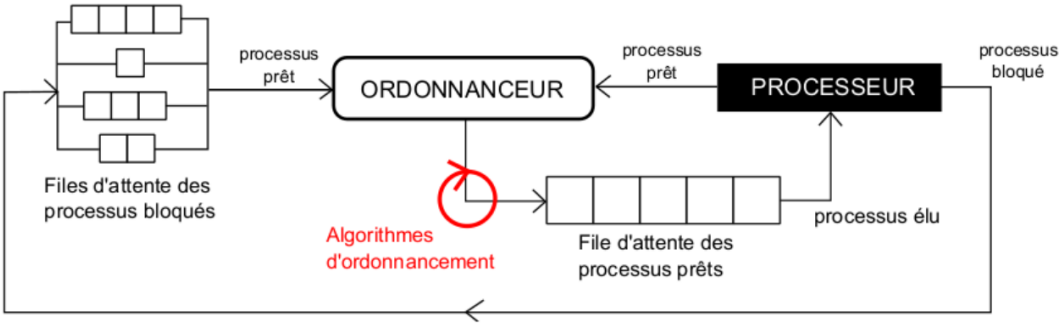
# Exercices - Gestion des processus et ressources

## Exercice 1

Un **système d'exploitation** attribue à des processus un état : élu, prêt ou bloqué .
Plus précisément, c'est l'**ordonnanceur** (un des composants du système d'exploitation) qui réalise cette tâche appelée **ordonnancement des processus** . L'objectif de l'ordonnanceur est de **choisir** le processus à exécuter à l'instant $t$ (le processus élu ) et déterminer le temps durant lequel le processeur lui sera alloué.




Ce choix est à faire parmi tous les processus qui sont dans l'état prêt , mais lequel sera élu ? et pour combien de temps ?

Des **algorithmes d'ordonnancement** sont utilisés et il en existe plusieurs selon la stratégie utilisée.
La suite de cet exercice présente les descriptions de 5 algorithmes d'ordonnancement.
A chacun de ces algorithmes, on associe un **diagramme de Gantt** représentant l'exécution des processus dans le temps.

1. Donnez plusieurs exemples de système d'exploitation.
2. Expliquez la différence entre un programme et un processus.
3. Expliquez pourquoi il est indispensable d'utiliser un ordonnanceur de processus.
4. Que veut dire selon vous l'expression *"Exécution concurrente"*.
5. Que veut dire selon vous l'expression *"Accès concurrents aux ressources"*.
6. Les définitions de 5 algorithmes d'ordonnancement vous sont données. Chaque définition associe un exemple représenté par un diagramme de Gantt. A vous de retrouver quel diagramme est associé à quelle définition.

### Définition 1

**Ordonnancement First Come First Served (FCFS)**
*Principe* : Les processus sont ordonnancés selon leur ordre d'arrivée ("premier arrivé, premier servi" en français).
*Exemple* : Les processus $P_1(53)$, $P_2(17)$, $P_3(68)$ et $P_4(24)$ arrivent dans cet ordre à $t = 0$ :
Cela signifie que $P_1$ , $P_2$, $P_3$ et $P_4$ ont besoin de respectivement 53, 17, 68 et 24 unités de temps pour s'exécuter.

### Définition 2

**Ordonnancement Shortest Job First (SJF)**
*Principe* : Le processus dont le temps d'exécution est le plus court est ordonnancé en premier.
*Exemple* : Les processus $P_1$ , $P_2$ , $P_3$ et $P_4$ arrivent dans cet ordre à $t = 0$.

## Définition 3

**Ordonnancement Shortest Remaining Time (SRT)**
*Principe* : Le processus dont le temps d'exécution restant est le plus court parmi ceux qui restent à exécuter est ordonnancé en premier.
*Exemple* : $P_3$ et $P_4$ arrivent à $t = 0$ ; $P_2$ à $t = 20$ ; $P_1$ à $t = 50$ :

## Définition 4

**Ordonnancement temps-partagé (Round-Robin)**
*Principe* : C'est la politique du **tourniquet** : allocation du processeur par tranche de temps . On appelle une tranche de temps un **quantum, noté q**.
Exemple : quantum q = 20, n = 4 processus.

## Définition 5

**Ordonnancement à priorités statiques**
*Principe* : Allocation du processeur selon des priorités statiques (= numéros affectés aux processus pour toute la vie de l'application).
*Exemple* : priorités $(P_1,\ P_2,\ P_3,\ P_4) = (3,\ 2,\ 0,\ 1)$ où la priorité la plus forte est $0$.
(attention, dans certains systèmes c'est l'inverse : 0 est alors la priorité la plus faible)

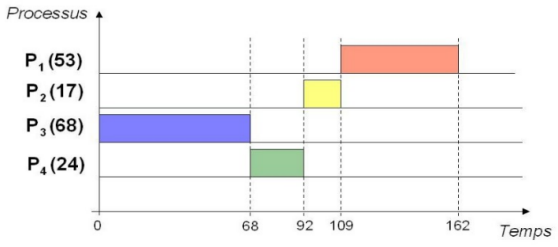



Diagramme 1

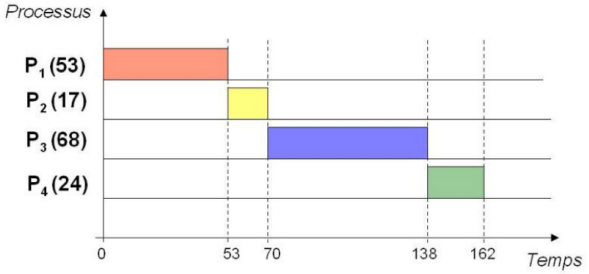



Diagramme 2

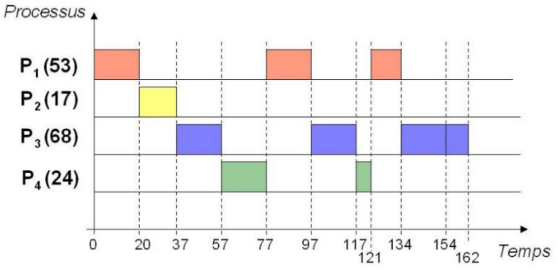



Diagramme 3

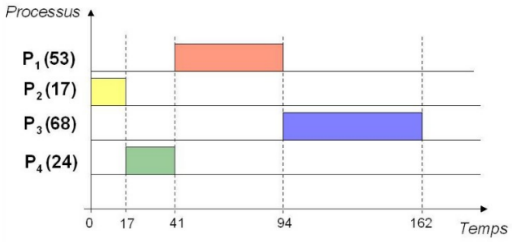



Diagramme 4

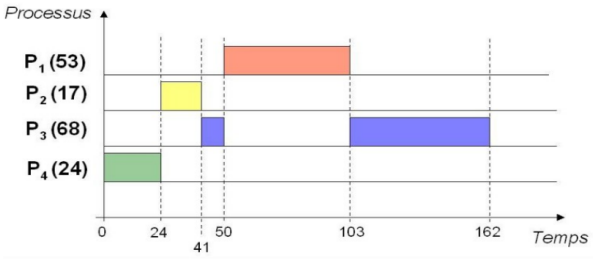



Diagramme 5

## Exercice 2

**Application de plusieurs algorithmes**
Donnez le diagramme de Gantt pour l'exécution de ces différents processus en utilisant successivement les algorithmes :

- FCFS
- RR (quantum = 2 unités de temps)
- SRT

| Processus | Date d'arrivée | Durée de traitement |
|---|---|---|
| $P_1$ | 0 | 3 |
| $P_2$ | 1 | 6 |
| $P_3$ | 3 | 4 |
| $P_4$ | 6 | 5 |
| $P_5$ | 8 | 2 |

**Performances des algorithmes d'ordonnancement**
On définit les métriques suivantes :

- **temps de séjour** (ou d'exécution) (ou de rotation) d'un processus : c'est la différence entre la date de fin d'exécution et la date d'arrivée :

$$Tsej = \text{date fin d'exécution} - \text{date d'arrivée}$$

- **temps d'attente** d'un processus : c'est la différence entre le temps de séjour et la durée du processus :

$$T_{att} = T_{sej} - \text{durée du processus}$$

- **rendement d'un processus** : c'est le quotient entre la durée du processus et le temps de séjour :

$$\text{rendement} = \frac{\text{durée du T processus}}{T_{sej}}$$

1. Pour chacun des trois algorithmes, calculez le temps de séjour, le temps d'attente et le rendement de chaque processus. Vous représenterez vos résulats sous la forme d'un tableau par algorithme.

| Processus | $P_1$ | $P_2$ | $P_3$ | $P_4$ | $P_5$ |
|---|---|---|---|---|---|
| Date d'arrivée | 0 | 1 | 3 | 6 | 8 |
| Durée d'exécution | 3 | 6 | 4 | 5 | 2 |
| Temps de séjour | | | | | |
| Temps d'attente | | | | | |
| Rendement | | | | | |

2. Quel vous semble être le meilleur des trois algorithmes dans notre exemple ? Expliquer.

## Exercice 3 QCM)

Pour chacune des questions, une seule des quatre réponses est exacte.

1. Parmi les commandes ci-dessous, laquelle permet d'afficher les processus en cours d'exécution ?
   (a) dir
   (b) ps
   (c) man
   (d) ls
2. Quelle abréviation désigne l'identifiant d'un processus dans un système d'exploitation de type UNIX ?
   (a) PIX
   (b) SIG
   (c) PID
   (d) SID

3. Comment s'appelle la gestion du partage du processeur entre différents processus ?
   (a) L'interblocage
   (b) L'ordonnancement
   (c) La planification
   (d) La priorisation

4. Quelle commande permet d'interrompre un processus dans un système d'exploitation de type UNIX ?
   (a) stop
   (b) interrupt
   (c) end
   (d) kill

## **Exercice** 4

Un processeur choisit à chaque cycle d'exécution le processus qui doit être exécuté. Le tableau ci-dessous donne pour trois processus P1, P2, P3 :

- la durée d'exécution (en nombre de cycles),
- l'instant d'arrivée sur le processeur (exprimé en nombre de cycles à partir de 0),
- le numéro de priorité.

Le numéro de priorité est d'autant plus petit que la priorité est grande. On suppose qu'à chaque instant, c'est le processus qui a le plus petit numéro de priorité qui est exécuté, ce qui peut provoquer la suspension d'un autre processus, lequel reprendra lorsqu'il sera le plus prioritaire.

| Processus | Durée d'exécution | Instant d'arrivée | Numéro de priorité |
|---|---|---|---|
| P1 | 3 | 3 | 1 |
| P2 | 3 | 2 | 2 |
| P3 | 4 | 0 | 3 |

1. Reproduire le tableau ci-dessous sur la copie et indiquer dans chacune des cases le processus exécuté à chaque cycle.


2. On suppose maintenant que les trois processus précédents s'exécutent et utilisent une ou plusieurs ressources parmi R1, R2 et R3. Parmi les scénarios suivants, lequel provoque un interblocage ? Justifier

| Scénario 1 |
|---|
| P1 acquiert R1 |
| P2 acquiert R2 |
| P3 attend R1 |
| P2 libère R2 |
| P2 attend R1 |
| P1 libère R1 |

| Scénario 2 |
|---|
| P1 acquiert R1 |
| P2 acquiert R3 |
| P3 acquiert R2 |
| P1 attend R2 |
| P2 libère R3 |
| P3 attend R1 |

| Scénario 3 |
|---|
| P1 acquiert R1 |
| P2 acquiert R2 |
| P3 attend R2 |
| P1 attend R2 |
| P2 libère R2 |
| P3 acquiert R2 |

## **Exercice** 5

1. La commande `ps` suivie éventuellement de diverses options permet de lister les processus actifs ou en attente sur une machine. Sur une machine équipée du système d'exploitation GNU/Linux, la commande `ps -aef` permet d'obtenir la sortie suivante (extrait) :

| UID | PID | PPID | C | STIME | TTY | TIME | CMD |
|---|---|---|---|---|---|---|---|
| root | 1 | 0 | 0 | 10:01 | ? | 00:00:02 | /sbin/init splash |
| root | 4 | 2 | 0 | 10:01 | ? | 00:00:00 | [kworker/0:0H] |
| root | 6 | 2 | 0 | 10:01 | ? | 00:00:00 | [mm_percpu_wq] |
| .... | .... | ..... | .... | ..... | ..... | ..... | ..... |
| bob | 3383 | 1 | 0 | 10:25 | ? | 00:00:00 | sh -c /usr/bin/google-chrome-sta |

   (a) Quelle est la particularité de l'utilisateur "root" ?

   (b) Quel est le processus parent du processus ayant pour PID 3383

2. Dans un bureau d'architectes, on dispose de certaines ressources qui ne peuvent être utilisées simultanément par plus d'un processus, comme l'imprimante, la table traçante, le modem. Chaque programme, lorsqu'il s'exécute, demande l'allocation des ressources qui lui sont nécessaires. Lorsqu'il a fini de s'exécuter, il libère ses ressources. .

| Programme 1 | Programme 2 | Programme 3 |
|---|---|---|
| demander(table traçante)<br>demander (modem)<br>exécution<br>libérer (modem)<br>libérer (table traçante) | demander (modem)<br>demander (imprimante)<br>exécution<br>libérer (imprimante)<br>libérer (modem) | demander (imprimante)<br>demander (table traçante)<br>exécution<br>libérer (table traçante)<br>libérer (imprimante) |

   On appelle p1, p2 et p3 les processus associés respectivement aux programmes 1, 2 et 3. Un ordonnancement de type round robin est uilisé. A chaque tour, un processus n'exécute qu'une seule instruction. L'ordre d'exécution est donc le suivant : p1, p2, p3, p1, p2, p3...

   (a) Justifier qu'une situation d'interblocage peut se produire.

   (b) Modifier l'ordre des instructions du programme 3 pour qu'une telle situation ne puisse pas se produire

## **Exercice** 6

Soient trois processus pour lesquels la date d'arrivée, la durée des calculs (durée de traitement, durée d'exécution par l'unité de calcul) sont données ci-après :

| Processus | Date d'arrivée | Durée de traitement |
|---|---|---|
| P1 | 0 | 4 |
| P2 | 0 | 5 |
| P3 | 1 | 2 |

Le processus dont le temps restant est le « plus court » de la file d'attente est traité en premier par l'unité de calcul. La situation est réévaluée lors de la terminaison d'un processus ou apparition d'un nouveau processus.

1. Donnez le diagramme de Gantt pour l'exécution de ces 3 processus. Vous indiquerez par un flèche sur le diagramme la date d'arrivée des 3 processsus.
2. Calculez le temps de séjour, le temps d'attente et le rendement de chaque processus. Vous représenterez vos résulats sous la forme d'un tableau.

| Processus | $P_1$ | $P_2$ | $P_3$ |
|---|---|---|---|
| Date d'arrivée | | | |
| Durée d'exécution | | | |
| Temps de séjour | | | |
| Temps d'attente | | | |
| Rendement | | | |

## **Exercice** 7

Soient cinq processus pour lesquels la date d'arrivée, la durée des calculs (durée de traitement, durée d'exécution par l'unité de calcul) sont données ci-après :

| Processus | Date d'arrivée | Durée de traitement |
|---|---|---|
| P1 | 0 | 7 |
| P2 | 1 | 4 |
| P3 | 1 | 2 |
| P4 | 2 | 2 |
| P5 | 3 | 1 |

Lors de leur apparition les processus sont stockés dans une file d'attente et sont traités par bloc d'instructions à tour de rôle pendant un quantum de temps maximum.

1. Donnez les diagrammes de Gantt pour l'exécution de ces 5 processus dans les deux cas suivants :
   - Quantum = 1 unité de temps
   - Quantum = 2 unités de temps
2. Pour chacun des deux cas, calculez le temps de séjour, le temps d'attente et le rendement de chaque processus. Vous représenterez vos résulats sous la forme d'un tableau.

| Processus | $P_1$ | $P_2$ | $P_3$ | $P_4$ | $P_5$ |
|---|---|---|---|---|---|
| Date d'arrivée | | | | | |
| Durée d'exécution | | | | | |
| Temps de séjour | | | | | |
| Temps d'attente | | | | | |
| Rendement | | | | | |



Source : Lycée Mounier - Angers