TD 3 : Processus

IUT Aix-Marseille / DUT R&T 1ière année
J. Seinturier (http://www.seinturier.fr)

1. Etude de processus

1.1. Quelle est la commande permettant de lister tous les processus du système ?

Soit le résultat d'affichage de la commande demandée ci-dessus : :

```
root 1680 0.2 0.0    3200  5684 ?    S 11:15 0:00 init
root 1683 0.2 0.0  119720  3624 ?    S 11:15 0:00 /usr/lib/udisks/udisks
root 1715 0.0 0.0    4204    88 ?    S 11:16 0:00 /usr/lib/kde4/graphicalui
root 1716 0.0 0.5  348520 23056 ?    S 11:16 0:00 /usr/lib/kde4/soundunit
jse  1717 0.0 0.2  352504 11012 ?    S 11:18 0:00 /usr/bin/konsole
jse  1720 0.5 0.9 1189684 36624 tty1 S 11:19 0:00 /usr/bin/play film.mp4 &
jse  1721 4.5 1.4 3184672 16421 ?    S 11:19 0:04 soundserver -t mp3
jse  1722 9.8 2.6  174531 25332 ?    S 11:19 0:00 ffdcodec -t avi
root 1723 0.1 0.0     596   265 tty2 S 11:23 0:00 ps -aux
```

1.2. A quelle heure le système a-t-il été démarré ?

1.3. Que peut être l'utilité du programme `udisks` ?

1.4. A partir de quelle heure l'utilisateur `jse` a-t-il commencé à utiliser l'ordinateur ?

1.5. Que fait l'utilisateur `jse` au moment du listing des processus ?

1.6. Proposer une arborescence des processus correspondant à l'état du système listé en 1.2

1.7. Que se passe-t'il si l'utilisateur `jse` tape la commande `kill -9 1680` ?

1.8. Que se passe-t'il si l'utilisateur `root` tape la commande `kill -15 1720 ?`

1.9. Quelle est la quantité de RAM en ko utilisée sur la machine par l'utilisateur `jse` ?

2. Ordonnancement

Un système d'exploitation installé sur une machine disposant d'un seul processeur doit gérer 5 processus A, B, C, D et E. Chaque processus est décrit par le nombre d'instructions qu'il contient, son temps de lancement, son PID et l'UID de l'utilisateur qui l'a lancé:

Processus	PID	#Instructions	Temps de lancement	UID
A	1	3	1	1
B	2	6	2	2
C	3	4	3	2
D	4	2	4	1
E	5	3	6	3

Le processeur de la machine peut traiter **1 instruction par cycle**, les temps de lancements étant, pour des raisons de simplicité, exprimés eux aussi en cycle processeurs (un temps de lancement de 2 indique que le programme a été lancé au début du deuxième cycle processeur).

Une table d'ordonnancement peut être écrite pour un ensemble de processus sous la forme du tableau :

Cycle	Processus en exécution	Etat de fin de cycle	#Instructions restantes

Ou la colonne cycle contient le numéro du cycle processeur (le premier étant 1), la colonne *Processus en exécution* contient le processus présent dans le processeur durant le cycle et *Etat de fin ce cycle* contient l'état du processus à la fin du cycle. Cet état peut être : **en attente**, **en exécution** ou **arrêté.**

2.1. Ordonnancement Round Robin (Tourniquet)

En utilisant un ordonnanceur de type *Round Robin* :

1. écrire la table d'ordonnancement du système pour traiter l'ensemble des processus.
2. Calculer pour chaque processus son temps d'attente (nombre de cycle avant la première exécution du processus) et son temps d'exécution total (nombre de cycles entre l'exécution de sa dernière instruction et son lancement).

2.2 Ordonnancement avec priorité

Nous utilisons maintenant un ordonnanceur basé sur des priorités qui fonctionne de la manière suivante :

- Chaque processus possède une priorité sous forme de nombre entier entre 0 et 10
- La priorité maximale d'un processus et 0, la minimale est 10
- Au lancement, un processus à une priorité de 0
- A chaque cycle d'exécution, la priorité du processus augmente de 1
- A chaque cycle, l'ordonnanceur exécute le processus le plus prioritaire
- A chaque cycle, la priorité des processus non exécuté diminue de 1 (sans pouvoir passer sous 0)
- Chaque fois que 2 processus on même priorité, celui qui a le moins d'instructions est choisi

En utilisant cet ordonnanceur :

3. écrire la table d'ordonnancement du système pour traiter l'ensemble des processus (ajouter à la table une colonne *priorité* qui contient la priorité du processus à la fin du cycle).
4. Calculer pour chaque processus son temps d'attente et son temps d'exécution.

2.3. Comparer les tables d'ordonnancement des questions 2.1 et 2.2. Quelle est la plus intéressante ?

2.4. Combien de cycles sont nécessaires pour traiter tous les processus dans les 2 cas ?