

EXERCICE 2 (6 points)

Cet exercice porte sur les systèmes d'exploitation, les commandes UNIX, les structures de données (de type LIFO et FIFO) et les processus.

“Linux ou GNU/Linux est une famille de systèmes d'exploitation open source de type Unix fondée sur le noyau Linux, créé en 1991 par Linus Torvalds. De nombreuses distributions Linux ont depuis vu le jour et constituent un important vecteur de popularisation du mouvement du logiciel libre.”

Source : Wikipédia, extrait de l'article consacré à GNU/Linux.

“Windows est au départ une interface graphique unifiée produite par Microsoft, qui est devenue ensuite une gamme de systèmes d'exploitation à part entière, principalement destinés aux ordinateurs compatibles PC. Windows est un système d'exploitation propriétaire.”

Source : Wikipédia, extrait de l'article consacré à Windows.

1. Expliquer succinctement les différences entre les logiciels libres et les logiciels propriétaires.
2. Expliquer le rôle d'un système d'exploitation.

On donne ci-dessous une arborescence de fichiers sur un système GNU/Linux (les noms encadrés représentent des répertoires, les noms non encadrés représentent des fichiers, / correspond à la racine du système de fichiers) :

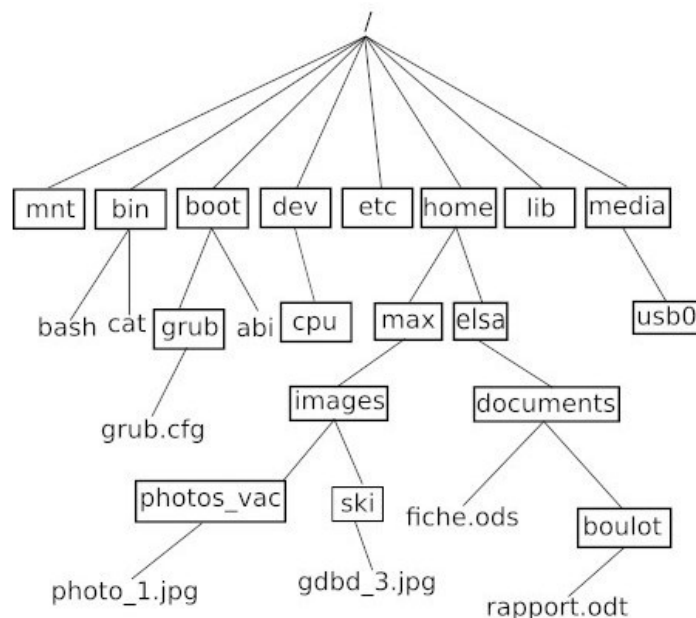


Figure 1. Arborescence de fichiers

3. Indiquer le chemin absolu du fichier `rapport.odt`.

On suppose que le répertoire courant est le répertoire `elsa`.

4. Indiquer le chemin relatif du fichier `photo_1.jpg`.

L'utilisatrice Elsa ouvre une console (aussi parfois appelée terminal), le répertoire courant étant toujours le répertoire `elsa`. On fournit ci-dessous un extrait du manuel de la commande UNIX `cp` :

NOM

`cp - copie un fichier`

UTILISATION

`cp fichier_source fichier_destination`

5. Déterminer le contenu du répertoire `documents` et du répertoire `boulot` après avoir exécuté la commande suivante dans la console :

```
cp documents/fiche.ods documents/boulot
```

“Un système d’exploitation est multitâche (en anglais : multitasking) s’il permet d’exécuter, de façon apparemment simultanée, plusieurs programmes informatiques. GNU/Linux, comme tous les systèmes d’exploitation modernes, gère le multitâche.”

“Dans le cas de l’utilisation d’un monoprocesseur, la simultanéité apparente est le résultat de l’alternance rapide d’exécution des processus présents en mémoire.”

Source : Wikipédia, extraits de l’article consacré au Multitâche.

Dans la suite de l’exercice, on s’intéresse aux processus. On considère qu’un monoprocesseur est utilisé. On rappelle qu’un processus est un programme en cours d’exécution. Un processus est soit élu, soit bloqué, soit prêt.

6. Recopier et compléter le schéma ci-dessous avec les termes suivants :
élu, bloqué, prêt, élection, blocage, déblocage.

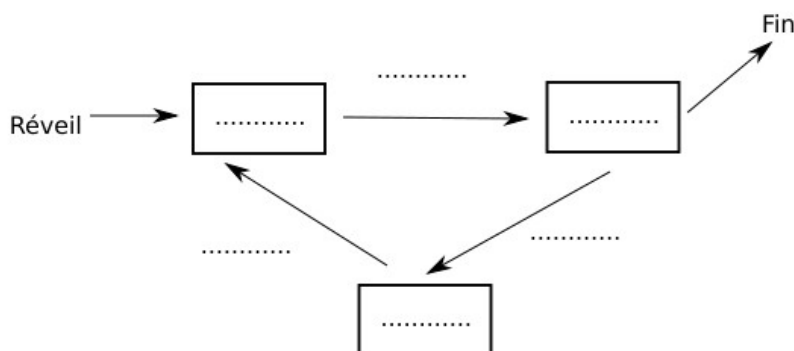


Figure 2. Schéma processus

7. Donner l'exemple d'une situation qui contraint un processus à passer de l'état élu à l'état bloqué.

“Dans les systèmes d’exploitation, l’ordonnanceur est le composant du noyau du système d’exploitation choisissant l’ordre d’exécution des processus sur le processeur d’un ordinateur.”

Source : Wikipédia, extrait de l’article consacré à l’ordonnancement.

L’ordonnanceur peut utiliser plusieurs types d’algorithmes pour gérer les processus.

L’algorithme d’ordonnancement par “ordre de soumission” est un algorithme de type FIFO (First In First Out), il utilise donc une file.

8. Nommer une structure de données linéaire de type LIFO (Last In First Out).

À chaque processus, on associe un instant d’arrivée (instant où le processus demande l’accès au processeur pour la première fois) et une durée d’exécution (durée d’accès au processeur nécessaire pour que le processus s’exécute entièrement).

Par exemple, l’exécution d’un processus P4 qui a un instant d’arrivée égal à 7 et une durée d’exécution égale à 2 peut être représentée par le schéma suivant :

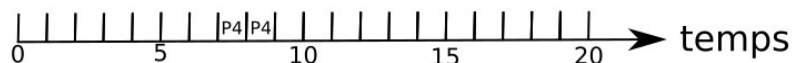


Figure 3. Utilisation du processeur

L’ordonnanceur place les processus qui ont besoin d’un accès au processeur dans une file, en respectant leur ordre d’arrivée (le premier arrivé étant placé en tête de file). Dès qu’un processus a terminé son exécution, l’ordonnanceur donne l’accès au processus suivant dans la file.

Le tableau suivant présente les instants d’arrivées et les durées d’exécution de cinq processus :

5 processus		
Processus	instant d’arrivée	durée d’exécution
P1	0	3
P2	1	6
P3	4	4
P4	6	2
P5	7	1

9. Recopier et compléter le schéma ci-dessous avec les processus P1 à P5 en utilisant les informations présentes dans le tableau ci-dessus et l’algorithme d’ordonnancement “par ordre de soumission”.

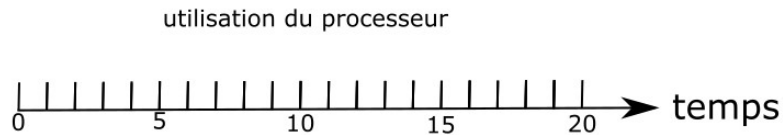


Figure 4. Utilisation du processeur

On utilise maintenant un autre algorithme d’ordonnement : l’algorithme d’ordonnement “par tourniquet”. Dans cet algorithme, la durée d’exécution d’un processus ne peut pas dépasser une durée Q appelée quantum et fixée à l’avance. Si ce processus a besoin de plus de temps pour terminer son exécution, il doit retourner dans la file et attendre son tour pour poursuivre son exécution.

Par exemple, si un processus $P1$ a une durée d’exécution de 3 et que la valeur de Q a été fixée à 2, $P1$ s’exécutera pendant deux unités de temps avant de retourner à la fin de la file pour attendre son tour ; une fois à nouveau élu, il pourra terminer de s’exécuter pendant sa troisième et dernière unité de temps d’exécution.

10. Recopier et compléter le schéma ci-dessous, en utilisant l’algorithme d’ordonnement “par tourniquet” et les mêmes données que pour la question 9, en supposant que le quantum Q est fixé 2.

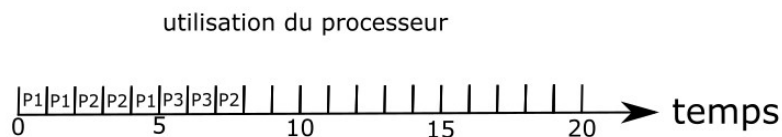


Figure 5. Utilisation du processeur

On considère deux processus $P1$ et $P2$, et deux ressources $R1$ et $R2$.

11. Décrire une situation qui conduit les deux processus $P1$ et $P2$ en situation d’interblocage.