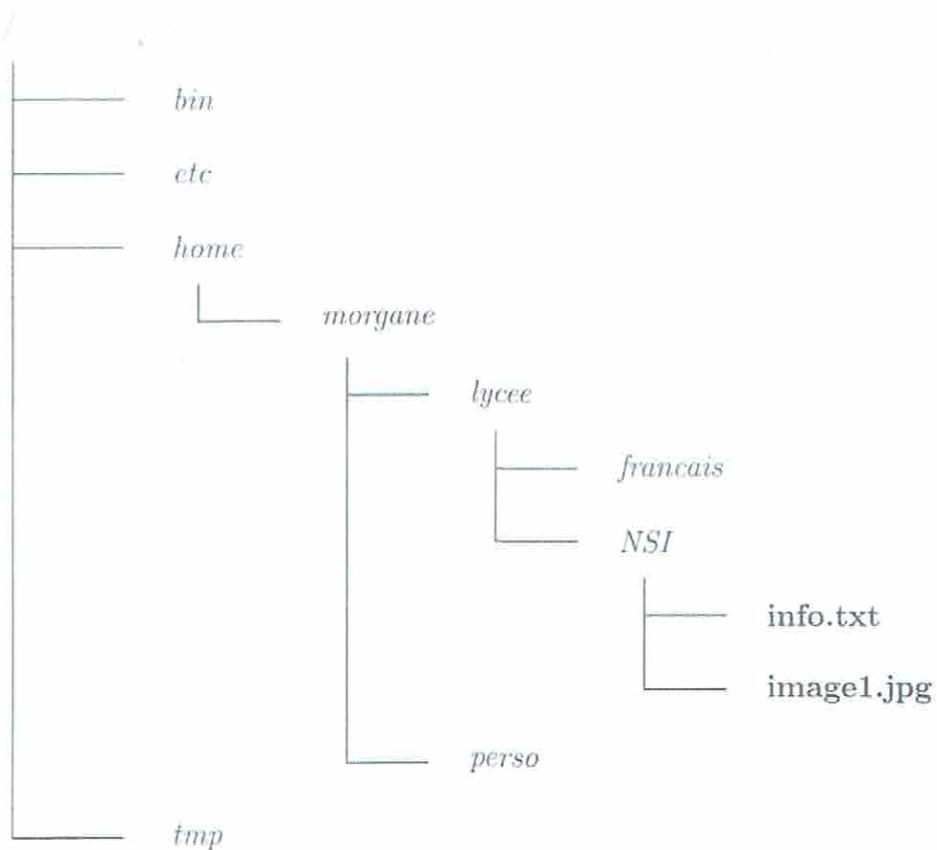


Cet exercice porte sur les systèmes d'exploitation et la gestion des processus par un système d'exploitation.

Cet exercice pourra utiliser des commandes de systèmes d'exploitation de type UNIX telles que cd, ls, mkdir, rm, rmdir, mv, cat.

1. Dans un système d'exploitation de type UNIX, on considère l'arborescence des fichiers suivante dans laquelle les noms de dossiers sont en italique et ceux des fichiers sont en gras :



On souhaite, grâce à l'utilisation du terminal de commande, explorer et modifier les répertoires et fichiers présents.

On suppose qu'on se trouve actuellement à l'emplacement **/home/morgane**.

- (a) Parmi les quatre propositions suivantes, donner celle correspondant à l'affichage obtenu lors de l'utilisation de la commande **ls**.

Proposition 1 : lycee francais NSI info.txt image1.jpg perso

Proposition 2 : lycee perso

Proposition 3 : morgane

Proposition 4 : bin etc home tmp

- (b) Écrire la commande qui permet, à partir de cet emplacement, d'atteindre le répertoire **lycee**.

On suppose maintenant qu'on se trouve dans le répertoire `/home/morgane/lycee/NSI`.

- (c) Écrire la commande qui permet de créer à cet emplacement un répertoire nommé `algorithmique`.
- (d) Écrire la commande qui permet, à partir de cet emplacement, de supprimer le fichier `image1.jpg`.
2. On rappelle qu'un processus est une instance d'application. Un processus peut être démarré par l'utilisateur, par un périphérique ou par un autre processus appelé parent.

La commande UNIX `ps` présente un cliché instantané des processus en cours d'exécution.

On a exécuté la commande `ps` (avec quelques options qu'il n'est pas nécessaire de connaître pour la réussite de cet exercice). Un extrait du résultat de la commande est présenté ci-dessous :

UID	PID	PPID	C	STIME	TTY	TIME	CMD
test	900	739	0	10:51	?	00:00:00	/usr/lib/gvfs/gvfs-udisks2-vol
test	907	838	0	10:51	?	00:00:00	/usr/lib/gvfs/gvfsd-trash --sp
test	913	739	0	10:51	?	00:00:00	/usr/lib/gvfs/gvfsd-metadata
test	918	823	0	10:51	?	00:00:00	/usr/lib/x86_64-linux-gnu/xfce
test	919	823	0	10:51	?	00:00:00	/usr/lib/x86_64-linux-gnu/xfce
test	923	1	0	10:51	?	00:00:02	xfce4-terminal
test	927	923	0	10:51	pts/0	00:00:00	bash
test	1036	1	0	11:18	?	00:00:02	mousepad /home/test/Documents/
test	1058	923	0	11:22	pts/1	00:00:00	bash
root	1132	2	0	11:37	?	00:00:00	[kworker/0:0-ata_sff]
root	1134	2	0	11:43	?	00:00:00	[kworker/0:2-ata_sff]
test	1140	739	0	11:43	?	00:00:00	/usr/lib/x86_64-linux-gnu/tumb
root	1149	2	0	11:43	?	00:00:00	[kworker/u2:0-events_unbound]
test	1153	927	0	11:44	pts/0	00:00:00	vi
test	1154	927	0	11:44	pts/0	00:00:00	python3 prog.py
test	1155	1058	0	11:44	pts/1	00:00:00	ps -aef

On rappelle que :

- l'UID est l'identifiant de l'utilisateur propriétaire du processus ;
- le PID est l'identifiant du processus ;
- le PPID est l'identifiant du processus parent ;
- C indique l'utilisation processeur ;
- STIME est l'heure de démarrage du processus ;
- TTY est le nom du terminal de commande auquel le processus est attaché ;
- TIME est la durée d'utilisation du processeur par le processus ;
- CMD le nom de commande utilisé pour démarrer le processus.

- (a) Donner le PID du parent du processus démarré par la commande `vi`.
- (b) Donner le PID d'un processus enfant du processus démarré par la commande `xfce4-terminal`.
- (c) Citer le PID de deux processus qui ont le même parent.
- (d) Parmi tous les processus affichés, citer le PID des deux qui ont consommé le plus de temps du processeur.

3. On considère les trois processus P_1 , P_2 et P_3 , tous soumis à l'instant 0 dans l'ordre 1, 2, 3 :

Nom du processus	Durée d'exécution en unité de temps	Ordre de soumission
P_1	3	1
P_2	1	2
P_3	4	3

- (a) Dans cette question, on considère que les processus sont exécutés de manière concurrente selon la politique du tourniquet : le temps est découpé en tranches nommées *quantums de temps*.

Les processus prêts à être exécutés sont placés dans une file d'attente selon leur ordre de soumission.

Lorsqu'un processus est élu, il s'exécute au plus durant un quantum de temps. Si le processus n'a pas terminé son exécution à l'issu du quantum de temps, il réintègre la file des processus prêts (côté entrée). Un autre processus, désormais en tête de la file (côté sortie) des processus prêts, est alors à son tour élu pour une durée égale à un quantum de temps maximum.

entrée → $P_3 \quad P_2 \quad P_1$ → sortie

Reproduire le tableau ci-dessous sur la copie et indiquer dans chacune des cases le processus exécuté à chaque cycle. Le quantum correspond à une unité de temps.

P1								
0	1	2	3	4	5	6	7	8

- (b) Dans cette question, on considère que les processus sont exécutés en appliquant la politique du « plus court d'abord » : les processus sont exécutés complètement dans l'ordre croissant de leurs temps d'exécution, le plus court étant exécuté en premier.

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

0	1	2	3	4	5	6	7	8

4. On considère trois ressources R_1 , R_2 et R_3 et trois processus P_1 , P_2 et P_3 dont les files d'exécution des instructions élémentaires sont indiquées ci-dessous :

Processus P_1
Demande R_1
Demande R_2
Libère R_1
Libère R_2

Processus P_2
Demande R_2
Demande R_3
Libère R_2
Libère R_3

Processus P_3
Demande R_3
Demande R_1
Libère R_3
Libère R_1

- (a) Rappeler les différents états d'un processus et expliquer pourquoi il y a ici risque d'interblocage, en proposant un ordre d'exécution des instructions élémentaires le provoquant.
- (b) Proposer un ordre d'exécution des instructions élémentaires sans interblocage.