Commandes de base

Question 3A

ps permet d'afficher une snapshot des processus courants

pstree affiche un arbre des processus

kill permet de fermer un processus (de le passer en mode T)

Question 3B

l'argument -a selectionne tout les processus et -ux permet de les trier pour les traiter par PID

- USER indique l'utilisateur à qui appartient le processus
- PID est son identifiant unique
- %CPU est son utilisation sur le processeur en % d'utilisation
- %MEM et le % d'utilitation de la memoire vive par le processus
- TTY est la session de shell depuis laquelle on lance le processus
- STAT est l'etat du processus
- START est sa date d'execution
- TIME est son temps d'execution

Question 3C

Ces colonnes permettent d'etablir une priorité entre les processus

PR represente la priorité interne du kernel pour le processus donner et NI est la "nice value" plus elle est elevé plus elle laissera les autres processus prendre la priorité. C'est la gestion des priorités coté utilisateur

Question 3D

```
ps -o pri -p <PID>
```

Question 3E

la commande nice permet de demarrer un processus avec une priorité differente preciser en option

si le processus est deja en cours on peut utiliser la commande renice

Question 3F

kill -3 correspond à SIGQUIT

kill -9 correspond à SIGKILL

dans la liste on retrouve :

```
SIGHUP          1          1       1       1
SIGINT          2          2       2       2
SIGQUIT         3          3       3       3
SIGILL          4          4       4       4
SIGTRAP         5          5       5       5
SIGABRT         6          6       6       6
SIGIOT          6          6       6       6
SIGBUS          7         10      10      10
SIGEMT          -          7       7       -
SIGFPE          8          8       8       8
SIGKILL         9          9       9       9
SIGSTOP         19          19       19       19
```

Question 3G

Les signaux SIGKILL et SIGSTOP ont une particularité, ils ne sont pas envoyés au processus mais directement adressés au système qui se chargera de stopper ou d'arrêter le processus

Question 3H

Nohup est une commande Unix permettant de lancer un processus qui restera actif même après une deconnexion de l'utilisateur ayant initialisé le processus

Question 3I

On utilise CTRL+Z pour passer un processus en pause, la commande bg pour le faire executer en arriere plan et fg pour le repasser au premier plan

Gestion des processus

Question 4A

On peut utilise la commande bg pour passer le processus en background ou utiliser l'esperluette quand on lance le programme

Question 4B

le processus continue en arriere plan malgrés le clear du terminal

Question 4C

on peut utiliser CTRL+Z pour mettre en pause un processus, on peut aussi utiliser la commande kill -19

Question 4D

on utilise la commande fg "UID"

Question 4E

On peut utiliser la commande kill -9 ou faire un CTRL+C

Question 4F

numero de tache sert pour du multithreading car la variable est partager en toutes les taches dans le meme thread et numero de processus pour le multiprocessing et differencier les differents processus.

Question 4G

le processus est lié au terminal car il s'interrompt quand on sort de notre session. On peut le verifier car lorsque qu'on se reconnecte au terminal le processus à disparu

Question 4H

nohup permet de lancer un processus qui n'est pas liée à une session utilisateur

disown permet de modifier un processus deja actif pour le detacher de la session auquel il est rattacher

Question 4I

```
nice ./randomgenerator -"numero de la priorité"
```

Question 4J

```
renice -"numero de la priorité" "PID"
```

Redirection de flux

Question 5A

Crée un fichier rand.txt et ecrit le resultat de la commande à l'interieur. Si il existe deja un fichier, on l'ecrase et on reecrit dedans

Question 5B

meme chose que la 5A mais n'ecrase pas le fichier si il est deja crée, on ecrit seulement à la suite.

Question 5C

demande à l'utilisateur le nombre de nombres aleatoires qu'il souhaite generer

Question 5D

permet de preciser le parametre dans un string qui sera directement apellé lors de la commande suivante

Question 5E

On crée un fichier response.txt et on ecrit 20 à l'interieur

Question 5F

lit le contenu du fichier response.txt et le precise directement à l'argument i

Question 5G

Meme chose que 5D mais trie les valeurs par ordre croissant et les affiches seulement lorsque le programme à fini de tourner

Question 5H

stdin est l'entrée standard

stdout est la sortie standard

stderr est la sortie d'erreur standard

Question 5I

Genere 50 nombres aleatoires et crée un fichier nommé data.txt pour ecrire ces nombres à l'interieur. On precise & car on lance le processus en background

Question 5J

affiche le contenu du fichier data.txt trié par ordre croissant

Question 5K

```
case 'n':
            if (!i_isset) {
                repet = atoi(optarg);
                n_isset = true;
            } else {
                errflag = true;
            }
            break;
```

Création d'un processus en C