ATELIER : Gestion des process et redirection

Process courant

1. Ouvrez une session et affichez votre numéro de process courant.

2. Créez un sous-shell en lançant **bash** (ou **ksh** selon votre environnement). Quel est le PID de ce sous-shell ? Est-il différent de votre process obtenu à votre connexion ?
3. Lancez la commande **ls -lR / > outfile 2> errfile &** puis affichez la liste des process lancés depuis votre terminal. Repérez la ligne de la commande **ls**.
4. Terminez votre shell enfant. Qu'est-ce qui se passerait si vous tapiez de nouveau exit ?
   ⇨ Un deuxième exit fait terminer le shell parent.

Contrôle des process lancés en tâche de fond.

5. Avec vi créez un script de commandes shell nommé sctest qui contient les lignes suivantes:
6. Rendez-le exécutable. Puis lancez en avant plan la commande **./sctest**
7. La pause de 30 secondes permet de suspendre le job. Suspendez le job que vous venez de lancer.
8. Affichez la liste des jobs que vous avez lancés sur le système et relancez en tâche de fond celui que vous avez suspendu ci-dessus.
9. Ramenez le job en avant-plan.
10. Lancez le script **sctest** avec **nohup** et en tâche de fond. Notez le numéro de job, son PID, puis déconnectez-vous.
11. Connectez-vous de nouveau et vérifiez que le process est toujours en cours d'exécution.
12. Une fois le process terminé, affichez le fichier qui contient le texte de sortie outfile. (si vous n'avez pas explicitement redirigé la sortie, alors par défaut le système crée un fichier qui se nomme nohup.out dans le répertoire d'où a été lancée la commande).

Terminer un processus

13. Lancez en tâche de fond, avec redirection des sorties, la commande **ls -lR /**. (*Cette commande met du temps avant de se terminer.)* Notez le PID du process.
14. Si vous n'avez pas pu noter le PID après avoir lancé la commande en tâche de fond, comment
15. Une fois le PID connu, tuez le process. Vérifiez qu'il a bien disparu.

Les redirections

16. Avec la commande **cat** et le mécanisme de redirection, créez un fichier de texte nommé **exemple** qui contient quelques lignes de texte. Utilisez **<Ctrl-d>** au début d'une nouvelle ligne pour mettre fin à votre saisie et retourner à l'invite shell $. Affichez le contenu du fichier pour vérifier.

17. Avec la commande **cat** ajoutez plusieurs lignes de texte au fichier **exemple**. Vérifiez vos modifications en affichant le contenu de ce fichier.

Pipes et filtres

18. Affichez le contenu de votre répertoire courant avec la commande ls. Notez le nombre de fichiers.

19. Listez de nouveau le contenu de votre répertoire courant, mais cette fois redirigez la sortie vers un fichier nommé **temp**

20. Utilisez la commande appropriée pour compter le nombre de mots du fichier temp. Est-ce le même nombre qu'à l'étape 18 ? Si non pourquoi ?
    ⇨ Nombre de mots affichés = Nombre de fichiers de l'étape 18 **+ 1 (présence du fichier temp)**

21. Affichez le contenu du fichier temp. Supprimez-le.

22. Utilisez un pipe '|'' pour compter le nombre de fichiers du répertoire courant. C'est bien le résultat attendu ? C'est le même nombre qu'à l'étape 18 ?
    ⇨ C'est le même résultat qu'à l'étape 18 puisqu'on a supprimé le fichier temp.

23. Reprenez la ligne de commande lancée à l'étape 21, mais cette fois insérez la commande **tee** entre les deux commandes pour écrire dans un fichier nommé **exemple2**. Est-ce que le nombre de fichier est affiché ?

24. Vérifiez que le fichier **exemple2** contient bien ce qui est prévu.

25. Listez le contenu du répertoire courant en ordre inversé.
    - Envoyez le résultat dans un fichier **exemple3**, et aussi vers une commande pour compter le nombre de mots de la liste inversée.
    - Ajoutez ce nombre au fichier **exemple3**. N'oubliez pas d'utiliser la redirection en mode ajout, sinon vous pourriez avoir des résultats inattendus.

26. Dans le dossier **/dev** il y a un fichier spécial qui représente votre terminal. Affichez le nom du fichier qui est associé à votre terminal. Il se présente sous la forme **tty0**, **lft0** ou **pts/x** . Répétez la commande de l'étape précédente, mais avec deux changements :
    - Plutôt que d'utiliser le fichier ordinaire **exemple3**, la commande tee envoie le résultat à votre écran (**/dev/<nom de votre terminal>**)
    - Ne renvoyez pas le résultat de la commande **wc** vers le fichier **exemple3** de façon à ce que le comptage soit envoyé à l'écran.

Exercice d'évaluation

1. Afficher dans un fichier « **ps_tri.txt** » la liste des processus triés par ordre alphabétique

2. En utilisant deux pipes et trois commandes différentes, compter le nombre de répertoires qui se trouvent dans le répertoire **/bin**

3. Lister dans un fichier « **ps_user.txt** » tous les processus appelés « **apache** » et qui appartiennent à un utilisateur dont le nom commence par **r** ou **R**

4. Ecrire la commande qui permet de compter le nombre de mots de toutes les lignes entre **3** et **20** d'un fichier **« f1»** et stocker ce résultat dans un fichier **« f2 »** qui existe déjà

5. Lister dans un fichier « **user_a .txt**» les 5 premiers utilisateurs, créés sur votre machine, dont les noms commencent par la lettre « **c** »

6. Ecrire dans un fichier « **upper_user.txt** », tous les noms d'utilisateurs créés sur votre machine en majuscule

Questions diverses

Q1- Quelle affirmation est fausse ?

Le seul processus n'ayant pas de père a le PID 1	Un processus peut avoir plusieurs processus pères en même temps
Un processus peut avoir zéro ou plusieurs processus fils	Un processus dont le processus père est mort meurt à son tour ou a comme nouveau père le processus identifié par le PID 1

Q2- Quelles commandes permettent de visualiser l'ensemble des processus en cours d'exécution sur le système ?

$ ps aux	$ ps -ef	$ top	$ pstree	$ lsps

Q3- Quel signal a pour effet de mettre fin systématiquement à un processus ?

SIGHUP	SIGINT	2	SIGKILL	SIGTERM

Q4- Comment lancer une commande en arrière-plan ?

$ commande	$ bg commande	$ kill SIGBGR commande	$ commande &

Q5- Que réalise la commande suivante ? grep ^F contacts_trie | grep s$ | head -5 2>>log.txt

A. Afficher toutes les lignes de « contacts_trie » qui contiennent F ou s
B. Afficher les 5 premières lignes de « contacts_trie » qui contiennent F ou s dans « log.txt »
C. Afficher toutes les lignes de « contacts_trie » qui commencent par F et se terminent par s dans « log.txt »
D. Afficher les 5 premières lignes de « contacts_trie » qui commencent par F et se terminent par s et rediriger les messages d'erreur dans « log.txt »

Q6- Que réalise la commande suivante ? Head –n 6 test.txt | tail –c 2 > result.txt

A. Extraire les 2 dernières lignes des 6 premières lignes de « test.txt » dans « result.txt »

B. Extraire les 2 derniers caractères de chacune des 6 premières lignes de « test.txt » et rediriger les messages d'erreur dans « result.txt »

C. Extraire les 2 derniers caractères des 6 premières lignes de « test.txt » et afficher le résultat dans « result.txt »