TP – Bash

Dans ce TP, nous essayerons de tout faire sans ouvrir d'autre application que le terminal. Quand vous ne savez pas quelque chose, servez-vous du manuel (commande `man`), qui est très complet (et difficile à lire au début) ! Pour éditer des fichiers, nous utiliserons `vim`, qui est très puissant (et difficile à utiliser au début) !

Exercice 1. *Premiers pas*

Lancer un terminal.

`pwd`

1. Dans quel dossier vous trouvez-vous au lancement de cet interprète de commandes Bash ?
Note : la commande `cd` vous permet, à tout moment, de revenir à votre `home` (`cd` = `cd ~`).

✎ Quelle est la différence entre un langage interprété (Bash, Python, *etc*) et un langage compilé (C, Java, *etc*) ?

`mkdir touch`

2. Où bon vous semble, créer un nouveau dossier `TP01`, se placer dedans, y créer un fichier nommé `fichier1`, afficher le contenu détaillé et complet du dossier `TP01`.

✎ Que désignent les dossiers `.` et `..` ? Où se trouve le répertoire désigné par `/` ?

`man`

3. Lire le manuel de `ls` pour trouver l'option qui colore son résultat.

`for seq touch`

4. Créer 99 fichiers nommés `fichier2` à `fichier100`.

`for echo >`

5. Créer un fichier nommé `fichier0` qui contient 100 fois votre prénom (une occurrence par ligne).

`cat more`

6. Afficher son contenu de façon non-paginée puis paginée.

`wc`

7. Afficher le nombre de caractères de `fichier0`.

`cat sed`

8. Afficher son contenu en remplaçant votre prénom par `toto`.

`cat sed`

9. Ajouter le résultat de la commande précédente à la fin de votre fichier (on peut passer par un fichier temporaire).

`ls`

10. Afficher la liste des fichier qui contiennent un `2` dans leur nom.

`ls tr cut`

11. Afficher les tailles des fichiers qui contiennent un `2` dans leur nom.

`ls tr cut`

12. Afficher les tailles des fichiers dont le nom ne termine pas par un chiffre entre `5` et `9`.

`cat grep`

13. Afficher les lignes de `fichier0` qui contiennent la lettre `t`.

14. Se placer dans votre `home`.

`who wc`

15. Afficher le nombre d'utilisateurs connectés.

`ls grep`

16. Lister uniquement les répertoires.

`df`

17. Afficher les détails d'occupation mémoire de votre répertoire `home` dans une unité compréhensible.

18. Retourner dans le dossier `TP01`.

`rm`

19. Supprimer les fichiers dont le nom comporte un `5`. **Note :** Attention avec la commande `rm`, on a vite fait de faire une bêtise.

`echo` **20.** Afficher `Bonjour` à l'écran.

`at` **21.** Afficher `Bonjour` à l'écran dans deux minutes.

`uptime` **22.** Depuis combien de temps votre machine est-elle allumée ?

`cal` **23.** Quel jour de la semaine tombera votre prochain anniversaire ?

Exercice 2. *`vim`*

`vim` a différents modes : insertion et visualisation sont les principaux. A la base, vous n'êtes dans aucun mode. `i` : passer en mode insertion ; `v` : passer en mode visualisation ; touche `sec` : sortir du mode courant.

`#!/bin/sh` **1.** Ecrire un script nommé `script1-bonjour.sh` qui affiche `Bonjour` à l'écran.

`chmod` **2.** Donner les droits d'exécution à l'utilisateur et l'exécuter.

3. Modifier le pour qu'il prenne en argument un `prénom` et affiche `Bonjour prénom`. Tester.

mode `V` **4.** Copier cette ligne pour afficher deux fois le message. Tester.

5. S'il n'y a pas d'argument, afficher le message suivant : `Erreur : un argument est attendu`. Tester.

6. Créer un fichier `ceci_est_une_poussiere`.

7. Ecrire un script nommé `script2-menage.sh` qui supprime tous les fichiers dont le nom comporte `poussiere` et qui ont été créés il y a plus de 10 minutes.

`date`

`at` **8.** Programmer le lancement de `script2-menage.sh` dans 10 minutes.

9. Ecrire un script `script3-affichage.sh` comprenant 30 lignes identiques, chacune affichant le mot `Hello`.

10. Remplacer chaque occurrence de `Hello` par `Bonjour` (avec `vim`).

mode `ctrl-v` **11.** Rajouter un symbole `#` de commentaire aux lignes 3 à 22.

12. Ecrire un script `compteur.sh` qui affiche séquentiellement les entiers à partir de 1.

`let`

`ctrl-c` **13.** Tester.

Exercice 3. *`~/.bashrc`*

`~/.bashrc` est le fichier de configuration du Bash, il est exécuté à chaque lancement d'un terminal.

1. Retourner dans votre `home`, afficher son contenu, puis son contenu complet détaillé.

✎ Que désignent les dossiers et fichiers dont le nom commence par un `.` ?

`alias` **2.** Ajouter l'alias `ll` pour `ls -la` dans votre `.bashrc`.

✎ À quoi servent les options `-l` et `-a` de la commande `ls` ?

3. Relancer le terminal et tester la commande `ll`.

4. Si vous aimez les couleurs, rajoutez-les !

5. Ajouter un alias à votre `.bashrc` pour que la commande `cd $TPOS` vous amène dans le dossier créer à l'exercice 1.

Nous allons faire en sorte que vos scripts soient accessibles de la même façon que les commandes usuelles. Lorsque vous tapez une commande, l'interprète cherche si la commande existe. Pour cela, la variable d'environnement `PATH` contient la liste des dossiers où sont placés les exécutables.

6. Créer (s'il n'existe pas) le dossier `~/bin`.
7. Ecrire un script `~/bin/mon_test` qui affiche `ok`.

✎ Que contient la variable `PATH` ?

`export`

8. Ajouter (en respectant les conventions) le chemin vers `~/bin` à la variable `PATH`.
9. Tester la commande `mon_test`.
10. Ajouter cette ligne à votre `.bashrc`.
11. Relancer le terminal et tester, puis supprimer le fichier `~/bin/mon_test`.

✎ À l'avenir, comment faire pour qu'un script que vous écrivez soit accessible facilement pour vous, quel que soit votre dossier courant ?

12. Faire cela pour le script `compteur.sh`.

Exercice 4. *`~/.vimrc`*

Choisir un thème pour `vim` sur le site `www.vim.org` ⟩ Scripts ⟩ Browse all ⟩ Rechercher "color theme".

1. Le placer dans le dossier `~/.vim/colors`.
2. Ajouter la ligne `colorscheme nom_du_theme` à votre `~/.vimrc`.
3. Tester (la coloration syntaxique est fonction de l'extension des fichiers ouverts).

Exercice 5. *terminator*

1. Comprendre le résultat de la commande `top`.

✎ Comment afficher les processus par ordre d'utilisation de la CPU ?

2. Comprendre le résultat de la commande `ps`.

✎ Comment afficher plus de processus ?

✎ Comment afficher l'utilisation de la CPU avec `ps` ?

3. Dans un second terminal, lancer votre script `compteur.sh`.
4. Dans le premier terminal, regarder le taux d'utilisation de la CPU de ce processeur.

`kill`

5. Depuis le premier terminal, tuer le processus d'exécution de `compteur.sh`.
6. Ecrire un script `urgence.sh` qui tue tous les processus utilisant plus de 75% de la CPU.
   Note : Si vous avez peur de l'exécuter par mégarde, ne le placez pas dans `~/bin`.

Exercice 6. *pour une fois…*

1. Ecrire un script qui vous sera utile.