PREMIERS SCRIPTS

Ouvrir un client `gnome-terminal` et taper la commande `/bin/bash` pour utiliser le shell `bash` (votre shell par défaut est un `csh`).

La commande interne(builtin) `set`

Une commande interne est un outil fourni à l'utilisateur du shell pour lui permettre de configurer, administrer et utiliser plus efficacement l'interpréteur de commande. Ici nous allons utiliser la commande interne `set` qui permet entre autres choses de visualiser les variables définies dans le shell.

1. Tapez la commande `set` et analyser le résultat.

Nous allons nous intéresser plus particulièrement à la variable `SHELLOPTS` qui contient une liste de flags qui influence le fonctionnement de votre shell.

1. Affichez la valeur de cette variable

   Chaque un mot clé séparé des autres par le symbole ':' représente un flag qui est valide. Nous allons modifier certain de ces flags.

2. Taper la commande `set -f` puis la commande `ls *`. Que se passe-t-il ?

3. Taper la commande `set +f` puis la commande `ls *`. Que se passe-t-il ?

   On aurait le même comportement avec la commande `set -o noglob` et `set +o noglob`. Essayez !

   Nous allons étudier le comportement des flags `noclobber` et `ignoreeof`

4. Taper la commande `set -o ignoreeof` et taper ensuite la séquence de touches Ctrl-D. Que ce passe-t-il ?

   Si vous tapez la séquence de touches Ctrl-D après la commande `set +o ignoreeof` l'interpréteur de commandes se termine.

5. La commande `set -C` et `set +C` ou la commande `set -o noclobber` et `set +o noclobber`.

   Le flag `noclobber` permet de gérer l'autorisation de la surcharge d'un fichier en cas de redirection, par exemple la commande `ls > /tmp/toto` sera exécutée si `n/tmp/toto` n'existe pas ou si le flag `noclobber` est faux et ne sera exécutée si `n/tmp/toto` existe et le flag `noclobber` est vrai.

6. Que faut-il faire pour que le flag `ignoreeof` soit toujours valide ?

La commande `set` permet aussi d'assigner une valeur aux variables de position `$1`,`$2`,`$3`,...,`$N`. La variable `$#` prend la valeur N.

1. Donner un nom de fruit à chacune des 5 (par exemple) premières variables de position et afficher les ainsi que le nombre de variables de position définies.

Variables locales

Ouvrir un deuxième client `gnome-terminal` qui utilise le shell `bash`. Dans l'un des deux, créer la variable locale de nom `prenom` et de valeur votre prénom. Vérifier dans l'autre fenêtre que la variable `prenom` n'est pas connue. Dans la première fenêtre, empiler un nouveau shell en tapant `/bin/bash` et essayer de visualiser la variable locale `prenom`.

1. Que se passe-t-il ?

Variables d'environnement (variable globale)

Définir une variable d'environnement `prenom` contenant votre prénom.

1. Visualiser la.
2. Que se passe-t-il si vous essayez de la visualiser dans l'autre fenêtre shell ?
3. Dans la première fenêtre, empiler un nouveau shell en tapant `bash` et essayer de visualiser la variable d'environnement `prenom`.
4. Que se passe-t-il ?

La variable d'environnement `PATH` contient la liste des chemins dans lesquels le shell va rechercher les programmes que l'utilisateur veut exécuter.

1. afficher la valeur de la variable d'environnement `PATH`.
2. Créer un répertoire bin dans votre répertoire de login (~) et dans ce répertoire créer un fichier `qui.sh` qui contient le code suivant :

```
#!/bin/bash
echo "voici la liste des utilisateurs connectés sur la machine :"
who | awk '{print $1}'
```

3. Donner les droits d'exécution pour tous au fichier `qui.sh`
4. Que faut-il faire pour que l'on puisse exécuter le fichier `qui.sh` en utilisant la commande `qui.sh` quelque soit le répertoire de travail courant ?

Les simples et doubles quotes et les back quotes

Les quotes simples `'` doivent être ouvertes et refermées sur la même ligne. Les doubles quotes doivent également refermées sur la même ligne, sauf si on utilise le caractère backslash (\) pour prolonger la ligne.

1. Essayez la commande `echo 'Il faut refermer la quote`.
2. Essayez ensuite la commande `echo 'Il faut refermer la quote'`.
3. Essayez la commande `echo "La date est `date`"`.
4. Essayez la commande `echo "Mon repertoire de travail est : $HOME"`.
5. Essayez maintenant la commande `echo 'Mon repertoire de travail est : $HOME'`.
6. Comment pouvez-vous créer un fichier de nom `?vilain?` ?
7. Comment pouvez-vous l'effacer ?
8. Trouvez une autre manière de faire.

Premiers scripts

1. Écrire un script shell `bash` qui affiche la phrase "Aujourd'hui `date_et_heure`, l'utilisateur `votre_login` ➥ est connecté".
2. Écrire une autre commande qui affiche toujours la date mais donne tous les utilisateurs connectés à la machine. On utilisera la commande `who` qui permet d'obtenir cette information.