



TP shell (2) : find, grep

EXERCICE 1 : commande `supprimeDouble`.

Ecrire un script `supprimeDouble` de supprimer tous les fichiers existant en plusieurs exemplaires dans un répertoire donné; on ne conservera qu'un seul exemplaire de chaque fichier.

Indication : utiliser la commande `cmp` et pour tester votre script, et évitez `rm -i` dans un premier temps (testez avec `echo "fichier à supprimer"`, ce qui est plus prudent...)

EXERCICE 2 : la commande `find`.

1. Syntaxe de la commande `find` :

```
find chemins expression
```

La commande `find` effectue une exploration récursive dans chaque chemin mentionné pour chercher les fichiers qui répondent à l'expression booléenne indiquée.

Par exemple, la commande :
`find . -name '*.txt' -print`
affiche (`print`) les noms des fichiers dont le nom (`name`) concorde avec `*.txt`

2. La variable `$HOME` contient votre *home directory* (répertoire d'accueil). Essayez :

```
echo $HOME  
find $HOME -name '*.txt' -print
```

3. L'opérateur `-o` représente l'opérateur logique ou :

```
find $HOME -name '*.txt' -o -name 'd*' -print
```

Attention, dans cet exemple, l'action `print` est seulement effectuée sur les noms des fichiers concordant avec le deuxième nom.

Complément : si un fichier a un nom qui répond aux deux sélections, ici `*.txt` et `d*`, seule la première sélection a son action exécutée.

On peut alors :

- (a) soit donner une action à chaque définition de nom de fichier :

```
find $HOME -name '*.txt' -print -o -name 'd*' -print
```

(b) soit factoriser le traitement :

```
find $HOME \( -name '*.txt' -o -name 'd*' \) -print
```

Les caractères () ont une signification particulière pour le shell. Pour que ces caractères soient envoyés à la commande `find` qui en a besoin pour comprendre la factorisation de la commande `print`, il faut les précéder chacun par le caractère `\`. Le caractère `\` demande simplement au shell d'ignorer le caractère qui suit.

Attention, il faut placer un `_` devant la chaîne `-name` sinon, il se produit une erreur : `segmentation fault` ou `find: invalid predicate '(-name'`

4. D'autres actions peuvent être exécutées sur les fichiers trouvés, par exemple :

```
find $HOME -name '*.txt' -exec ls -l {} \;
```

où :

- on demande l'exécution (`exec`) d'une et une seule commande Unix particulière, ici `ls -l` qui affiche les caractéristiques d'un fichier ; cette commande ne doit comporter ni `;` ni `|`
- les accolades `{ }` permettent de passer en argument de la commande Unix le nom de fichier courant (trouvé par la commande `find`) ;
- il faut terminer la commande demandée par `exec` par un `;` De plus, ce caractère `;` doit être précédé par le caractère `\`, afin que le `;` ne soit pas interprété par le *shell* comme un séparateur entre deux commandes.

Remarques :

- bien mettre des espaces `_` devant chaque entité utilisée dans la commande `find`, y compris devant `\` ;
`find_ $HOME_ -name_ '*.txt'_ -exec_ ls_ -l_ {}_ \;`
- il est aussi correct de spécifier les noms des fichiers entre guillemets à la place des quotes,
- les commandes spécifiées dans la commande `exec` sont interprétées par *sh* ; aussi il n'est par exemple pas possible d'utiliser des commandes aliasées.

EXERCICE 3 : la commande `menage`.

Ecrire une commande `menage` qui nettoie les répertoires de l'utilisateur, récursivement à partir de son *home* et utilise la commande `find`. Par exemple, cette commande supprime les fichiers `core`, `*~`, `###` de tous vos répertoires avec demande de confirmation avant chaque destruction.

Les noms des fichiers à détruire ne sont pas passés en argument de la commande, mais figurent en "toute lettre" dans la commande.

Vous testerez votre commande de `menage` en écrivant dans un premier temps vos scripts avec `ls` et non pas `rm -i` afin d'éviter des destructions involontaires de fichiers...

EXERCICE 4 : Utilisation de `grep` - recherche de motifs

Recopiez chez vous le fichier `test_grep.txt` et cherchez les commandes en ligne qui effectuent les recherches demandées.

Remarque : Les lignes affichées par la commande `grep` seront précédées par le numéro de la question à laquelle elles répondent. Exactement 3 lignes par question doivent être affichées, éventuellement accompagnées par des réponses à d'autres questions.

Pensez à utiliser `man grep`...

1. recherche de toutes les occurrences du mot `oscilloscope`

2. recherche de toutes les occurrences de la chaîne `L'oscilloscope` avec affichage des numéros de ligne ;

3. recherche de toutes les occurrences de la chaîne `theta` en fin de ligne ;

4. recherche de toutes les occurrences du caractère `x` suivi d'un caractère espace, suivi d'au moins un caractère numérique ;

5. recherche de toutes les occurrences de la chaîne `-␣Régler`

6. recherche de toutes les lignes contenant indistinctement la chaîne `méthode` en minuscules ou en majuscules ;

7. recherche de toutes les occurrences du caractère `f`, suivi d'une parenthèse ouvrante (`,` suivi d'un nombre quelconque de caractères quelconques, suivi d'une parenthèse fermante `)`

8. recherche des mots `une trace` séparés par au moins un caractère espace ;

9. recherche de toutes les lignes contenant au moins 5 caractères numériques consécutifs ;

10. recherche de toutes les lignes contenant au moins 5 caractères numériques, quelle que soit leur disposition dans la ligne ;

11. recherche de toutes les lignes contenant la chaîne `omega` ou la chaîne `phi`

12. recherche de toutes les lignes contenant la chaîne `alpha` et la chaîne `beta` dans cet ordre;

13. recherche de toutes les lignes contenant la chaîne `delta` et la chaîne `gamma` dans un ordre quelconque ;

14. recherche de toutes les occurrences des lignes contenant exactement 12 chiffres consécutifs;

15. recherche de toutes les lignes contenant exactement 2 fois la chaîne `ellipse`
