TD1 · Langages de commande

NUnix · L3 MIAGE · Université de Rennes

Elisabeth Bouriah et Louis Ledoux 2026–2027

Exercice 1 : caractères spéciaux

Lister les entrées de `/usr/bin` dont le nom :

1. Commence par `m`.
2. Commence par `m` et comporte exactement trois caractères.
3. Commence par `m` et comporte au moins trois caractères.
4. Commence par `m` et comporte une extension (un suffixe après un point).

Exercice 2 : redirections

On considère l'arborescence suivante. Le répertoire courant est `user2`.

```
practice/
├── user1/
│   ├── f1
│   ├── f2
│   └── f3
└── user2/
    └── f4
```

En utilisant `cat` et les redirections :

1. Créer `f5` contenant la ligne `5 2 3 -1`.
2. Créer `f6` contenant `f5` suivi de `f4`.
3. Ajouter le fichier `f1` de `user1` à la fin de `f6`.
4. Copier le fichier `f2` de `user1` dans un nouveau fichier `f7` de `user2`.

Exercice 3 : un script avec trois noms de fichiers

1. Écrire un script shell `execution`, appelé avec trois noms de fichiers : `n1 n2 n3`. Il doit effectuer les traitements suivants :

Entrée	Programme	Sortie
Fichier `n1`	`p1`	Fichier `n1.r1`
Fichier `n2`	`p1`	Fichier `n2.r1`
`n2.r1` suivi de `n1.r1`	`p2`	Fichier `n3`

   Les programmes `p1` et `p2` lisent l'entrée standard et écrivent sur la sortie standard. On les suppose exécutables et présents dans le répertoire courant.
2. Avec l'arborescence de l'exercice 2, que se passe-t-il si `user1` lance cette commande depuis son répertoire ? Identifier les fichiers lus et l'emplacement de chaque fichier produit.
   `./execution f1 ../user2/f4 f5`
3. Écrire une version interactive qui demande `n1`, `n2` et `n3` à l'utilisateur au lieu de les recevoir comme arguments.

Exercice 4 : arguments de la commande

Écrire un script shell `argument` qui affiche ses arguments un à un :

```
$ ./argument un deux trois
3 arguments :
 1 - un
 2 - deux
 3 - trois
```

Utiliser `expr` pour mettre à jour le compteur. Cette commande évalue des expressions arithmétiques et logiques et affiche leur résultat ; par exemple, `expr 1 + 2` affiche `3`.

Exercice 5 : recherche d'un motif

1. Écrire `mysearch`, appelé sous la forme `./mysearch motif fichier`. Il affiche `fichier contient le motif` ou `fichier ne contient pas le motif`.
2. Écrire `mysearch2 fichier`. Il demande successivement les motifs à rechercher dans le fichier, jusqu'à ce que l'utilisateur saisisse `end`.
3. Écrire `mysearch3 f1 ... fn`. Pour chaque motif saisi, rechercher dans tous les fichiers passés en arguments. Arrêter lorsque l'utilisateur saisit `end`.

Vous pouvez rediriger les affichages indésirables vers `/dev/null`.

Exercice 6 : la commande find

Expliquer l'effet de chaque commande. Les lire avant de les exécuter :

```
find ./ -name tp2 -exec {} < f_in > {}.out \;
find /var/log -exec grep "www.univ-rennes1.fr" {} \;
find . -name "rc.conf" -exec chmod o+r {} \;
find /usr/local -type f -name "*.html"
```