

TRAVAUX PRATIQUES 1

Introduction aux langages de commande Linux

L'objectif de ce TP est de voir des commandes utiles disponibles sur les systèmes d'exploitation de type UNIX pour travailler et interagir avec celui-ci. Ce système apparu à partir de 1969 est multitâche et multi-utilisateur. L'un des composants de ce système est l'interpréteur de commandes (ou shell) permettant d'exécuter des programmes utilitaires.

Toutes les commandes utilisées dans ce TP sont à taper dans une fenêtre shell qui les lance et les exécute.

Une fois votre session ouverte sous un système d'exploitation de type UNIX, ouvrez un terminal pour entrer les commandes décrites dans l'énoncé. Dans la suite, le prompt du terminal sera symbolisé par « \$> ».

I. Informations générales

Nous allons tout d'abord étudier quelques commandes générales permettant d'obtenir des informations utiles.

1) Manuel

Un système d'exploitation fournit un ensemble de commandes possédant chacune un certain nombre d'options pour paramétrer les informations retournées. Il est difficile de connaître toutes les options pour chaque commande. Ainsi, la commande **man** (pour *manuel*) fournit un descriptif détaillé sur l'utilisation d'une commande système passée en paramètre. Pour sortir du manuel appuyez sur la touche **q** du clavier. Il ne faut donc pas hésiter à consulter cette documentation qui est un véritable aide-mémoire.

Question 1 – Entrez dans un terminal la commande donnée ci-dessous. Expliquez ce que réalise la commande `clear`.

```
$> man clear
```

Question 2 – Tapez la commande `clear` dans un terminal. Indiquez quel est le résultat obtenu par cette commande.

Correction : cette commande permet de réinitialiser l'affichage du shell.

2) Nom utilisateur

Les systèmes d'exploitation que nous utilisons aujourd'hui sont des systèmes dits multi-utilisateurs, c'est-à-dire qu'à tout moment plusieurs utilisateurs peuvent lancer des programmes, qui auront des droits d'exécutions hérités des droits accordés à l'utilisateur ayant initié son exécution. Il est possible de connaître le nom de l'utilisateur courant sous lequel vous êtes identifié. La commande **whoami** retourne le nom de l'utilisateur sous lequel vous êtes actuellement identifié.

Question 3 – Entrez dans un terminal la commande donnée ci-dessous. Comparez le résultat retourné par la commande `whoami` avec le nom du compte sous lequel vous vous êtes authentifié.

```
$> whoami
```

3) Informations système

Il est possible d'obtenir des informations générales sur le système sur lequel vous vous trouvez. Pour cela il est possible d'utiliser la commande **uname**, plusieurs options peuvent être utilisées afin d'obtenir des informations différentes.

- Pour obtenir le nom du noyau (cœur du système d'exploitation), tapez la commande `uname` sans option.

Question 4 – Indiquez quel noyau est utilisé par le système Linux que vous utilisez.

- Le système Linux est en perpétuel évolution, certaines applications peuvent nécessiter une version précise du noyau système afin de fonctionner correctement. Pour obtenir la version du noyau système il faut utiliser l'option **-v**. Cette option permet également d'obtenir la distribution Linux utilisée (une distribution comprend un noyau Linux ainsi qu'un regroupement de programmes permettant de répondre aux besoins des utilisateurs).

Question 5 – Indiquez quelle version de noyau et distribution sont utilisées par le système Linux que vous utilisez.

- Tout système d'exploitation est conçu pour une architecture matérielle particulière. Certains systèmes d'exploitation peuvent prendre en charge diverses architectures, c'est le cas du système d'exploitation Linux. Afin d'obtenir l'architecture considérée par le système Linux que vous utilisez il faut utiliser l'option **-m**.

Question 6 – Indiquez quelle architecture est considérée par le système Linux que vous utilisez.

- Chaque machine est désignée par un nom qui a été défini en général lors de l'installation du système d'exploitation. Ce nom est notamment utilisé pour désigner la machine lorsqu'elle est connectée un réseau local (LAN).

Question 7 – Indiquez quel nom a été attribué à la machine que vous utilisez.

Correction : Il faut utiliser la commande `$> uname -n`

- Il est également possible d'avoir l'ensemble de ces informations à l'aide de l'option **-a** de la commande `uname`.

II. Interactions avec le Système de Gestion de Fichiers

Après avoir vu quelques commandes permettant d'obtenir des informations générales sur le système d'exploitation, nous allons maintenant nous intéresser à un certain nombre de commandes utiles pour interagir avec le système de gestion de fichiers.

1) Répertoire de travail

Lorsque vous ouvrez un shell, le système se place à la racine du répertoire associé au compte utilisateur avec lequel vous vous êtes authentifié (en général `/home/jdoe` pour l'utilisateur `jdoe`). Vous pouvez connaître à tout moment votre répertoire de travail actuel (ou répertoire courant) en tapant la commande `pwd`.

Question 8 – Tapez la commande `pwd` dans un terminal et indiquez quel est le résultat retourné par la commande. Que pouvez-vous en déduire ?

2) Création de répertoires

Vous pouvez créer un nouveau répertoire à l'aide de la commande `mkdir` (pour *make directory* en anglais) suivie du nom du répertoire à créer. Si vous indiquez un chemin (valide dans l'arborescence) devant le nom de répertoire alors le répertoire sera créé à l'endroit indiqué, sinon le répertoire sera créé par défaut dans le répertoire courant.

Question 9 – Créez un répertoire de travail nommé `rep_travail` dans le répertoire courant, pour cela entrez la commande suivante :

```
$> mkdir rep_travail
```

Question 10 – Donnez la commande pour créer un répertoire de travail nommé `rep_travail2` dans le répertoire `rep_travail` que vous venez de créer.

Correction : Il faut utiliser commande suivante `$> mkdir rep_travail/rep_travail2`

Question 11 – Donnez l'option permettant le même résultat que les deux questions précédentes avec un seul appel à `mkdir` (c'est-à-dire la sous-arborescence `rep_travail/rep_travail2`).

Correction : Il faut utiliser commande suivante `$> mkdir -p rep_travail/rep_travail2`

3) Lister le contenu d'un répertoire

Vous pouvez afficher le contenu d'un répertoire à l'aide de la commande `ls` (pour *list* en anglais). Cette commande prend plusieurs paramètres, parmi ceux-ci il y a le chemin du répertoire dont il faut lister le contenu. Si aucun répertoire n'est indiqué, le répertoire courant est utilisé.

Question 12 – Entrez la commande suivante pour lister le contenu du répertoire que vous venez de créer et indiquez ce qui est fourni par cette commande :

```
$> ls rep_travail
```

Comme dans la majorité des systèmes d'exploitation, les fichiers et répertoires disposent de plusieurs attributs comme par exemple le nom du fichier, le type de fichier, les droits d'accès, le nom du propriétaire, la taille, la date de dernière modification.

Afin d'obtenir toutes ces informations pour chaque élément du répertoire listé il faut utiliser l'option **-l** (pour affichage *long*).

L'exemple ci-dessous montre des informations fournies par l'exécution de la commande `ls -alh` d'un répertoire :

```
$> ls -alh
total 32K
drwxr-xr-x  2  steph  steph  4,0K  nov. 24 19:18  .
drwxr-xr-x  6  steph  steph  4,0K  oct.  3 15:45  ..
-rwxr-xr-x  1  steph  steph  7,2K  juin 10 11:28  ecr
-rw-r--r--  1  steph  steph  410   juin 10 11:28  ecr.c
-rwxr-xr-x  1  steph  steph  7,2K  juin 10 11:27  lec
-rw-r--r--  1  steph  steph  359   juin 10 11:27  lec.c
```

La première ligne indique la taille totale des fichiers et répertoires contenus dans le répertoire courant listé, ici 32 kilo-octets (multiple de 4 ko car les données sont stockées dans des blocs disque de taille 4 Ko).

Les informations fournies ensuite sont regroupées par colonnes :

- **Première colonne (premier caractère) :** indique le type fichier (symbole `-`) ou répertoire (lettre `d` pour *directory* en anglais). Il existe d'autres types de fichiers utilisés par le système.
- **Première colonne (autres caractères) :** indique les droits d'accès au fichier ou répertoire. Il y a trois ensembles de trois lettres (`r` pour le droit en lecture, `w` pour le droit en écriture, `x` pour le droit en exécution). Le symbole `-` indique que le droit correspondant à la position n'est pas autorisé. Le premier triplet correspond aux droits du propriétaire (ayant créé le fichier en général), le deuxième triplet correspond au groupe auquel appartient le propriétaire et enfin le dernier triplet correspond aux autres utilisateurs du système.
- **Deuxième colonne :** indique le nombre de références vers cette entrée.
- **Troisième colonne :** nom du propriétaire du fichier.
- **Quatrième colonne :** nom du groupe auquel appartient le propriétaire.
- **Cinquième colonne :** taille du fichier (un répertoire possède la taille d'un bloc suffisant en général pour stocker la liste et les attributs des fichiers et répertoire qu'il contient).
- **Sixième colonne :** date de création ou de dernière modification du fichier.
- **Septième colonne :** nom du fichier ou du répertoire.

La deuxième et la troisième ligne sont particulières car elles fournissent des informations respectivement sur le répertoire courant et le répertoire parent. Tandis que les lignes suivantes indiquent les informations associées à chaque fichier ou sous-répertoire contenu dans le répertoire courant.

Question 13 – Indiquez les différents types de fichiers contenus dans le répertoire `rep_travail` listés par la commande `ls -l`.

Question 14 – Indiquez les différents types de fichiers contenus dans le répertoire `rep_travail` listés par la commande `ls -al`. A quoi correspondent les lignes supplémentaires affichées ?

Correction : tous les répertoires dont le nom commence par « . » sont également listés. Certaines applications créent un répertoire dans le répertoire de chaque utilisateur commençant par « . » afin de stocker des fichiers de configuration. De plus, l'entrée « . » et « .. » font référence respectivement au répertoire listé et au répertoire parent dans l'arborescence.

Question 15 – Même question en utilisant la commande `ls -alh`. Quelle différence remarquez-vous ?

Correction : la différence avec le résultat de la question précédente se situe au niveau de l'affichage des tailles de chaque élément du répertoire listé. Les tailles ne sont pas indiquées en octet mais en utilisant une unité (kilo-octets, mega-octets ...) plus lisible pour un humain (option `h` pour human).

4) Changement de répertoire courant

Il est possible de changer de répertoire courant en utilisant la commande `cd` (pour *change directory* en anglais) suivie du chemin du répertoire à atteindre dans l'arborescence. Pour remonter d'un niveau dans l'arborescence de fichiers il faut utiliser le symbole « .. » faisant référence au répertoire parent dans l'arborescence du système de fichiers.

Question 16 – Donnez la commande pour changer de répertoire courant et allez dans le répertoire `rep_travail/rep_travail2` que vous avez créé précédemment. Que retourne la commande `pwd` ?

Correction : la commande à taper est `$> cd rep_travail/rep_travail2`

Question 17 – A partir du répertoire `rep_travail2`, donnez la commande permettant de se déplacer vers le répertoire parent `rep_travail`.

Correction : la commande à taper est `$> cd ..`

Lorsque la commande `cd` est utilisée sans paramètre, alors le système change de répertoire courant en se déplaçant dans le répertoire racine associé au compte utilisateur avec lequel vous vous êtes authentifié (en général `/home/jdoe` pour l'utilisateur `jdoe`). D'autre part, ce répertoire racine est symbolisé par « ~ » et peut être utilisé avec toute commande prenant comme paramètre un chemin d'accès. Les symboles « . », « .. » et « ~ » permettent de construire des chemins d'accès relatifs dans l'arborescence du système de fichiers.

5) Création de fichiers

La commande `touch` permet de créer un fichier dont le nom est indiqué en paramètre.

Question 18 – Donnez la commande pour créer un fichier de nom `newfich` dans le répertoire `rep_travail`.

Correction : la commande à taper est `$> touch ~/rep_travail/newfich`

Question 19 – Indiquez la taille du fichier *newfich* que vous venez de créer.

Correction : le fichier créé possède une taille nulle car celui-ci ne contient aucune information.

6) Copie de fichiers ou répertoires

La commande `cp` permet de copier un fichier ou répertoire SOURCE vers un répertoire DESTINATION.

Question 20 – Donnez la commande permettant de copier le fichier *newfich* contenu dans le répertoire `rep_travail` vers le répertoire racine associé au compte utilisateur avec lequel vous vous êtes authentifié.

Correction : la commande à taper est `$> cp ~/rep_travail/newfich ~`

Il est également possible de demander à la commande `cp` d'effectuer des vérifications avant de copier un fichier. En effet, il est possible que DESTINATION contienne déjà un fichier de même nom que SOURCE. Le comportement par défaut est d'écraser le fichier contenu dans DESTINATION. L'option `-u` permet de réaliser la copie dans DESTINATION soit si le fichier dans DESTINATION est moins récent ou si il n'y a aucun fichier de même nom que SOURCE dans DESTINATION.

Question 21 – Créez un fichier *newfich* dans le répertoire racine de votre compte, puis donnez la commande permettant de copier le fichier `~/rep_travail/newfich` vers votre répertoire racine en testant les dates de création. Vérifiez si le fichier contenu à la racine de votre compte n'a pas été écrasé, pour vous aider vous pouvez comparer les dates de création des deux fichiers.

Correction : la commande à taper est `$> cp -u ~/rep_travail/newfich ~`

Il est à noter que les options `-n` permet de ne pas écraser un fichier existant dans DESTINATION présent dans SOURCE.

Un répertoire ne peut être copié directement comme un fichier, car contrairement à ce dernier un répertoire peut contenir des éléments. Pour pouvoir copier un répertoire et son contenu, il faut utiliser l'option `-r` (pour *recursive* en anglais).

Question 22 – Créez un fichier *newfich2* dans le répertoire `rep_travail/rep_travail2`, puis donnez la commande permettant de copier le répertoire `rep_travail2` et son contenu vers le répertoire racine associé au compte utilisateur avec lequel vous vous êtes authentifié. Est-ce que le répertoire `~/rep_travail2` contient bien le fichier *newfich2* ?

Correction : la commande à taper est `$> cp ~/rep_travail/rep_travail2 ~`

7) Déplacement ou renommage de fichiers et répertoires

La commande `mv` (pour *move* en anglais) permet de déplacer un fichier ou répertoire SOURCE vers un répertoire DESTINATION. Dans le cas particulier où SOURCE et DESTINATION désigne le même répertoire alors le fichier ou répertoire SOURCE sera renommé comme indiqué par DESTINATION.

Question 23 – Donnez la commande permettant de déplacer le fichier `~/rep_travail/newfich` vers le répertoire `~/rep_travail/rep_travail2`.

Correction : les commandes à taper sont `$> mv -u ~/rep_travail/newfich ~/rep_travail/rep_travail2`

Question 24 – Donnez les commandes permettant de renommer respectivement le fichier `~/newfich` en `~/fich` et le répertoire `~/rep_travail/rep_travail2` en `~/rep_travail/sub_rep`.

Correction : les commandes à taper sont `$> mv -u ~/newfich ~/fich`

`$> mv -u ~/rep_travail/rep_travail2 ~/rep_travail/sub_rep`

Il est à noter que les options `-u` et `-n` pour la commande `mv` ont le même comportement qu'avec la commande `cp`.

8) Suppression de répertoires

La commande `rmdir` permet de supprimer le répertoire passé en paramètre. Dans le cas où le répertoire à supprimer n'est pas vide, le répertoire ne sera pas supprimé et la commande retourne une erreur.

Question 25 – Créez le répertoire `~/rep_travail/rep_test`, puis donnez la commande permettant de supprimer le répertoire que vous venez de créer.

Correction : Il faut utiliser les commandes suivantes

`$> mkdir ~/rep_travail/rep_test`
`$> rmdir ~/rep_travail/rep_test`

Question 26 – Créez l'arborescence de répertoires `~/rep_test/sub_test1/sub_test2`, puis donnez la commande permettant de supprimer toute la sous-arborescence `~/rep_test/sub_test1/sub_test2` en une seule commande.

Correction : Il faut utiliser l'option `-p` comme dans les commandes suivantes

`$> mkdir -p ~/rep_test/sub_test1/sub_test2`
`$> rmdir -p ~/rep_test/sub_test1/sub_test2`

9) Suppression de fichiers

La commande `rm` permet de supprimer un fichier dont le chemin d'accès est passé en paramètre.

Question 27 – Donnez la commande pour supprimer le fichier `~/newfich`.

Correction : Il faut utiliser la commande suivante `$> rm ~/newfich`

Question 28 – Donnez l’option de la commande `rm` permettant de supprimer un répertoire. Qu’observez-vous par rapport au comportement de `rmdir` ?

Correction : Il faut utiliser la commande suivante

```
$> rm -r ~/rep_travail/rep_travail2
```

Il est à noter que contrairement à la commande `rmdir` le répertoire indiqué en paramètre et son contenu sont supprimé sans message d’erreur. En effet, avec cette option la commande `rm` supprime récursivement le contenu du répertoire avant de supprimer celui-ci.

10) Lien entre fichiers

La commande `ln` permet de réaliser des liens symboliques entre éléments dans le système de fichiers, à l’instar de raccourci sous le système Windows. La commande prend en paramètre le chemin vers le fichier ou répertoire ciblé puis le nom du lien à créer.

Question 29 – Placez vous à la racine de votre compte et donnez la commande pour créer un lien symbolique nommé `lsym` faisant référence au fichier `~/rep_travail/newfich`.

Correction : Il faut utiliser la commande suivante

```
$> ln -s ~/rep_travail/newfich lsym
```

Question 30 – Supprimez le lien symbolique que vous venez de créer et dites si le fichier `~/rep_travail/newfich` pointé par celui-ci est toujours présent dans système de fichiers.

Correction : Il suffit de supprimer le lien `lsym` comme on le ferait avec simple fichier, la suppression du lien symbolique n’impact en rien le fichier pointé par celui-ci.

Question 31 – Qu’arrive –t-il au lien symbolique si le fichier pointé par celui-ci est supprimé ?

Correction : Dans ce cas le lien pointe sur vide et le lien est inutilisable.

Afin d’éviter ce comportement, il est possible de ne pas utiliser l’option `-s` de la commande `ln`. Cela permet de créer un lien vers les données physiquement enregistrées sur le disque. En revanche les données ne sont pas supprimées du disque tant qu’il existe un lien vers ces données.

Question 32 – Reprenez les trois dernières questions en utilisant la commande `ln` sans l’option `-s`. Que constatez-vous ?

11) Recherche de fichiers

La commande `find` permet de réaliser des recherches dans l’arborescence du système de fichiers. Le premier paramètre est le répertoire dans lequel la recherche doit être réalisée, puis ensuite l’option `-name` suivi d’un nom de fichier ou de répertoire entre guillemets doit être fourni. Le chemin d’accès des fichiers correspondant à la recherche seront listés en résultat.

Question 33 – Donnez la commande pour effectuer une recherche des fichiers ayant comme nom *newfich* à partir de la racine de votre compte.

Correction : il faut utiliser la commande suivante :

```
$> find ~ -name "newfich"
```

Il est possible de réaliser la même recherche mais en considérant uniquement les répertoires, pour cela il suffit d'utiliser l'option **-type d**.

Il existe bien d'autres options pour indiquer des critères de recherche sur les fichiers et répertoires.

III. Visualisation de fichiers

Nous allons nous intéresser à plusieurs commandes permettant d'afficher dans un shell le contenu d'un fichier texte. Cela peut être particulièrement utile si vous sur une machine sans gestionnaire de fenêtres ou tout simplement connecté à une machine distante avec un shell.

1) Affichage complet de fichiers

La commande `cat` permet d'afficher tout le contenu du fichier passé en paramètre sur un terminal.

Question 34 – Affichez tout d'abord à l'aide de la commande `cat` l'historique des commandes que vous avez tapées, cela est stocké pour un shell de type `bash` (le plus couramment utilisé) dans le fichier `~/.bash_history`.

Donnez l'option de cette commande permettant d'afficher le contenu d'un fichier en numérotant les lignes.

Correction : il faut utiliser les commandes suivantes :

```
$> cat ~/.bash_history  
$> cat -n ~/.bash_history
```

L'option -n (pour number) permet de numérotter les lignes affichées.

2) Affichages interactif

Deux commandes `more` et `less` permettent d'afficher le fichier à visualiser par morceau. La première commande permet d'afficher le fichier passé en paramètre écran par écran, pour afficher la série de *k*-lignes suivantes (avec *k* correspondant à la taille l'écran) il faut utiliser la touche « *Espace* ».

La seconde commande émule la première mais constitue un programme plus léger que `more`, ce qui permet de gagner en réactivité pour la visualisation de fichiers de plusieurs Go. Cela est dû au fait que la commande `less` n'a pas besoin de lire tout le contenu du fichier avant de l'afficher. De plus, il est possible d'utiliser les touches standard pour naviguer dans l'affichage du fichier (touches « *Page Up* », « *Page Down* », « *Flèche Haut* », « *Flèche Bas* »).

Pour arrêter la visualisation du fichier, il faut presser la touche « **q** » du clavier pour ces deux commandes.

Question 35 – Utilisez tout d'abord la commande `more` pour afficher le contenu de l'historique `bash`.

Réalisez la même manipulation à l'aide de la commande `less`.

Correction : il faut utiliser les commandes suivantes :

```
$> more ~/.bash_history  
$> less ~/.bash_history
```

3) Affichage morcelé

Il est possible de n'afficher qu'une partie d'un fichier, à savoir les k premières ou dernières lignes d'un fichier (avec k une quantité de lignes).

Les commandes `head` et `tail` permettent d'afficher par défaut respectivement les 10 premières lignes et 10 dernières lignes d'un fichier.

Question 36 – Utilisez tout d'abord la commande `head` pour afficher le contenu de l'historique `bash`.

Réalisez la même manipulation en affichant les 15 premières lignes au lieu des 10 premières.

Correction : il faut utiliser les commandes suivantes :

```
$> head ~/.bash_history  
$> head -n 15 ~/.bash_history
```

Question 37 – UDonnez la commande permettant d'afficher tout le fichier excepté les 15 dernières lignes de l'historique `bash`.

Correction : il faut utiliser la commande suivante :

```
$> head -n -15 ~/.bash_history
```

Question 38 – Utilisez tout d'abord la commande `tail` pour afficher le contenu de l'historique `bash`.

Réalisez la même manipulation en affichant les 15 dernières lignes au lieu des 10 dernières.

Correction : il faut utiliser les commandes suivantes :

```
$> tail ~/.bash_history  
$> tail -n 15 ~/.bash_history
```

IV. Droits d'accès

Un aspect important dans tout système d'exploitation est la gestion des droits d'accès des utilisateurs du système.

Le système Linux considère 3 classes d'utilisateurs pour chaque ressource dont il faut gérer les droits d'accès : le propriétaire (**u** pour *user*), le groupe auquel appartient le propriétaire (**g** pour *group*) et les autres utilisateurs (**o** pour *other*). Pour chaque classe, 3 modes d'accès à la ressource sont considérés : accès en lecture (**r**), en écriture (**w**) ou en exécution (**x**).

Nous allons nous intéresser plus particulièrement à la gestion des droits d'accès dans le système de fichiers.

Voici ci-dessous un exemple d'informations retournées par la commande `$> ls -alh`

```
total 32K
drwxr-xr-x  2  steph  steph  4,0K  nov. 24 19:18  .
drwxr-xr-x  6  steph  steph  4,0K  oct.  3 15:45  ..
-rwxr-xr-x  1  steph  steph  7,2K  juin 10 11:28  ecr
-rw-r--r--  1  steph  steph  410   juin 10 11:28  ecr.c
-rwxr-xr-x  1  steph  steph  7,2K  juin 10 11:27  lec
-rw-r--r--  1  steph  steph  359   juin 10 11:27  lec.c
```

Question 39 – Indiquez dans l'exemple ci-dessus quels sont les droits d'accès autorisés au fichier `ecr.c` pour chaque classe d'utilisateurs.

Correction : Les droits d'accès sont : lecture et écriture pour le propriétaire, lecture pour le groupe du propriétaire et tous les autres utilisateurs.

1) Identifiants utilisateur et groupe

La commande `id` permet d'obtenir les numéros d'identification associé à l'utilisateur courant, sont groupe ainsi que les autres groupes configurés dans le système.

Dans l'exemple ci-dessous indique l'identifiant de l'utilisateur `steph` (ici 1000), l'identifiant de son groupe `users` (ici 100) et l'identifiant des autres groupes existants

```
$> id
uid=1000(steph) gid=100(users) groups=100(users),24(cdrom),44(video)
```

Question 40 – Exécutez la commande `id` et indiquez l'identifiant associé par le système au compte utilisateur que vous utilisez.

Comparez le numéro obtenu avec `id` et celui indiqué dans le fichier `/etc/group`.

2) Changement de propriétaire

La commande `chown` (pour *change owner*) a comme effet de changer l'utilisateur propriétaire ou le groupe d'utilisateur du fichier ou répertoire indiqué en paramètre. Il est nécessaire de fournir le nom ou l'identifiant soit du propriétaire, soit du groupe, ou les deux comme suit `[PROPRIETAIRE][:[GROUPE]]` (les accolades « [] » indiquent un choix).

Question 41 – Indiquez à l'aide de la commande `ls` l'utilisateur et le groupe auquel appartient le fichier `~/newfich`.

Donnez la commande permettant d'associer le groupe `users` au fichier `~/newfich`.

Correction : la commande correspondant est `$> chown users ~/newfich`

Question 42 – IDonnez l'option permettant d'appliquer les changements de propriétaire ou groupe à une sous-arborescence.

Correction : la commande correspondant est `$> chown -R users ~/newfich`

3) Changement de droits d'accès

La commande `chmod` permet de changer les droits d'accès d'un fichier ou répertoire. Il est possible de modifier les droits associés soit au propriétaire (option `u`), le groupe du propriétaire (option `g`), les autres utilisateurs (option `o`) ou les trois types d'utilisateurs à la fois (option `a`). Pour cela, étant donné le groupe d'utilisateurs cible il faut indiquer

l'ajout (symbole `+`) ou la suppression (symbole `-`) de chaque type d'accès lecture (**r**), écriture (**w**) ou exécution (**x**).

Question 43 – Indiquez ce que réalise la commande suivante :

```
$> chmod g+w ~/newfich
```

Correction : la commande permet d'ajouter le droit en écriture aux utilisateurs appartenant au groupe du propriétaire du fichier ~/newfich

Question 44 – Donnez la commande permettant d'ajouter le droit en exécution au fichier ~/newfich à tous les utilisateurs.

Correction : la commande correspondant est `$> chmod a+x ~/newfich`

Question 45 – Donnez la commande permettant de retirer le droit en lecture au fichier ~/newfich aux utilisateurs n'étant ni le propriétaire du fichier et n'appartenant pas au groupe du propriétaire.

Correction : la commande correspondant est `$> chmod o-r ~/newfich`

Il est également possible d'utiliser une notation numérique composé de trois chiffres en octal (de 0 à 7) pour indiquer les nouveaux droits d'accès à affecter au fichier. Le premier chiffre correspond aux droits du propriétaire, le second aux droits du groupe du propriétaire et le dernier aux autres utilisateurs.

Chaque chiffre en octal est obtenu en additionnant les valeurs associées aux trois accès : 4 pour le droit en lecture, 2 pour le droit en écriture et 1 pour le droit en exécution. Ainsi le chiffre $6=4+2$ correspond aux droits en lecture et écriture.

Question 46 – Donnez la commande permettant d'affecter les droits suivant au fichier ~/newfich :

- lecture, écriture et exécution pour le propriétaire,
- lecture et écriture au groupe du propriétaire,
- aucun droit aux autres utilisateurs.

Correction : la commande correspondant est `$> chmod 760 ~/newfich`

Il peut être utile de changer d'utilisateur afin de pouvoir obtenir les accès nécessaires pour réaliser une tâche. La commande `su nom_utilisateur` permet s'authentifier comme `nom_utilisateur` pour exécuter des commandes avec les droits associés à `nom_utilisateur` indiqué en paramètre. Dans le cas où aucun nom d'utilisateur n'est indiqué, la commande considère que l'on demande à s'authentifier sous l'utilisateur `root` (super-utilisateur ayant tous les droits dans le système).

Il est à noter que l'on peut terminer la session ouverte sous un utilisateur en tapant la commande `exit` dans le shell, pour revenir à la session ouverte sous le précédent utilisateur. Si aucune autre session n'a été ouverte sous un autre utilisateur cette commande aura pour effet de fermer le shell.

Il est également possible de demander au système d'exploitation d'exécuter une commande nécessitant des droits d'accès non accordés à compte utilisateur que vous utilisez. Il faut utiliser la commande `sudo cmd`, avec `cmd` la commande et ses paramètres à exécuter. Afin que cela fonctionne, il faut que la commande `sudo` soit correctement configurée pour vous autoriser à l'utiliser, c'est-à-dire que l'utilisateur sous lequel vous vous êtes authentifié ait les droits d'accès à cette commande.

4) Espaces disque

Il est possible d'obtenir des informations générales sur les différents points de montage qui sont des parties de l'arborescence du système de fichiers représentant une partition disque ou des répertoires particuliers comme des périphériques.

La commande `df` permet d'obtenir la taille totale, la taille d'espace disque utilisé et disponible pour chaque point de montage. Il est possible d'afficher ces informations sous un format plus lisible pour un humain à l'aide l'option **-h**.

Un exemple de rendu obtenu avec cette commande est donné ci-dessous :

Sys. fich.	Taille	Util.	Dispo	Uti%	Monté sur
Rootfs	52G	10G	39G	21%	/
Udev	10M	0	10M	0%	/dev
Tmpfs	101M	660K	100M	1%	/run
Tmpfs	5,0M	0	5,0M	0%	/run/lock
Tmpfs	582M	80K	582M	1%	/run/shm
/dev/sda6	9,9G	5,5G	4,0G	59%	/home

Question 47 – Utilisez la commande `df` pour afficher les informations sur les points de montage de la machine que vous utilisez.

5) Taille fichiers et répertoires

La commande `du` permet d'obtenir la taille d'un fichier ou d'un répertoire passé en paramètre. Plusieurs options sont disponibles :

- **-a** : permet de lister tous les fichiers et répertoire à partir du chemin passé en paramètre et d'indiquer la taille de chacun,
- **-h** : permet d'afficher des tailles en utilisant des unités plus lisibles pour les humains,
- **-c** : permet d'afficher en plus la taille totale du fichier ou répertoire passé en paramètre.

Question 48 – Utilisez la commande `du` pour afficher la taille des fichiers et répertoire contenu dans répertoire `~/rep_travail` créé précédemment.

Correction : la commande correspondant est `$> du -ah ~/rep_travail`

Question 49 – Donnez les options permettant de n'afficher que la taille totale du fichier ou répertoire passé en paramètre à la commande `du`.

Correction : la commande correspondant est `$> du -sh ~/rep_travail`

V. Archivage

Afin de pouvoir manipuler plus facilement une sous-arborescence du système de fichier, il est possible de réaliser une archive. Pour cela un certain nombre d'utilitaires sont disponibles, nous en verrons deux qui sont les plus utilisés.

1) Zip et Unzip

La commande `zip` permet de réaliser une archive compressée, alors que la commande `unzip` permet de décompresser une archive au format zip.

Voici ci-dessous un exemple d'utilisation de ces commandes :

```
$> zip -r foo.zip foo
```

```
$> unzip foo.zip
```

La première commande permet d'archiver un répertoire (option `-r`) de nom `foo` et créé une archive dénommée `foo.zip`, tandis que la seconde décompresse l'archive `foo.zip` dans le répertoire courant.

Question 50 – Réalisez une archive contenant une sous-arborescence du système de fichier et nommez la `test.zip`, puis copiez cette archive dans un autre répertoire et décompressez la.

Question 51 – Il existe plusieurs niveaux de compression allant de `-0` (aucune compression) à `-9` (taux de compression élevé mais exécution lente). Réalisez la même opération que précédemment mais en utilisant un niveau de compression de `-9`.

Correction : la commande correspondant est

```
$> zip -9 test.zip ~/rep_travail
```

2) Tar

La commande `tar` permet aussi de réaliser une archive, mais également de décompresser une archive en fonction des options passées en paramètre.

Plusieurs options sont disponibles, voici ci-dessous quelques principales options :

- `-c` : créer une archive,
- `-x` : extraire une archive,
- `-f` : nom de l'archive à créer ou à décompresser,
- `-z` : compression/décompression de l'archive,
- `-v` : affiche le détail des opérations réalisées.

Question 52 – Réalisez une archive contenant une sous-arborescence du système de fichier et nommez la `test.tgz`, puis copiez cette archive dans un autre répertoire et décompressez la.

Correction : les commandes correspondantes sont

```
$> tar -cvzf test.tgz ~/rep_travail
```

```
$> tar -xvzf test.tgz
```

VI. Périphériques

Nous allons considérer quelques commandes permettant d'obtenir des informations sur les divers périphériques matériels reconnus par le système d'exploitation Linux que vous utilisez.

1) Processeur

Pour connaître les spécifications du processeur disponible sur votre machine vous pouvez utiliser la commande `lscpu`.

Voici ci-dessous un exemple d'informations retournées par cette commande. On peut voir ici que l'architecture processeur est de type x86 64 bits, il y a un processeur possédant 1 seul cœur cadencé à 1,7 GHz. La mémoire cache de niveau 1 a une taille de 32 Ko, alors que le cache de niveau 2 a une taille de 6 Mo.

```
Architecture :      x86_64
Mode(s) opératoire(s) des processeurs : 32-bit, 64-bit
Boutisme :          Little Endian
Processeur(s) :     1
Liste de processeur(s) en ligne : 0
Thread(s) par cœur : 1
Cœur(s) par socket : 1
Socket(s) :         1
Nœud(s) NUMA :      1
Identifiant constructeur : GenuineIntel
Famille de processeur : 6
Modèle :            58
Révision :          9
Vitesse du processeur en MHz : 1796.380
BogoMIPS :          3592.76
Cache L1d :         32K
Cache L1d :         32K
Cache L2d :         6144K
Nœud NUMA 0 de processeur(s) : 0
```

Question 53 – Tapez la commande `lscpu` sur votre machine et indiquez quelles sont les caractéristiques du processeur de la machine que vous utilisez.

2) Périphériques USB

Il est également possible d'obtenir une liste de périphériques connectés sur le bus USB avec leur nom et identifiant. Pour cela, il faut utiliser la commande `lsusb`.

Voici ci-dessous un exemple d'informations retournées par cette commande. On peut voir ici qu'il y a deux périphériques USB sur le bus 1, le premier d'identifiant 1 et le second d'identifiant 13.

```
Bus 001 Device 001: ID 1d6b:0001 Linux Foundation 1.1 root hub
Bus 001 Device 013: ID 80ee:0021 USB Tablet
```

Question 54 – Tapez la commande `lsusb` sur votre machine et indiquez quelles sont les périphériques connectés sur le bus USB reconnus par la machine que vous utilisez.

3) Autres périphériques

Comme pour les périphériques USB, il est possible d'obtenir la liste des autres périphériques connectés via le bus PCI (ou SCSI) en utilisant la commande `lspci` (ou `lsscsi`).

Voici ci-dessous un exemple d'informations retournées par cette commande. On peut voir ici quelques exemples de périphériques, une carte ethernet, le contrôleur USB, le contrôleur de disque SATA, le contrôleur de cartes ISA.

```
00:00.0 Host bridge: Intel Corporation 440FX - 82441FX PMC [Natoma] (rev 02)
00:01.0 ISA bridge: Intel Corporation 82371SB PIIX3 ISA [Natoma/Triton II]
00:01.1 IDE interface: Intel Corporation 82371AB/EB/MB PIIX4 IDE (rev 01)
00:02.0 VGA compatible controller: InnoTek Systemberatung GmbH Graphics
Adapter
00:03.0 Ethernet controller: Intel Corporation 82540EM Gigabit Ethernet
Controller (rev 02)
00:05.0 Multimedia audio controller: Intel Corporation 82801AA AC'97 Audio
Controller (rev 01)
00:06.0 USB controller: Apple Inc. KeyLargo/Intrepid USB
00:07.0 Bridge: Intel Corporation 82371AB/EB/MB PIIX4 ACPI (rev 08)
00:0d.0 SATA controller: Intel Corporation 82801HM/HEM (ICH8M/ICH8M-E)
SATA Controller [AHCI mode] (rev 02)
```

Question 55 – Tapez la commande `lspci` (ou `lsscsi`) sur votre machine et indiquez quelles sont les périphériques connectés et reconnus par la machine que vous utilisez.