Filière (DUT) : Génie Informatique /S2

Département : Génie Informatique

Manuel de TP :
Système d'exploitation(Linux)

Préparé par :

Idriss CHANA

Enseignant chercheur au Département GI

EST-UMI Meknès

i.chana@umi.ac.ma

Au Profit des Etudiants de La Première Année de la filière GI inscrits en semestre S2

Années Universitaires : 2018/2019

Séance TP1 :

Initiation à l'environnement GNU-Linux/Ubuntu

I- Introduction

Un système GNU-**Linux** est un système d'exploitation répondant à un standard garantissant un ensemble de fonctionnalités utilisables de manière normalisée. Par exemple l'ensemble des commandes étudiées dans ce TP sur **Linux Ubuntu** restent valables sur le système Unix BSD (à la base d'OS X) ou sur le système Unix Solaris etc...

Le standard **Unix** étant une référence incontournable de l'informatique professionnelle, nous souhaitons vous voir pratiquer ce système en tant que future technicien supérieur

Cette séance est un 1er contact avec les commandes et schémas usuels d'une utilisation de système linux en ligne de commande : nous envoyons des ordres et recevons des réponses en mode textuel dans une console ou terminal selon le vocabulaire Linux.

Que vous utilisiez les machines avec connexion à distance sur Linux, ou que vous utilisiez un Linux installé sur une machine personnelle, vous serez sur un Bureau avec icônes, lanceur d'applications, navigateur de fichier etc... Tout le confort moderne. C'est pratique mais ce n'est PAS ce qui nous intéresse car cette interface de haut niveau est très changeante (d'un système à l'autre, selon les versions, les options...) et ne constitue qu'un emballage superficiel du système : les commandes en terminal sont standardisées et présentent un intérêt pour des compétences à long terme.

II- Lancer votre session Linux-Ubuntu et ouvrir un terminal

Sur votre machine:

Accédez au menu **Applications**, dérouler **→ Accessoires → Terminal**

Comme nous allons manœuvrer dans le système de fichiers (arborescence de répertoires) à partir du terminal, il peut être utile de vérifier ce qui se passe depuis une interface plus intuitive : menu Raccourcis en haut du bureau → Poste de travail.

III- Le terminal et l'interpréteur de commande : shell

Le terminal ou la console est une interface textuelle qui permet à l'utilisateur d'entrer les commandes à faire exécuter par le système.

Une ligne de commande commence par le **nom** de la commande et est éventuellement suivie par des **options** ou **paramètres**. La commande est validée (l'action est déclenchée) par retour ligne (touche Entrée). Les lignes de commande ne sont pas directement transmises du terminal au système mais passent par l'interpréteur de commande ou **shell** qui est le véritable interlocuteur, le terminal n'étant

qu'une boite de saisie et d'affichage de texte. Le **shell** vérifie la cohérence syntaxique de la ligne, gère le contexte des commandes (répertoire courant où a lieu l'action...) et lance effectivement l'exécutable associé à la commande. Dans le terminal vous devez voir une invite de commande ou prompt signalant que le shell est prêt à recevoir vos ordres. Les commandes s'écrivent directement après ce prompt. Vous validerez les commandes suivantes en vérifiant que le résultat décrit correspond bien à ce qui est indiqué.

1- QUI SUIS-JE et Où je suis dans l'arborescence ?

A- Taper **whoami :** ça renvoie le nom de votre l'utilisateur (le logging)

B- Taper **pwd** : ça indique le répertoire de travail le "répertoire courant"

Si vous lancer des commandes sur des fichiers ou répertoires sans préciser leurs emplacement, se sont les fichiers (et répertoires) de répertoire courant qui vont subir ces commandes **par default**

Sous GNU/Linux (aussi Unix) un répertoire (ou un fichier) est désigné par un chemin (path) qui est la séquence des répertoires qu'il faut traverser depuis la racine / pour y accéder en descendant l'arborescence.

echo : pour afficher un texte**,** une valeurs d'une variables

C- Taper **echo bonjour mes collègues**

$** : Quand on utilise le symbole **$ avant une variable c'est qu'on veut manipuler (afficher, affecter …) la valeur de cette variable

1.1 Exemple :

D- Taper dans l'invite de commande: **x=25**

E- Puis taper : **echo $x**

F- Taper aussi **: echo $HOME**

Cette dernière, nous renvoie l'arborescence du répertoire dédié à l'utilisateur (chaque utilisateur a son répertoire personnel). Au démarrage d'un nouveau terminal, le **HOME** porte la valeur répertoire courant.

La commande **echo** demande un affichage immédiat des paramètres. Dans la commande suivante, les noms commençant par **$** sont des **variables du système**, qui sont des chaînes, on demande leur valeur.

G- echo C'est moi l'utilisateur $USER mon shell est le $SHELL

Cette dernière commande semble ne pas marcher : les guillemets simples utilisés en apostrophes encadrent **'est moi l'** comme une chaîne à utiliser telle quelle (strong quoting).

- On peut adopter une "quotation" plus faible pour garder la substitution des variables par leur contenu mais échapper les guillemets simples :

H- echo " C'est moi l'utilisateur $USER mon shell est le $SHELL "

- On peut adopter une banalisation à ces guillemets simples en utilisant le back-slash « **\ »** pour éviter leur effet :

I- echo C\'est moi l\'utilisateur $USER mon shell est le $SHELL

Le **shell** que nous utilisons est le **BASH**, un shell très répandu mais pas le seul à exister il y a :

- sh (Bourne shell)
- bash (Bourne again shell)
- csh (C shell)
- Tcsh (Tenex C shell)
- ksh Korn shell
- zsh Zero shell

1.2 Quelques commandes :

J- ls

Lister le contenu du répertoire courant

K- ls -l

(l comme Liste) Lister le contenu du répertoire courant, option longue = infos détaillées

La colonne de gauche indique pour chaque fichier ou dossier :

d pour "directory" → dossier

r pour "**read**" droit en lecture **w** pour "**write**" droit en écriture **x** pour "**eXecute**" droit en exécution

Le **1er** bloc **rwx** concerne le **propriétaire**,

Le **2ème** concerne le **groupe** (ici nous n'entrons pas dans les détails)

Le **3ème** concerne tout le **monde**

1.3 Exemple : **-rw-r--r--** Fichier (pas de **d**) lecture pour tous, écriture propriétaire, pas d'exécution

L- ls –a

Lister le contenu du répertoire courant, option tous (**all**) = afficher les fichiers cachésLes fichiers ou répertoires commençant par**. (point)** sont des fichiers de configuration, cachés par défaut.Les répertoires. **(Point)** et **..** (**Deux points**) sont des répertoires spéciaux qui représentent respectivement, le répertoire courant et le répertoire parent (au dessus du répertoire courant dans l'arborescence)

M- clear : sert au nettoyage de la console**.**

2- Aides (**whatis** pour what is this**, man** pour manual**) !**

N- whatis ls (cp, mv, mkdir cd, rm ….)
O- whatis whatis

La commande **whatis** donne un bref descriptif du rôle joué par la commande donnée en paramètre

P- man ls (cp, mv, mkdir cd ….)

Pour un descriptif plus complet on utilise le manuel, qui explique en détail toutes les options d'une commande. Vous pouvez retrouver **-a** et -l. Inutile de s'attarder (on ne va pas apprendre tout ça!) appuyer sur q pour quitter le manuel et revenir au shell.

Essayer

Q- man man (manuel du manuel) et jeter un œil à la 1ère page.

3- Changer de direction et aller dans d'autres répertoires !

Vous allez naviguer dans le système de fichier, ajouter des répertoires, créer/voir des fichiers...

R- ls ..
S- pwd

Affiche le contenu du répertoire parent du répertoire courant. Le répertoire courant est inchangé.

T- ls .. -l -a
U- pwd

Affiche la totalité du répertoire parent avec infos détaillées (on peut écrire-la au lieu de -l -a) Le répertoire courant est inchangé.

V- cd ..
W- pwd
X- ls

cd = Change Directory : Change le répertoire courant. Ici on atterrit dans le répertoire parent.

Y- cd /
Z- ls -la

Aller à la racine du système de fichier et voir en détail tout ce qui s'y trouve.

AA- cd bin
BB- ls
Aller dans le répertoire des commandes exécutables **bin** (binaries) et voir les commandes disponibles. Il y en a davantage à d'autres emplacements, voir par exemple dans le répertoire **/usr/bin : ls /usr/bin** (la commande **which** permet de savoir où se trouve une commande)
CC- cd ~ : Retour au répertoire HOME : le ~ représente votre répertoire personnel HOME
DD-Taper **pwd** pour s'assurer

On peut exécuter une commande sur un fichier ou répertoire existant dans un autre endroit (répertoire) à partir de notre emplacement : Par exemple si on veut lister le contenu de **/lib** on reste dans le HOME et on fait **ls /lib** directement :

A noter que les chemins qui commencent par **/** sont des chemins **absolus** (on part de la racine) alors que ceux qui ne commencent pas par **/** sont des chemins **relatifs** (on part du courant)

Par exemple **/tmp** se réfère au répertoire **tmp** à la racine du système tandis que **tmp** se réfère à un répertoire **tmp** supposé être dans le répertoire courant.

4- Historique des commandes et Complétion

Par utilisation de **[ctrl+p]** ou de **[echap +k]** ou aussi des **flèches haut** et **bas** on peut voir (et relancer, ou éditer et relancer) une commande précédente.

Essayer de lancer **echo Rebonjour les collègues** en retrouvant la commande initiale et en ne modifiant que le "Re"

History : renvoie la liste des commande que vous avez tapées c'est un historique numéroté de commandes.

history 10 : pour avoir seulement les **10** dernières...

Vous pouvez relancer une commande en tapant **!** Immédiatement suivi du numéro de commande de l'historique. Essayez avec une des commandes déjà entrée...

Quand on tape une commande qui nécessite d'indiquer un chemin un peu long il est possible de demander la **complétion** de nom en appuyant sur la touche [**Tab]ou deux fois sur la touche [echap]**

Nous allons voir (sans déplacements avec **cd**) les fichiers en-têtes standards du C à notre disposition dans le répertoire **/usr/include**

ls /usr/inc[Appuyer sur Tab] → on obtient la complétion **ls /usr/include/**

Donc on écrit le début d'un nom de répertoire (ou de fichier) et en appuyant sur tab on a la suite. Si il y a plusieurs noms correspondant **Tab** ne complète pas, **Tab** une 2ème fois pour avoir la liste...

ls /u[Tab] → ls /usr/ ajouter juste **s ls /usr/s[Tab][Tab]** → plusieurs réponses

compléter **ls /usr/sh[Tab]** → **ls /usr/share/** (liste de différents utilitaires du système)

L'efficacité du terminal/shell pour piloter un système tient à l'utilisation assidue de ces raccourcis.

Par exemple vous pouvez compléter les commandes : **his[Tab] → history**

5/ Le système de fichier (sous votre HOME) est votre terrain de jeu et le shell est votre bulldozer...

Il est temps de créer un espace de travail.

Vérifiez que vous êtes bien dans votre HOME avec **pwd**, refaire **cd ~** si nécessaire.

mkdir tp1

Make Directory : créer répertoire tp1 dans le répertoire courant (dans HOME donc)

Attention pour vos choix de noms de fichiers : utiliser des caractères spéciaux (autres que _ ou -) accents ou même seulement des espaces c'est chercher les ennuis. Il reste possible de "quoter" pour gérer des chemins avec espaces : cd **"Mon TP1 avec espaces"** par exemple mais ça complique.

cd tp1

mkdir test1

mkdir test2

cd test1

Vous pouvez vérifier qu'on a bien dans home un sous-répertoire tp1 qui lui même contient 2 sous-répertoires test1 et test2 en utilisant l'explorateur de fichiers (poste de travail).

6/ Redirection des flux standards stdin (<)et stdout(>)

La commande **cat** prend les caractères du flux d'entrée stdin (par défaut le clavier) et les répercute directement sur le flux de sortie **stdout** (par défaut l'affichage du terminal).

cat

Vous remarquez qu'il n'y a plus de prompt : nous ne sommes plus en communication avec le shell mais ce que nous écrivons est maintenant intercepté par **cat** en cours d'exécution.

Entrer le texte suivant (↲ représente un appui sur la touche Entrée) :

mon message↲

est reproduit↲

[Appuyer sur la combinaison Ctrl-D pour terminer la saisie]

Ctrl-D envoie une "fin de fichier" (End Of File, EOF) ce qui marque la fin de notre entrée, normalement vous retrouvez le prompt du shell. En cas de blocage dans une commande qui ne rend pas la main au shell il est toujours possible en dernier recours de tuer brutalement la commande avec **Ctrl-C.**

L'intérêt de la commande cat apparaît avec l'utilisation des redirections de flux :

cat >fichier.txt

Puis entrer le texte suivant (terminer par Ctrl-D)

mon message↲

est enregistré↲

[Ctrl-D]

Le symbole **>** redirige le flux de sortie vers un fichier (créé pour l'occasion si il n'existait pas)

ls -l

On a bien créé fichier.txt dans le répertoire courant. Vérifions son contenu :

cat <fichier.txt (vous avez pensé à n'utiliser que **<f** puis **Tab** pour compléter le nom ?)

Cette fois ci on redirige le flux d'entrée pour que l'information vienne du fichier au lieu du clavier.

Ces redirections **<** et **>** sont valables quelque soit la commande (l'exécutable) utilisé.

En fait cat peut recevoir directement un nom de fichier en paramètre (à la place de stdin)

cat fichier.txt

C'est la façon usuelle d'afficher le contenu d'un court fichier texte. La commande s'appelle **cat** car elle permet de concaténer plusieurs fichiers, par exemple si on disposait de 3 fichiers on pourrait grouper leur contenu : **cat fic1.txt fic2.txt fic3.txt >tout.txt**

Si le fichier à voir est trop grand on peut demander moins avec la commande **less** …

ls -l /bin >commandes.txt

ls -l

On obtient le listing des commandes principales du répertoire /bin dans le fichier commandes.txt

less commandes.txt

On peut scroller tranquillement dans le fichier (haut/bas page_haut/page_bas) q pour quitter

Lorsqu'il est utilisé sans fichier en paramètre, **less** prend comme contenu à traiter ce qu'il reçoit sur

stdin (c'est le cas de nombreuses commandes Unix). On aurait pu écrire **less <commandes.txt**

En résumé on a redirigé le flux de sortie de **ls** sur un fichier et ensuite on redirige le flux d'entrée de **less** depuis ce fichier.

ls -l /bin | less

On va retravailler avec fichier.txt ...

cat >fichier.txt

Un nouveau message↲

redirigé vers le fichier↲[Ctrl-D]

cat fichier.txt

On a effacé/remplacé l'ancien contenu par le nouveau. Pour ajouter il faut utiliser la redirection **>>**

cat >>fichier.txt

et ceci↲

vient s'ajouter à la suite!↲[Ctrl-D]

cat fichier.txt

On vérifie bien qu'il y a ajout.

7- Commandes de changement de nom et copie de fichiers.

Vérifier bien que le fichier fichier.txt existe dans le répertoire courant si non :

ls fichier.txt

Puis faire **: cp fichier.txt fichier_copie.txt**

Cette commande copie le contenu de dans le fichier fichier_copie.txt

Attention ! : Si le fichier fichier_copie.txt existe déjà il va s'écraser, si non il sera créer.

mv fichier.txt fichier_avec_autre_nom.txt

Le fichier **fichier.txt** change de nom et devient **fichier_avec_autre_nom.txt**

Attention ! : Si le fichier fichier_avec_autre_nom.txt existe déjà il va s'écraser.

TP N° 2-A

- **Familiariser avec la console**
- **Manipulation des fichiers et répertoires**
- **La hiérarchie des répertoires**

Des fichiers se sont accumulés dans votre répertoire personnel. Vous avez décidé de mettre de l'ordre dans votre compte. Vous prévoyez de créer plusieurs sous-répertoires et de copier et déplacer vos fichiers selon votre organisation. Vous possédez également plusieurs fichiers inutiles qui doivent être supprimés.

1. Connectez-vous en tant que l'utilisateur « estm » avec le mot de passe « estm2014 ».

2. Une fois connecté vous devriez être dans votre répertoire personnel. Comment s'en rassurer.

$ **pwd**

/home/estm

3. Vérifiez de 2 manières que vous possédez des fichiers sur votre répertoire personnel par le biais des commandes ci-dessous.

$ ls

$ ls –a

Pourquoi le résultat des 2 commandes est différent.

Réponse : l'option « -a » rajoutée à la commande « ls » permet de lister aussi les fichiers cachés. Ces fichiers commencent par un « . »

4. Regarder la taille des fichiers contenus dans votre répertoire personnel par le biais de la commande ci-dessous.

$ ls –la

La taille des fichiers est représentée en unité « octet » sur le 5ème champ du résultat de la commande précédente.

5. A ce point on va créer des fichiers et une arborescence pour organiser ces fichiers.

On va tout d'abord utiliser la commande « **touch** » pour la création de fichiers vides.

$ touch {rapport,graph}_{jan,fev,mar}

C'est une manière qui permet de créer 6 fichiers d'un seul coup.

Utilisez la commande « ls » pour examiner le résultat de la dernière commande.

$ ls

graph_fev graph_jan graph_mar rapport_fev rapport_jan rapport_mar

Pour organiser ces fichiers on va créer des nouveaux répertoires.

On utilisera « **mkdir** »

6. On désire disposer de l'arborescence « projets/graphs »

Citez deux manières pour le faire.

i. **$ mkdir projets**

$ **mkdir projets/graphs**

ii. $ **mkdir –p projets/graphs**

La méthode « b » permet de creér d'un seul coup toutes les branches d'une nouvelle arborescence.

7. Créer l'arborescence projets/rapports

$ **mkdir projets/rapports**

8. Déplacer vous au sein du répertoire « rapports » puis créer le répertoire « projets/copies »

$ **cd projets/rapports**

$ **mkdir ../copies**

Les « .. » permettent de remonter dans l'arborescence Linux

9. A partir de cette étape on commencera à faire le ménage dans notre répertoire personnel en déplaçant les fichiers dans leur bon endroit.

Lister le contenu du répertoire « projets »

$ **ls projets**

copies graphs rapports

10. Déplacer en premier lieu le fichier « graph_jan » dans le sous-répertoire « graphs » puis par la suite déplacer d'un seul coup les 2 autres fichiers « graphs » restant dans le même répertoire.

$ **mv graph_jan projets/graphs**

$ **mv graph_fev graph_mar projets/graphs**

La commande « **mv** » permet de déplacer plusieurs fichiers sur la même ligne de commande

Vérifiez le contenu du sous-répertoire « graphs »

$ **ls –l projets/graphs**

11. Ensuite déplacez les deux fichiers « rapport_jan » et « rapport_fev » dans le répertoire « rapports » et supprimer le fichier « rapport_mar ».

$ **mv rapport_jan rapport_fev projets/rapports**

$ **rm rapport_mar**

12. On va créer des copies des fichiers du mois de Janvier dans le répertoire « copies » en utilisant pour l'un le chemin absolu et pour l'autre un chemin relatif.

$ **cd projets/copies**

$ **pwd**

/home/etudiant/projets/copies

$ **cp ../rapports/rapport_jan .**

$ **cp /home/etudiant/projets/graphs/graph_jan .**

Le point à la fin est la destination : le répertoire de travail courant.

Fin

TP N° 3

- **Manipulation Redirections < , << , > et >> La commande cat**
- **Les commandes Tail head**
- **Les commandes sort et uniq**

1. Redirections > et >>

Linux, comme tout système de type Unix, possède des mécanismes permettant de rediriger les entrées-sorties standards vers des fichiers.

Ainsi, l'utilisation du caractère «>» permet de rediriger la sortie standard d'une commande située à gauche vers le fichier situé à droite :

a- Se positionner dans votre maison (Home) par : cd ~
b- Taper **ls -al /bin/ > toto.txt**
c- Taper **echo "bonjour mon ami Toto" > $HOME/toto.txt**
d- La commande suivante est équivalente à une copie de fichiers :
cat toto.txt > toto2.txt
cp toto.txt toto3.txt

Comparer **toto2.txt et toto3.txt ?**

La redirection **«>»** a pour but de créer un nouveau fichier. Ainsi, si un fichier du même nom existait, celui-ci sera écrasé. La commande suivante crée tout simplement un fichier vide :

e- **> fichier1.txt**

L'emploi d'un double caractère «>>» permet de concaténer la sortie standard vers le fichier, c'est-à-dire ajouter la sortie à la suite du fichier, sans l'écraser.

f- Taper les commandes suivantes :

echo " ce ci est le contenu d'un fichier : son non est Fich1.txt " > fich1.txt

cat fich1.txt

echo " un deuxieme fichier est sera cree : et son nom sera Fich2.txt " > fich2.txt

more fich2.txt

cat fich1.txt >> **fich2.txt**

more fich2.txt

qu'est ce qu est arrive au fichier : fich2.txt ?

g- afficher à l'aide de **more** ou **less** ou **cat** le contenu de fichier fich2.txt
Quesque que vous remarquez ?

2. Redirections < et <<

De manière analogue, le caractère «<» indique une redirection de l'entrée standard. La commande suivante envoie le contenu du fichier toto.txt en entrée de la commande cat, dont le seul but est d'afficher le contenu sur la sortie standard (exemple inutile mais formateur) :

cat < toto.txt

Enfin l'emploi de la redirection «<<» permet de lire sur l'entrée standard jusqu'à ce que la chaîne située à droite soit rencontrée. Ainsi, l'exemple suivant va lire l'entrée standard jusqu'à ce que le mot STOP soit rencontré, puis va afficher le résultat :

cat << STOP

3. Les Commandes tail , head , sort et uniq

a- A l'aide la commande cat et de flux de redirection (<<, et > remplir le fichier « fil_tmp.txt) par le texte suivant :

tt
g13
jj
bbbbbb
bb
122
66
3
115
jj
Hh
Nk
Jdjd
Yy
Tt
g13
Ze

b- Afficher le contenu de 3 premières lignes
c- Afficher le contenu des 2 dernières lignes
d- Trier et afficher le contenu de fil_tmp.txt selon l'ordre alphabétique
e- Trier et afficher le contenu de fil_tmp.txt selon l'ordre numérique
f- Constater la différence entre « d » et « e »
g- Renverser le trie
h- Créer les fichiers suivant :
   - **fil_tmp_sans_doub.txt** : ayant le contenu de **fil_tmp.txt** sans **doublons**
   - **fil_tmp_doub.txt** : contenant les lignes en double dans **fil_tmp.txt** .
   - **fil_tmp_trie_num.txt** : ayant le contenu de **fil_tmp.txt** trier selon l'ordre numérique

- **fil_tmp_trie_alpha.txt** : ayant le contenu de **fil_tmp.txt** trier selon l'ordre alphabétique

Fin

TP 4

- **Gestion des liens**
- **La commande find**
- **Expression**
- **Rédaction d'un Compte Rendu**

Exercice 1 : **Liens Symboliques**

1. Créer sur votre répertoire maison un lien symbolique ***monpass*** vers le fichier ***/etc/passwd.***

   $ ln -s /etc/passwd monpass

2. Saisissez la commande suivante

   $ ls –li monpass /etc/passwd

Faites les remarques suivantes :

a) Chaque fichier a son propre numéro d'inode→un lien symbolique est un fichier différent de l'original.

b) Le 1er caractère de la longue liste pour un symlink est la lettre l

c) Remarquez les permissions particulières du lien symbolique

d) Remarquez la taille du lien symbolique, c'est 11 vu que la chaîne « /etc/passwd » est composé de 11 caractères

Exercice 2 : **Liens Physiques**

1. Créer un fichier linux : **$ touch linux**

2. Créer un lien physique fedora sur le fichier linux : **$ ln linux fedora**

3. Saisissez la commande qui suit :

$ ls –li linux fedora

4. Faîtes les remarques suivantes :

a) Les deux fichiers linux et fedora ont exactement le même **inode**.

b) Il existe un seul fichier sous-jacent, mais il existe 2 points d'entrée

c) linux et fedora sont tous deux des fichiers réguliers

d) Le nombre de liens a été incrémenté à 2 car deux noms de chemins pointent vers le même fichier

Exercice 3 :

1. Créer un fichier ***test*** contenant la ligne **Ceci est un fichier de test** .

2. Créer une copie de ce fichier, ***test1***, puis un lien ***test2*** et un lien symbolique ***test3*** sur test.

3. Comparer les numéros d'inodes de ces quatres fichiers

4. Effacer ***test*** puis afficher le contenu de ***test1***, ***test2*** et ***test3***.

Préciser l'action de rm.

5. Créer un nouveau fichier ***test*** contenant la ligne **Ceci est un deuxième fichier test**. Afficher à nouveau le contenu des quatres fichiers. Commentez ?

Exercice 4 : **Manipulation de la commande find**

1. Chercher dans l'arborescence ***« / »*** tous les fichiers et répetoires de nom ***« bash »***

   $ find / -name bash –print

Cette commande parcourera tout le système de fichiers ***« / »***.

Si vous lancez cette commande en tant qu'un utilisateur autre

que le ***« root »*** il est normal que des messages d'erreur de type

suivant apparaîssent vu les restrictions de droits sur certains

fichiers auxquels seul l'utilisateur ***« root »*** qui peut y accéder.

find: /var/spool/cron: Permission non accordée

2. Rechercher tous les fichiers du systèmes se terminant avec la
   Chaîne « sh »

   a- **$ find / -name *sh**
   b- $ find / -name *sh –print

   Faîtes attention au caractère **<\>** qui permet la non interprétation du **<*>** par le shell.

3. Executez une commande file sur tous les fichiers de ***/etc/mail***

   $ find /etc/mail –exec file {} \ ;

4. chercher et Afficher en format long tout les fichiers appartenant à l'utilisateur ESTM

5. chercher et Afficher en format long les fichiers de vote maison datant plus de 20 jours

6. Chercher et Afficher en format long dans **/usr** les fichiers dont la taille dépasse 1Mo

7. Chercher et Afficher en format long dans **/bin** les fichiers dont les droits sont fixés à 755

8. Chercher dans le répertoire courant tout les fichiers qui commence par un caractère numérique

Exercice 5 **Expressions arithmétiques**

1- Ecrire la suite des commandes suivantes :

```
$n=1
$echo $(( n + 1 ))
$ p = $(( n * 5 / 2 ))
$ echo $p
```

2- Soit a= 6 , b=21 et c=9 à l'aides des expressions arithmétiques :

Calculer et afficher la somme de a et b.
Calculer et afficher la moyenne de de a , b et c.
Calculer et afficher la moyenne de de a , b et c.

A la fin rédiger un compte rendu pour tout ce TP

Solution

4 find . –user student –exec ls –l {}\; (affiche le format long de la commande ls pour les fichiers trouvés)

5 **find . –type f –atime +13 –exec rm {} \;**

6 find /usr -size + 10M -print 2>/dev/null

7 find /bin –type f -perm 755 –exec ls –lrt {} \;

8 find . –name [0-9]*

TP N°5

- **Manipulation de texte avec l'editeur vi**
- **Commande cut , sed et grep**
- **Les pipes « | »**
- **Tar et gzip**
- **Droit d'accès au fichiers**

Exercice 1

Saisie d'un document et sa manipulation avec « vi »

1. Connectez-vous en tant que l'utilisateur « estm ». Utilisez « vi » afin de créer un nouveau document :

   $ vi linux

   Vous vous retrouvez alors dans l'éditeur « vi ». Passez dans le mode ajout en appuyant sur la touche « a »

   Vous êtes maintenant en mode ajout. Tout ce que vous saisissez fera partie du document. Saisissez le dans vi aussi vite que possible. Ne vous arrêtez pas pour corriger des fautes. On y reviendra.

   Gnu/Linux est un système d'exploitation semblable à Unix, autant similaire que les nombreuses versions d'Unix le sont entre elles. Sur un niveau conceptuel, tout ce que les autres versoins d'Unix peuvent faire, le système d'exploitation Linux peut le faire, bien que les moyens puissent légèrement changer.

   Gnu/Linux est un système d'exploitation multi-utilisateurs et multi-tâches. Cela signifie que plus d'une personne peut être connectée sur le même ordinateur Gnu/Linux au même moment. Gnu/Linux est également multi-tâche : un utilisateur peut avoir plus d'un processus executé au même moment.

2. Enregistrez cette première version en quittant l'éditeur :

   <Echap>:wq

3. Editez de nouveau le document :

$ vi linux

Maintenant examinez l'extrait avec attention. Recherchez des fautes d'orthographe et trouvez la première. Déplacer le curseur jusqu'au début du mot épelé incorrectement à l'aide des commandes **h, j, k** et **l** pour déplacer votre curseur à gauche, en bas, en haut et à droite, respectivement.

Vous pouvez également utiliser la commande **w** pour avancer d'un mot à la fois. Corrigez le mot. Par exemple, si vous avez mal écrit le mot « versions », positionner vous au début du mot et remplacer-le comme suit :

jjjwwwwwwwww

ou en étant plus fantaisiste :

3j9w

Cette séquence vous amènera au bon mot, en supposant que vous avez saisi le document donné. Il est évident que des lignes, des mots ou des espaces supplémentaires (ou inférieurs) changeront cette séquence.

Corriger le mot à l'aide de la commande :

cwversions**<Echap>**

4. Examinez chaque ligne, une à une, trouvant d'autres erreurs. Essayez d'utiliser la fonction de recherche pour déplacer le curseur vers un mot mal épelé. Par exemple, si « légèrement » est mal épelé, recherchez le mot ou une partie du mot. Par exemple : /léger**<Entrée>**

5. Trouvez les autres ereurs. Corrigez-les à l'aide des commandes comme **cw** pour changer un mot, **dw** pour supprimer des mots en trop, **dd** pour supprimer des lignes, **cl** pour changer des lettres individuelles. Souvenez-vous d'utililser l'utile commande **u** pour annuler des changements faits par erreur.

6. Enregistrez cette version en quittant l'éditeur :

:wq

Exercice 2 (cut , grep et sed)

1. Taper le texte suivant et enregistrer le dans un fichier de nom **resutlat.txt**

nom,	filière,	note,	Commentaire
taha,	AAI,	11.5/20,	fais plus d'effort!
tata,	GI,	14.0/20,	Bien
toto,	AAI,	15.5/20,	résultat encourageant
titi,	IEEA,	13.0/20,	assez bien
tantan,	AII,	17.0/20,	très bien
tintin,	IEEA,	14.5/20,	Bien

2. Afficher le nombre de ligne de fichier **resutlat.txt**
3. Utiliser la commande **cut** pour :

- afficher uniquement les caractères 2 à 5 de chaque ligne de fichier resutlats.txt
- afficher du 1er au 3ème caractère de chaque ligne de fichier resutlats.txt
- afficher du 3ème au dernier caractère de chaque ligne de fichier resutlats.txt

3- Utiliser encore la commande **cut** (penser aux options –d –f et le séparateur) pour :

- Afficher juste les filières
- Afficher juste les filières sans doublons

- Afficher dans un fichier notes_nom.txt les noms et les notes de chaque étudiant

4- En utilisant la commande **grep** sur le fichier resulatat.txt

- Créer un fichier reslat_ta.txt ne contenant que les lignes qui commence par « **ta** » dans le fichier resulatat.txt

- Créer un fichier reslat_sans_ta.txt ne contenant que les lignes qui ne commence pas par « **ta** » dans le fichier resulatat.txt

5- Créer un répertoire **essai-grep** dans votre home directory. Dans ce répertoire créer les fichiers suivants:

tomate poire pomme cerise Fraise fraise courgette POMME3 afraise pomMmier

- Combien de fichiers existent sur votre répertoire home (**utiliser ls + pipe (|)**)
- Editez les fichiers (sortie de la commande ls redirigée vers grep) avec les critères sur leur nom suivant: plus le **grep** utiliser aussi ls + **pipe (|))**

Critère 1	Le nom doit être Fraise ou fraise
Critère 2	« se » est en fin de nom
Critère 3	« ai » est présent dans le nom
Critère 4	Nom contenant un chiffre numérique
Critère 5	Nom contenant la chaîne mm ou MM
Critère 6	Nom contenant la chaîne de 3 « **m »** (minuscule ou majuscules) le résultat sera **pomMmier**

6- En utilisant la commande **sed** sur le fichier linux que vous avez creé à l'exercice 1 :

- changer la 1re occurrence de la chaîne multi par MULTI
- changer toutes les occurrences de la chaîne GNU/Linux par Gnu-LINUX
- enregistrez ce dernier résultat dans un fichier à part en utilisant deux méthodes

- **Exercice 3 (archivage tar)**

Le fichier **fich_archive.tar** est une archive au format tar non compressé contenant les hardcopies de fenêtres Ubuntu-KDE. Cette archive n'inclut pas de répertoire de décompression.

1- « détarer » cette archive dans un sous répertoire nommé « fich_archive» de votre HOME.

2- « tarer » le répertoire « fich_archive» en une archive nommée « MonArchive.tar ».

3- « tarer » le répertoire « fich_archive» en une archive compressée nommée « MonArchiveCompressee.tar.gz ».

4- Quel est le rapport de tailles entre ces deux archives?

5- Calculer ce rapport de tailles pour une archive de fichiers texte faites ça sur un fichier test

Exercice 4 (droit d'acces chmod)

RQ : *les questions de 1 à 5 peuvent être faites de deux façons :*

- *La notation décimale ou En utilisant les lettre w,r,x, et o, g, u, a, choisissez la bonne pour chaque question*

1. Créez un répertoire Linux et déplacez-vous dans celui-ci

2. Créez le fichier vide mon_fichier, et examinez ensuite ses permissions.

3. Donnez-lui successivement les droits nécessaires pour que vous puissiez.

a. Lire, modifier et exécuter votre fichier.

b. Lire, modifier mais ne pas exécuter votre fichier.

c. Lire mais ne pas modifier ou exécuter votre fichier.

4. Accordez maintenant toutes les permissions au propriétaire et la lecture seulement pour le groupe.

5. Expliquez. ce que vous pouvez faire sur ce fichier si changez de session (changez d'utilisateur)

6- Comment faire pour avoir les droits par défaut 644 pour les fichiers et 755 pour les répertoires lors de la création

Réponses

1- mkdir Fichiers puis tar -xvf Fichiers.tar -C Fichiers

2- tar -cvf MonArchive.tar Fichiers
3- tar -zcvf MonArchiveCompressee.tar.gz Fichiers
4- L'archive non compressée occupe 1003520 octets. L'archive compressée occupe 908384 octets. Le gain est donc de 9,48%.
5- Sur un exemple test on obtient une taille non compressée de 2580480 octets pour une taille compressée de 519117 octets. Le gain est donc de 79,88%.

Solution1*:*

mkdir ~/essai-grep

cd ~/essai-grep

touch tomate poire pomme cerise Fraise fraise courgette POMME3 afraise ,FRAISE

Critère 1	ls | grep "^[fF]raise$"
Critère 2	ls | grep "se$"
Critère 3	ls | grep "ai"
Critère 4	ls | grep "[0-9]"
Critère 5	ls | grep "[mM]\{2\}"

1. Créez un répertoire Linux et déplacez vous dans celui-ci

mkdir Linux

cd Linux

2. Créez le fichier vide mon_fichier, et examinez ensuite ses permissions.

touch mon_fichier

ls -l mon_fichier

3. Donnez-lui successivement les droits nécessaires pour que vous puissiez.

a. Lire, modifier et exécuter votre fichier.

chmod u+rwx mon_fichier

b. Lire, modifier mais pas exécuter votre fichier.

chmod u-x mon_fichier

c. Lire mais pas modifier ou exécuter votre fichier.

chmod u-w mon_fichier

4. Accordez maintenant toutes les permissions au propriétaire et la lecture seulement pour le groupe.

chmod 740 mon_fichier

5. Expliquez. ce que vous pouvez faire sur ce fichier si changez de session (changez d'utilisateur)

Vous ne pouvez pas consulter le contenu de ce fichier, mais vous pouvez le voir dans la liste des fichiers si vous listez le contenu du répertoire