CTP – R108 : Bases des systèmes d'exploitation (Compte rendu individuel)

Exercice 1 — Fichiers et arborescence (2 pts)

Créez dans votre répertoire personnel un dossier nommé `Examen_R108`.
À l'intérieur, créez les sous-dossiers `work` et `result`.
Commande utiliser : **mkdir -p ~/Examen_R108/work ~/Examen_R108/result**
Créez dans `work` trois fichiers (`t1.txt`, `t2.txt`, `t3.txt`) contenant chacun une phrase différente.
leo_wouters@j004-09:~$ **echo "Blabla1" > ~/Examen_R108/work/t1.txt**
leo_wouters@j004-09:~$ **echo "albalb2" > ~/Examen_R108/work/t2.txt**
leo_wouters@j004-09:~$ **echo "Une phrase différente3" > ~/Examen_R108/work/t3.txt**
Copiez tous les fichiers `.txt` de `work` vers `result`.
leo_wouters@j004-09:~$ **cp ~/Examen_R108/work/* ~/Examen_R108/result**
Affichez la structure complète du dossier `Examen_R108` avec les chemins absolus.
leo_wouters@j004-09:~$ **ls -R ~/Examen_R108/**
Résultat :
/home/leo_wouters/Examen_R108/:
result work

/home/leo_wouters/Examen_R108/result:
t1.txt t2.txt t3.txt

/home/leo_wouters/Examen_R108/work:
t1.txt t2.txt t3.txt
📝 *Indiquez les commandes utilisées (`mkdir`, `cp`, `ls -R`, `pwd`, etc.) et précisez à quoi chacune sert.*
mkdir → Créer un dossier (-p (le parent))
cp → Copie colle
ls -R → Affiche la structure complete d'un dossier
echo → permet d'écrire et de créer le fichier si inexistant.

Exercice 2 — Recherche et critères avec find et grep (2 pts)

Recherchez dans le répertoire `/bin` tous les fichiers exécutables dont le nom contient la lettre **s**.

Affichez uniquement les **5 premiers** résultats.

leo_wouters@j004-09:~$ **find /bin -name "s*" | head -n 5**

(Aucun résultat..vide?)

1. Sauvegardez ces résultats dans le fichier `executables.txt` dans le dossier `result`.

   find /bin -name "*s*" | head -n 5 > ~/Examen_R108/result/executables.txt
   (Aucun résultat..vide?)

2. Ajoutez à la fin du fichier la date du jour (commande `date`) et votre nom d'utilisateur (`whoami`).

   echo "$(date) $(whoami)" >> ~/Examen_R108/result/executables.txt

📝 Expliquez la différence entre une recherche par **nom de fichier** (`find`) et une recherche par **contenu** (`grep`).

Find va trouver les fichiers dans un dossier tandis que Grep va chercher le contenu d'un fichier.

Explication commande utiliser :

head -n 5 (les 5 premières lignes)

>> → ça vient rajouter au fichier sans écraser !

Exercice 3 — Droits, comparaisons et redirections (2 pts)

1. Dans le dossier `Examen_R108`, créez un fichier `identite.txt` contenant votre **nom**, **prénom** et **adresse mail**.

   leo_wouters@j004-09: **echo -e "Nom: Wouters\nPrénom: Leo\nEmail: Leo@gmail.com" > ~/Examen_R108/identite.txt**

   leo_wouters@j004-09:~/Examen_R108$ cat identite.txt

   Nom: Wouters

   Prénom: Leo

   Email: Leo@gmail.com

2. Modifiez ses droits pour qu'il soit :

   - lisible et modifiable uniquement par le propriétaire,
   - lisible par le groupe,
   - invisible pour les autres.

   chmod u+x ~/Examen_R108/identite.txt

3. Créez une copie `identite_copie.txt` et comparez les deux fichiers avec `diff`.

   cp ~/Examen_R108/identite.txt ~/Examen_R108/identite_copie.txt

4. Enregistrez le résultat de la comparaison dans `comparaison.log`.

   leo_wouters@j004-09: **diff ~/Examen_R108/identite.txt ~/Examen_R108/identite_copie.txt > ~/Examen_R108/comparaison.txt**

5. Rediriger toute erreur éventuelle vers un fichier `erreurs.txt` (en l'ajoutant à la suite du fichier s'il existe déjà).

   chmod u+x ~/Examen_R108/comparaison.log

   ~/Examen_R108/comparaison.log >> ~/Examen_R108/erreur.txt

📝 Expliquez brièvement la signification des **trois chiffres** dans un `chmod` (exemple : 640). **chmod 644 définit les autorisations pour le fichier comme. -rw-r--r-- ce qui signifie que l'utilisateur a l'autorisation de lire et d'écrire et que tous les autres ont l'autorisation de lecture seule. le premier pour les autorisations du propriétaire, le deuxième pour le groupe et le troisième pour tous les autres utilisateurs du système.**

Exercice 4 — Filtres, tri et traitement de texte (2 pts)

Créez un fichier `employes.txt` contenant les lignes suivantes :

Alice 2300

Karim 1900

Sophie 2500

Hugo 1800

Alice 2300

Léa 2100

```
echo -e "Alice 2300\nKarim 1900\nSophie 2500\nHugo 1800\nAlice 2300\nLéa 2100" > ~/Examen_R108/employes.txt
```

1. Triez le fichier selon la deuxième colonne (salaire) par ordre croissant.

   ```
   sort -k2 -n ~/Examen_R108/employes.txt
   ```

2. Affichez uniquement les noms sans doublons.

   ```
   cut -d' ' -f1 ~/Examen_R108/employes.txt | sort | uniq
   ```

3. Affichez le nombre total de lignes du fichier original.

   ```
   wc -l ~/Examen_R108/employes.txt
   ```

4. Cherchez toutes les lignes contenant la lettre “e” (insensible à la casse).

   ```
   grep -E "e" ~/Examen_R108/employes.txt
   ```

📝 Donnez toutes les commandes utilisées (`sort`, `uniq`, `wc`, `grep -i`, `cut`, etc.) et précisez le rôle de chacune.

sort → trie les lignes dans les fichiers

-k2 → pour la deuxième colonne

cut → extrait des sections de chaque ligne

uniq → pas de doublons

wc -l → affiche le nombre total de ligne

grep → viens nous chercher uniquement ce que qu'on demande ("e")

Exercice 5 — Variables, scripts et gestion des processus (2 pts)

1. Créez une variable `LOGIN` contenant votre nom d'utilisateur et affichez sa valeur.

   LOGIN=$USER; echo $LOGIN

2. Créez un script `infos_user.sh` qui :

   nano ~/Examen_R108/infos_user.sh

   - affiche le nom d'utilisateur courant et le shell utilisé ;
   - affiche la date et l'heure du jour ;

demande à l'utilisateur son âge et affiche un message personnalisé :

Bonjour <user>, vous avez <âge> ans. Nous sommes le <date>.

```
echo "Nom d'utilisateur : $USER"
echo "Shell utilisé : $SHELL"
echo “Date et heures : $(date)”
read -p "Quel est votre âge ? " age
echo "Bonjour $USER, vous avez $age ans. Nous sommes le $(date)"
```

3. Rendez ce script exécutable et exécutez-le.

   chmod u+x ~/Examen_R108/infos_user.sh

   ./Examen_R108/infos_user.sh

4. Lancez une commande `sleep 45` en arrière-plan, affichez son PID avec `ps`, puis terminez-la proprement avec `kill`.

   Commandes :

   sleep 45 &

   ps aux | grep sleep

   kill %1

5. Vérifiez qu'elle a bien été arrêtée.

   leo_wouters@j004-09:~/Examen_R108$ ps aux | grep sleep

   leo_wou+ 2807723 0.0 0.0 9104 2340 pts/0 S+ 09:38 0:00 grep sleep

   [1]+ Complété sleep 45

📝 Incluez le contenu complet du script (`cat infos_user.sh`) et les commandes liées au processus (`ps`, `kill`, `jobs`, etc.).

Script infos_user.sh :

leo_wouters@j004-09:~/Examen_R108$ cat infos_user.sh

echo "Nom d'utilisateur : $USER"

echo "Shell utilisé : $SHELL"

echo "Date et heures : $(date)"

read -p "Quel est votre âge ? " age

echo "Bonjour $USER, vous avez $age ans. Nous sommes le $(date)"

Commandes utilisé :

& → process en cours d’execution en Arrière Plan.

ps → “process status”, et aux → les processus en cours d'exécution.

kill%1 → vient finir le processus numéro [1].