25/01/2024

Automatisation des tâches d'administration

R405

MR OUAMRI MOHAMED AMINE
IUT DE VILLETANEUSE P13

EXERCICE 1 - Commandes de base

1.Pour chaque commande, décrivons en une phrase ce qu'elle fait et indiquer le rôle des options indiquées entre crochets .

cd : Changer de répertoire

- **mkdir [-p]** : Créer un répertoire

  L'option facultative -p permet la création de répertoires parentaux s'ils n'existent pas.

- **rmdir** : Supprimer le répertoire
- **pwd** : Chemin du répertoire actuel

  Affiche le chemin du répertoire de travail actuel.

- **man** : Manuel

  Affiche le manuel d'une commande.

- **ls [-l] [-a] [-R] [-1]** : Affiche une liste de fichiers et de répertoires.

  Les options incluent :

  - **-l** : montrant des détails supplémentaires.
  - **-a** : Inclure les fichiers cachés.
  - **-R** : Liste de manière récursive les sous-répertoires.
  - **-1** : Liste un fichier par ligne.

- **rm [-i] [-r]** : Supprime des fichiers ou des répertoires.

Les options incluent :

  - **-i** : Mode interactif, demande confirmation avant la suppression.
  - **-r** : Supprime de manière récursive les répertoires et leur contenu.

- **cp [-i] [-r] [-a]** : Copie des fichiers ou des répertoires

  Les options incluent :

  - **-i** : Mode interactif, demande confirmation avant l'écrasement.
  - **-r** : Copie de manière récursive les répertoires et leur contenu.
  - **-a** : Préserve les attributs des fichiers lors de la copie.

- **mv [-i]** : Déplace ou renomme des fichiers ou des répertoires.

  -i demande confirmation avant l'écrasement.

- **date** : Afficher ou définir la date et l'heure du système.
- **echo [-n]** : Afficher un message, imprime un message dans le terminal.

  L'option -n exclut le saut de ligne, maintenant le curseur sur la même ligne.

2- Pour gagner du temps lors des différents TP, organisez bien vos fichiers. Une sauve-

garde personnelle sur clé USB pourra vous servir. Vous devez avoir dans votre répertoire de connexion les répertoires suivants

3- Commande man

```
SYNOPSIS
       less -?
       less --help
       less -V
       less --version
       less [-[+]aABcCdeEfFgGiIJKLmMnNqQrRsSuUVwWX~]
            [-b space] [-h lines] [-j line] [-k keyfile]
            [-{oO} logfile] [-p pattern] [-P prompt] [-t tag]
            [-T tagsfile] [-x tab,...] [-y lines] [-[z] lines]
            [-# shift] [+[+]cmd] [--] [filename]...
       (See  the  OPTIONS section for alternate option syntax with long option
       names.)
```

4- Commande ls

1. Liste simple.

```
sina@p20102:~$ ls
Bureau  Documents  Public  Téléchargements  test  test~  test.txt  trash
```

2. Liste montrant les fichiers cachés (ceux dont le nom commence par ”.”). On remarquera la présence des 2 entrées ”.” et ”..”.

```
sina@p20102:~$ ls -a
.              .cache      .ICEauthority  Public           test.txt
..             .config     .lesshst       .ssh             trash
.bash_logout   .dmrc       .local         Téléchargements  .wget-hsts
.bashrc        Documents   .mozilla       test             .Xauthority
Bureau         .gnupg      .profile       test~            .xsession-errors
sina@p20102:~$
```

3. Liste avec descriptif complet de chaque référence (droits, nombres de liens, dates, taille user group ...).

```
sina@p20102:~$ ls -l
total 32
drwxr-xr-x 2 sina sina 4096 24 janv. 14:53 Bureau
drwxr-xr-x 2 sina sina 4096 25 janv.  2022 Documents
drwxr-xr-x 2 sina sina 4096 25 janv.  2022 Public
drwxr-xr-x 2 sina sina 4096 24 janv. 14:43 Téléchargements
-rw-r--r-- 1 sina sina    8 24 janv. 15:07 test
-rw-r--r-- 1 sina sina    8 24 janv. 15:05 test~
-rw-r--r-- 1 sina sina    8 24 janv. 15:07 test.txt
drwxr-xr-x 2 sina sina 4096 25 janv.  2022 trash
sina@p20102:~$
```

4. Liste avec descriptif complet et avec un format plus compréhensibles concernant la taille des fichiers.

```
sina@p20102:~$ ls -lh
total 32K
drwxr-xr-x 2 sina sina 4,0K 24 janv. 14:53 Bureau
drwxr-xr-x 2 sina sina 4,0K 25 janv.  2022 Documents
drwxr-xr-x 2 sina sina 4,0K 25 janv.  2022 Public
drwxr-xr-x 2 sina sina 4,0K 24 janv. 14:43 Téléchargements
-rw-r--r-- 1 sina sina    8 24 janv. 15:07 test
-rw-r--r-- 1 sina sina    8 24 janv. 15:05 test~
-rw-r--r-- 1 sina sina    8 24 janv. 15:07 test.txt
drwxr-xr-x 2 sina sina 4,0K 25 janv.  2022 trash
sina@p20102:~$
```

5. Liste récursive (descend dans les sous-répertoires).

```
sina@p20102:~$ ls -R
.:
Bureau  Documents  Public  Téléchargements  test  test~  test.txt  trash

./Bureau:
'TP1 R405.odt'

./Documents:

./Public:

./Téléchargements:
Cours-IntroSE-M3206-4.pdf  TP-Automatisation_des_taches.pdf

./trash:
oracle_vbox_2016.asc
sina@p20102:~$
```

6. Liste par ordre chronologique (la commande touch peut servir à changer la date de modification d'un fichier).

```
sina@p20102:~$ ls -lt
total 32
-rw-r--r-- 1 sina sina    8 24 janv. 15:07 test.txt
-rw-r--r-- 1 sina sina    8 24 janv. 15:07 test
-rw-r--r-- 1 sina sina    8 24 janv. 15:05 test~
drwxr-xr-x 2 sina sina 4096 24 janv. 14:53 Bureau
drwxr-xr-x 2 sina sina 4096 24 janv. 14:43 Téléchargements
drwxr-xr-x 2 sina sina 4096 25 janv.  2022 trash
drwxr-xr-x 2 sina sina 4096 25 janv.  2022 Documents
drwxr-xr-x 2 sina sina 4096 25 janv.  2022 Public
sina@p20102:~$
```

7. Liste par date d'accès au lieu de la date de création. Pour constater un changement, utiliser la commande cat "nom de fichier" pour modifier la date du dernier accès.

```
sina@p20102:~$ ls -lu
total 32
drwxr-xr-x 2 sina sina 4096 24 janv. 15:03 Bureau
drwxr-xr-x 2 sina sina 4096 24 janv. 14:52 Documents
drwxr-xr-x 2 sina sina 4096 24 janv. 15:29 Public
drwxr-xr-x 2 sina sina 4096 24 janv. 15:29 Téléchargements
-rw-r--r-- 1 sina sina    8 24 janv. 15:07 test
-rw-r--r-- 1 sina sina    8 24 janv. 15:03 test~
-rw-r--r-- 1 sina sina    8 24 janv. 15:07 test.txt
drwxr-xr-x 2 sina sina 4096 24 janv. 15:29 trash
```

8. Liste simple du contenu avec affichage du type de fichier (répertoire /, lien symbolique @, exécutable *).

```
sina@p20102:~$ ls -F
Bureau/  Documents/  Public/  Téléchargements/  test  test~  test.txt  trash/
sina@p20102:~$
```

9. Liste avec numéro inode. (vous pouvez vérifier en créant avec la commande ln un lien physique vers un fichier existant).

```
sina@p20102:~$ ls -i
271030 Bureau     271033 Public           266170 test   266186 test.txt
271034 Documents  271031 Téléchargements  266166 test~  271014 trash
sina@p20102:~$
```

EXERCICE 2 – Redirections

1- A l'aide d'une redirection et de la commande echo, créez un fichier contenant la ligne de texte : « Bonjour ».

```
sina@p20102:~$ echo "Bonjour" > test.txt
sina@p20102:~$ nano test
sina@p20102:~$ cat test
Bonjour
sina@p20102:~$
```

2- Ajouter la ligne de texte « Hello » au fichier précédemment créé

```
sina@p20102:~$ echo "Hello" >> test
sina@p20102:~$ echo "Hello" >> test.txt
sina@p20102:~$ cat test
Bonjour
Hello
sina@p20102:~$
```

3- Que fait la commande wc ? Et wc -l

wc (word count) affiche le nombre de lignes ,le nombre d'octets ,le nombre de mots
wc -l'affiche juste les lignes

Illustration

```
sina@p20102:~$ wc test
 2  2 14 test
```

```
sina@p20102:~$ wc -l test
2 test
sina@p20102:~$
```

4-5 A l'aide des commandes ls et wc (avec options si besoin) et d'une redirection vers un fichier crée dans le répertoire /tmp, faire afficher le nombre de total de fichiers (ou répertoires) présents dans le répertoire /etc.

```
sina@p20102:~$ touch yess
sina@p20102:~$ ls -A /etc | wc -l > /tmp/yess.txt
sina@p20102:~$
```

EXERCICE 3 - Trouvez les informations suivantes et notez-les sur votre compte-rendu :

1. Quelle est l'adresse IP de votre machine ?
Commande utilisée : ip a
Résultat :

```
sina@p20102:~$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue sta
t qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
```

2. Quelle est l'adresse IP du DNS principal ?
Fichier ou commande utilisé(e) : dig ou cat/etc/resolv.conf
Résultat :

```
sina@p20102:~$ cat /etc/resolv.conf
# Generated by NetworkManager
search iutv.univ-paris13.fr
nameserver 192.168.1.20
```

```
;; Query time: 4 msec
;; SERVER: 192.168.1.20#53(192.168.1.20)
```

3. Quelle est le nom de votre machine ?
Fichier ou commande utilisé(e) : hostname
Résultat :

```
sina@p20102:~$ hostname
p20102
sina@p20102:~$
```

4. Quelle est la mémoire vive installée ?
Fichier ou commande utilisé(e) : free -h
Résultat :

```
sina@p20102:~$ free -h
               total        utilisé      libre     partagé tamp/cache   disponible
Mem:           7,4Gi       2,4Gi       3,7Gi       115Mi       1,3Gi       4,7Gi
Partition d'échange:      351Gi          0B       351Gi
sina@p20102:~$
```

5. Quelle est votre répertoire de connexion (HOME) ?
Fichier ou commande utilisé(e) : echo $HOME
Résultat :

```
sina@p20102:~$ echo $HOME
/home/sina
```

6. Quelle est l'espace disque disponible sur ce répertoire ?
Fichier ou commande utilisé(e) : df -h $home
Résultat :

```
sina@p20102:~$ df -h $home
df: /run/user/1000/doc: Opération non permise
Sys. de fichiers Taille Utilisé Dispo Uti% Monté sur
udev                3,7G       0  3,7G   0% /dev
tmpfs               762M    1,6M  761M   1% /run
/dev/nvme0n1p2      123G    5,1G  112G   5% /
tmpfs               3,8G       0  3,8G   0% /dev/shm
tmpfs               5,0M    4,0K  5,0M   1% /run/lock
/dev/nvme0n1p1      511M    5,8M  506M   2% /boot/efi
tmpfs               762M     60K  762M   1% /run/user/1000
```

EXERCICE 4 - Commande find

1- Afficher (avec find) les noms de tous les fichiers du répertoire /usr ayant une taille Supérieure à 250Ko.

```
magic.mgc
nl80211.h
sina@p20102:~$ find /usr -type f -size +250k -exec basename {} \;
```

```
libipcclientcerts.so
TwemojiMozilla.ttf
firefox-esr
plugin-container
libmozsqlite3.so
libfreeblpriv3.so
libmozavcodec.so
phar.so
opcache.so
fileinfo.so
git-daemon
git-imap-send
git-remote-http
git-shell
git
git-http-backend
git-http-fetch
git-sh-i18n--envsubst
git-http-push
nm-iface-helper
nm-initrd-generator
Xorg
libglx.so
```

2- Afficher les noms de tous les fichiers du répertoire /var ayant été modifiés après votre répertoire de connexion.

```
sina@p20102:~$ find /var -type f -newer $HOME
find: '/var/local': Permission non accordée
```

```
find: '/var/tmp/systemd-private-bfb878de538c4a54ae22867cd61a4698-upower
rTixai': Permission non accordée
/var/lib/systemd/timers/stamp-phpsessionclean.timer
/var/lib/systemd/timers/stamp-anacron.timer
/var/lib/systemd/timesync/clock
find: '/var/lib/docker': Permission non accordée
find: '/var/lib/containerd': Permission non accordée
find: '/var/lib/polkit-1': Permission non accordée
find: '/var/lib/apt/lists/partial': Permission non accordée
find: '/var/lib/private': Permission non accordée
find: '/var/lib/lightdm': Permission non accordée
find: '/var/lib/php/sessions': Permission non accordée
find: '/var/lib/udisks2': Permission non accordée
find: '/var/lib/NetworkManager': Permission non accordée
/var/log/Xorg.0.log
/var/log/journal/ab35fa27b248499dae879b38d0d7895b/system.journal
/var/log/auth.log
find: '/var/log/private': Permission non accordée
/var/log/daemon.log
find: '/var/log/lightdm': Permission non accordée
/var/log/syslog
```

3- A l'aide des commandes find et grep, afficher toutes les lignes contenant le mot automatic dans les fichiers d'extension .h situés dans le répertoire /usr/include et tous ses sous-répertoires.

```
sina@p20102:~$ find /usr/include -type f -name "*.h" -exec grep -H "automatic" {
} \;
/usr/include/event2/bufferevent.h:  drained automatically.  The user of a buffer
event no longer deals
/usr/include/event2/bufferevent.h:  descriptor automatically as it becomes avail
able for writing.
/usr/include/event2/bufferevent_compat.h:  input and output buffers that get fil
led and drained automatically.  The
/usr/include/event2/http.h:  automatically added.
/usr/include/event2/http.h:   is called on the main http server, it will be auto
matically freed, too.
/usr/include/event2/http.h: *           automatically on return.
/usr/include/event2/event.h:  automatically. The user of a buffered event no lon
ger deals directly
/usr/include/event2/event.h: * Persistent event: won't get removed automatically
 when activated.
/usr/include/mtd/ubi-user.h: * volume/device number they want to create or to le
t UBI automatically assign
/usr/include/mtd/ubi-user.h: * device is passed in @ubi_num. UBI may automatical
ly assign the number if
/usr/include/mtd/mtd-abi.h: * @MTD_OPS_AUTO_OOB:        OOB data are automatical
ly placed at the free areas
```

EXERCICE 5 – Métacaractères

1- Devinez ce que fait le programme Python suivant :

Le programme prend le nombre d'itérations (n) en tant que premier argument de la ligne de commande (`sys.argv[1]`).

1. Ensuite, il utilise une boucle `for` pour créer n fichiers texte.
2. Pour chaque itération, il ouvre un fichier en mode écriture avec un nom de fichier "f" suivi du numéro d'itération.
3. Il écrit le numéro d'itération suivi d'un saut de ligne dans le fichier.
4. Enfin, il ferme le fichier.

En exécutant le programme nous verrons qu'il créera cinq fichiers texte (`f0`, `f1`, ..., `f4`), chacun contenant un numéro d'itération différent suivi d'un saut de ligne.

Qu'observez-vous ? Quelle est la taille en octets des fichiers crées ? Pourquoi ?

2- A l'aide d'une seule commande shell, créez un fichier "tous" dont le contenu soit la concaténation des fichiers précédemment créés.

```
sina@LAPTOP-KASV3QIN:~$ touch fichier1
sina@LAPTOP-KASV3QIN:~$ touch fichier2
sina@LAPTOP-KASV3QIN:~$ touch fichier3
```

```
sina@LAPTOP-KASV3QIN:~$ touch fichier3
sina@LAPTOP-KASV3QIN:~$ cat fichier1 fichier2 fichier3 > tous
sina@LAPTOP-KASV3QIN:~$
```

3- Quelle est la taille du fichier tous ? Combien de lignes comporte-t-il ?

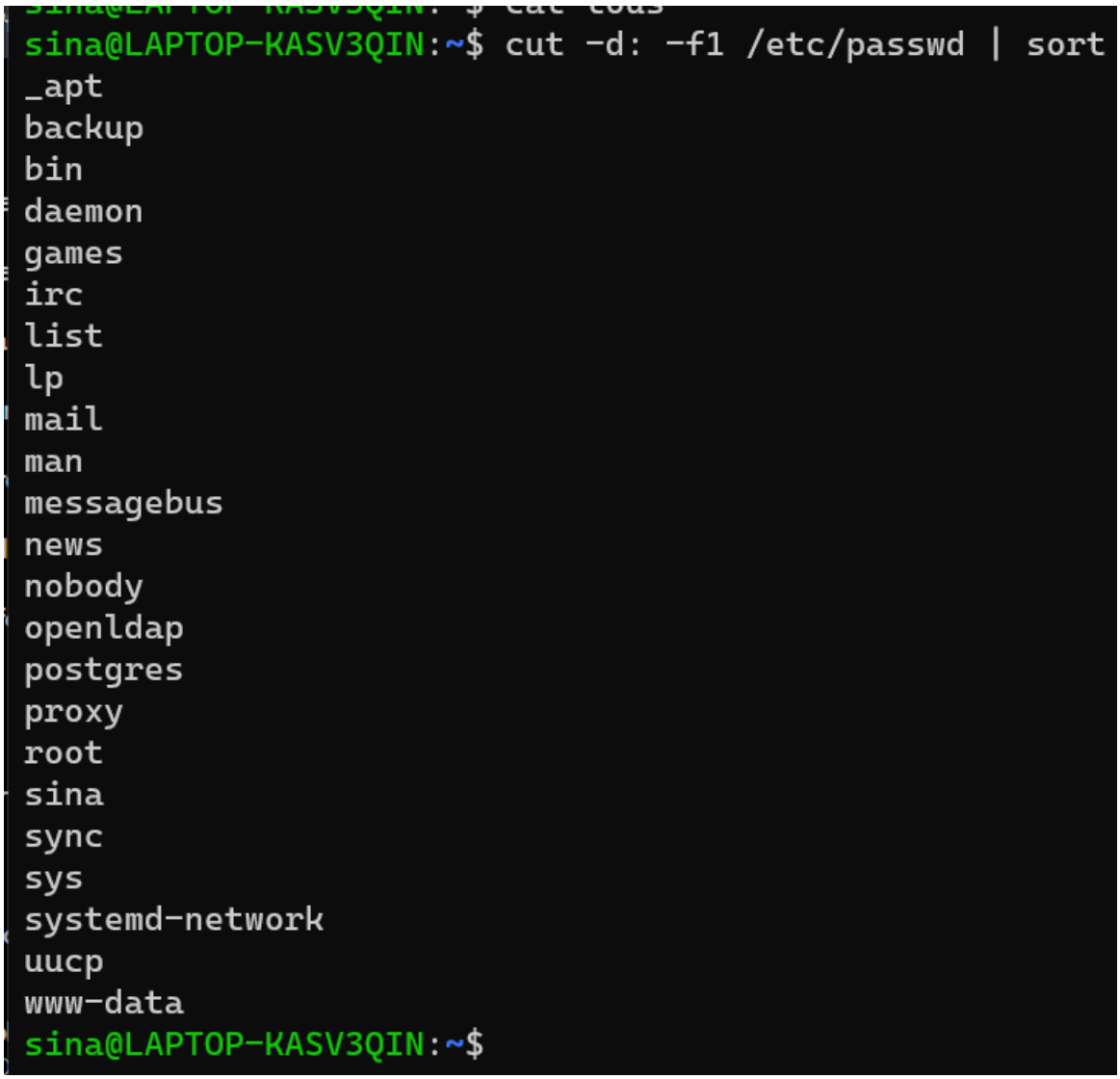
```
sina@LAPTOP-KASV3QIN:~$ du -h tous
0       tous
```

```
sina@LAPTOP-KASV3QIN:~$ wc -l tous
0 tous
```

4- A l'aide des commandes grep et wc, afficher le nombre de lignes du fichier tous qui contiennent le chiffre 1.

```
sina@LAPTOP-KASV3QIN:~$ grep '1' tous | wc -l
0
```

5- A l'aide des commandes cut et sort, afficher la liste des noms de login définis sur votre système, triée par ordre alphabétique (voir le fichier /etc/passwd).



6- Afficher les noms de tous les fichiers de /usr/include qui commencent par “std” et terminent par “.h”.

```
www-data
sina@LAPTOP-KASV3QIN:~$ find /usr/include -type f -name 'std*.h'
/usr/include/stdlib.h
/usr/include/c++/12/stdatomic.h
/usr/include/c++/12/ext/stdio_filebuf.h
/usr/include/c++/12/ext/stdio_sync_filebuf.h
/usr/include/c++/12/stdlib.h
/usr/include/c++/12/tr1/stdlib.h
/usr/include/c++/12/tr1/stdbool.h
/usr/include/c++/12/tr1/stdarg.h
/usr/include/c++/12/tr1/stdint.h
/usr/include/c++/12/tr1/stdio.h
/usr/include/c++/12/bits/std_function.h
/usr/include/c++/12/bits/std_abs.h
/usr/include/c++/12/bits/std_mutex.h
/usr/include/c++/12/bits/std_thread.h
/usr/include/linux/stddef.h
/usr/include/stdc-predef.h
/usr/include/stdint.h
/usr/include/stdio.h
/usr/include/stdio_ext.h
/usr/include/x86_64-linux-gnu/c++/12/bits/stdtr1c++.h
/usr/include/x86_64-linux-gnu/c++/12/bits/stdc++.h
/usr/include/x86_64-linux-gnu/bits/stdlib.h
/usr/include/x86_64-linux-gnu/bits/stdlib-bsearch.h
/usr/include/x86_64-linux-gnu/bits/stdlib-ldbl.h
```

EXERCICE 6 - Variables d'environnement en shell (bash)

1- Afficher la liste des variables d'environnement. Quel genre d'informations trouve-t-on ?

```
/usr/include/features.h:   not preincluded this header automatically
sina@p20102:~$ printenv
SHELL=/bin/bash
SESSION_MANAGER=local/p20102:@/tmp/.ICE-unix/39605,unix/p20102:/tmp/
605
WINDOWID=71303171
QT_ACCESSIBILITY=1
COLORTERM=truecolor
XDG_CONFIG_DIRS=/etc/xdg
XDG_SESSION_PATH=/org/freedesktop/DisplayManager/Session0
XDG_MENU_PREFIX=xfce-
SSH_AUTH_SOCK=/tmp/ssh-lkv8lkCzoC4C/agent.39605
DESKTOP_SESSION=lightdm-xsession
SSH_AGENT_PID=39647
GTK_MODULES=gail:atk-bridge
XDG_SEAT=seat0
PWD=/home/sina
LOGNAME=sina
XDG_SESSION_DESKTOP=lightdm-xsession
XDG_SESSION_TYPE=x11
XAUTHORITY=/home/sina/.Xauthority
```

2- Le shell recherche les commandes dans la liste des répertoires indiqués dans la variable d'environnement PATH.

```
sina@p20102:~$ PATH=/usr/bin:/bin:/usr/local/bin
sina@p20102:~$
```

1. Quelle est la valeur de PATH ?

```
sina@p20102:~$ echo $PATH
/usr/bin:/bin:/usr/local/bin
sina@p20102:~$
```

2. Créer (s'il n'existe pas déjà) dans votre répertoire de connexion un sous-répertoire nommé outils et y placer un exécutable (par exemple un script shell).

```
/usr/include/x86_64-linux-gnu/bits/stdio.h
sina@LAPTOP-KASV3QIN:~$ mkdir -p ~/outils
sina@LAPTOP-KASV3QIN:~$ cd ~/outils
sina@LAPTOP-KASV3QIN:~/outils$ echo '#!/bin/bash' > script.sh
sina@LAPTOP-KASV3QIN:~/outils$ echo 'echo "Hello, Voici ton script!"' >> script.sh
sina@LAPTOP-KASV3QIN:~/outils$ chmod +x script.sh
sina@LAPTOP-KASV3QIN:~/outils$
```

3. Ajouter ce répertoire ~/outils `a votre PATH.

```
sina@LAPTOP-KASV3QIN:~/outils$ echo 'export PATH=$PATH:~/outils' >> ~/.bashrc
```

4. Vérifier que vous pouvez maintenant lancer l'exécution de votre script quel que soit le répertoire courant.

```
ft VS Code/bin:/home/sina/outils:/home/sina/ou
sina@LAPTOP-KASV3QIN:~/outils$ script.sh
Hello, Voici ton script!
sina@LAPTOP-KASV3QIN:~/outils$ cd
sina@LAPTOP-KASV3QIN:~$ script.sh
Hello, Voici ton script!
sina@LAPTOP-KASV3QIN:~$ cd repertoire1/
sina@LAPTOP-KASV3QIN:~/repertoire1$ script.sh
Hello, Voici ton script!
sina@LAPTOP-KASV3QIN:~/repertoire1$
```

EXERCICE 7 - Propagation des variables d'environnement.

Etudier la s´séquence de commandes shell suivante :

```
 0    echo $ZORGLUB          ; cette var. n'existe pas !
 1    export TRUC=machin     ;  cree la variable TRUC
 2    TRAC=22
 3    echo $TRUC $TRAC       ; l'affiche
 4    bash                   ; lance un nouveau shell
 5    echo $TRUC             ; affiche la valeur de TRUC
 6    echo $TRAC             ; ?
 7    export TRUCBIS=hoho    ; une autre variable
 8    echo $TRUCBIS
 9    exit                   ; termine le second shell
10    echo $TRUC
11    echo $TRUCBIS          ; ??
```

--Que se passe-t-il lors de la première commande (ligne 0) ?

La commande echo $ZORGLUB tente d'afficher la valeur de la variable d'environnement ZORGLUB. Cependant, comme cette variable n'existe pas, seule la partie de la commande qui est statique (en dehors de $ZORGLUB) sera affichée. Dans ce cas, la phrase "cette var. n'existe pas !" sera imprimée

--Comparer avec ce qui arrive dans d'autres langages que vous connaissez si on utilise une variable qui n'existe pas.

*Bash/Shell (comme dans cet exemple) : Si vous tentez d'utiliser une variable qui n'existe pas sans la déclarer au préalable, dans un script shell ou directement dans le terminal, le comportement par défaut est de laisser la variable non définie, et cela n'entraîne généralement pas d'erreur. Vous obtiendrez simplement une chaîne vide ou un résultat dépendant de l'interpréteur shell utilisé.

*Python : En Python, si vous essayez d'utiliser une variable qui n'a pas été définie, vous obtiendrez une erreur NameError. Le programme s'arrêtera avec un message indiquant que la variable n'est pas définie.

--La commande bash (ligne 4) ouvre un nouveau shell, qui hérite des variables de l'ancien. Que s'affiche-t-il `a la ligne 11 ?

Ligne 11: echo $TRUCBIS ; ??
Cette commande tente d'afficher la valeur de la variable d'environnement TRUCBIS créée dans le nouveau shell. Cependant, cette variable n'est pas accessible dans le shell parent, donc rien ne sera affiché.

--Expliquer pourquoi.

En conséquence, à la ligne 11, la commande echo $TRUCBIS tente d'afficher la valeur de la variable TRUCBIS dans le shell parent. Cependant, cela ne fonctionnera pas car la variable TRUCBIS a été créée dans le nouveau shell et n'a pas été exportée vers le shell parent. Par conséquent, rien ne sera affiché, et la sortie sera une ligne vide ou un message similaire indiquant que la variable n'est pas définie.

En résumé, la variable TRUCBIS est limitée au nouveau shell créé à la ligne 4 et n'est pas accessible dans le shell parent après la fermeture du nouveau shell à la ligne 9.

EXERCICE 8 – **Avec la commande find,**

Ecrire une commande qui affiche le nombre fichiers présents dans un répertoire donné et tous ses sous-répertoires ainsi que leurs descendants

```
find /chemin/du/repertoire -type f | wc -l
```

EXERCICE 9 - **Révision sur les tubes**

1- Quelle est la différence entre tee et cat ?

tee est utilisé pour rediriger la sortie d'une commande vers un fichier tout en l'affichant à l'écran, tandis que cat est utilisé pour afficher ou concaténer le contenu de fichiers.

2- Que font les commandes suivantes :

--$ ls | cat : Affiche la liste des fichiers du répertoire courant.

--$ ls -l | cat > liste : Crée un fichier appelé "liste" contenant la liste détaillée des fichiers du répertoire courant.

--$ ls -l | tee liste : Affiche la liste détaillée des fichiers du répertoire courant à l'écran et l'enregistre également dans un fichier appelé "liste".

--$ ls -l | tee liste | wc -l : Affiche la liste détaillée des fichiers du répertoire courant à l'écran, l'enregistre dans un fichier appelé "liste" et affiche le nombre total de lignes, représentant le nombre de fichiers ou répertoires dans le répertoire courant.