TP13 – Introduction aux scripts shell.

Sommaire :

1. L'exécution de scripts

▪ Créez un **script** avec l'éditeur **vim** :

```
root@DS1: ~#vim un_script.sh_
```

```
#!/bin/bash
date
uptime
uname -a
```

```
root@DS1: ~#ls -l
total 40
-rw-r--r-- 1 root root 146  8 déc.  16:17 avecdoublons
-rw-r--r-- 1 root root 146  8 déc.  16:17 avecdoublons.txt
-rw-r--r-- 1 root root   0  7 déc.  22:16 cal.txt
-rw-r--r-- 1 root root   0  8 déc.  16:30 eleves.txt
-rw-r--r-- 1 root root  65  8 déc.  16:30 erreurs.log
-rw-r--r-- 1 root root  73  4 déc.  12:36 etudiants.txt
-rw-r--r-- 1 root root 211  7 déc.  22:56 notes.csv
-rw-r--r-- 1 root root 211  7 déc.  21:58 prenoms_tries
-rw-r--r-- 1 root root  73 21 déc.  20:56 prenoms_tries.txt
-rw-r--r-- 1 root root  73  8 déc.  16:19 sansdoublons.txt
-rw-r--r-- 1 root root 130  8 déc.  16:38 sio1.txt
-rw-r--r-- 1 root root  33 17 mai   10:12 un_script.sh
root@DS1: ~#
```

▪ **Exécutez**-le via le **shell** standard

```
root@DS1: ~#sh un_script.sh
ven. 17 mai 2024 10:13:30 CEST
 10:13:30 up 26 min,  1 user,  load average: 0,00, 0,01, 0,00
Linux DS1 6.1.0-12-amd64 #1 SMP PREEMPT_DYNAMIC Debian 6.1.52-1 (2023-09-07) x86_64 GNU/Linux
root@DS1: ~#_
```

▪ Exécutez le **script** en affichant la **phase d'interprétation** et la **trace** des **commandes** :

```
root@DS1: ~#bash -x un_script.sh
+ date
ven. 17 mai 2024 10:14:22 CEST
+ uptime
 10:14:22 up 27 min,  1 user,  load average: 0,00, 0,01, 0,00
+ uname -a
Linux DS1 6.1.0-12-amd64 #1 SMP PREEMPT_DYNAMIC Debian 6.1.52-1 (2023-09-07) x86_64 GNU/Linux
root@DS1: ~#
```

▪ Exécutez le **script** sous forme d'une **commande** à partir du **répertoire courant** :

```
root@DS1: ~#./un_script.sh
-bash: ./un_script.sh: Permission non accordée
root@DS1: ~#chmod +x un_script.sh
root@DS1: ~#ls -l
total 40
-rw-r--r-- 1 root root 146  8 déc.  16:17 avecdoublons
-rw-r--r-- 1 root root 146  8 déc.  16:17 avecdoublons.txt
-rw-r--r-- 1 root root   0  7 déc.  22:16 cal.txt
-rw-r--r-- 1 root root   0  8 déc.  16:30 eleves.txt
-rw-r--r-- 1 root root  65  8 déc.  16:30 erreurs.log
-rw-r--r-- 1 root root  73  4 déc.  12:36 etudiants.txt
-rw-r--r-- 1 root root 211  7 déc.  22:56 notes.csv
-rw-r--r-- 1 root root 211  7 déc.  21:58 prenoms_tries
-rw-r--r-- 1 root root  73 21 déc.  20:56 prenoms_tries.txt
-rw-r--r-- 1 root root  73  8 déc.  16:19 sansdoublons.txt
-rw-r--r-- 1 root root 130  8 déc.  16:38 sio1.txt
-rwxr-xr-x 1 root root  33 17 mai   10:12 un_script.sh
root@DS1: ~#./un_script.sh
ven. 17 mai 2024 10:15:46 CEST
 10:15:46 up 29 min,  1 user,  load average: 0,00, 0,00, 0,00
Linux DS1 6.1.0-12-amd64 #1 SMP PREEMPT_DYNAMIC Debian 6.1.52-1 (2023-09-07) x86_64 GNU/Linux
root@DS1: ~#un_script.sh
-bash: un_script.sh : commande introuvable
root@DS1: ~#
```

▪ Affichez le **contenu** de la variable **PATH** :

```
root@DS1: ~#echo $PATH
/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin
root@DS1: ~#_
```

▪ **Modifiez** le contenu de la variable **PATH** à partir du fichier **/etc/bash.bashrc** pour pouvoir **exécuter** le **script** à partir d'un **répertoire quelconque** :

```
# enable bash completion in interactive shells
#if ! shopt -oq posix; then
#  if [ -f /usr/share/bash-completion/bash_completion ]; then
#    . /usr/share/bash-completion/bash_completion
#  elif [ -f /etc/bash_completion ]; then
#    . /etc/bash_completion
#  fi
#fi

# if the command-not-found package is installed, use it
if [ -x /usr/lib/command-not-found -o -x /usr/share/command-not-found/command-not-found ]; then
        function command_not_found_handle {
                # check because c-n-f could've been removed in the meantime
                if [ -x /usr/lib/command-not-found ]; then
                   /usr/lib/command-not-found -- "$1"
                   return $?
                elif [ -x /usr/share/command-not-found/command-not-found ]; then
                   /usr/share/command-not-found/command-not-found -- "$1"
                   return $?
                else
                   printf "%s: command not found\n" "$1" >&2
                   return 127
                fi
        }
fi
export PATH=$PATH:/root

^G Aide      ^O Écrire    ^W Chercher  ^K Couper     ^T Exécuter   ^C Emplacement M-U Annuler
^X Quitter   ^R Lire fich. ^\ Remplacer ^U Coller     ^J Justifier  ^- Aller ligne M-E Refaire

Debian GNU/Linux 12 DS1 tty1

DS1 login: root
Password:
Linux DS1 6.1.0-12-amd64 #1 SMP PREEMPT_DYNAMIC Debian 6.1.52-1 (2023-09-07) x86_64

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Fri May 17 09:46:52 CEST 2024 on tty1
root@DS1: ~#echo $PATH
/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/root
root@DS1: ~#_
```

▪ Vous pouvez maintenant **exécuter** le **script sans indiquer** son **chemin** :

```
root@DS1: ~#un_script.sh
ven. 17 mai 2024 10:23:02 CEST
 10:23:02 up 2 min,  1 user,  load average: 0,00, 0,00, 0,00
Linux DS1 6.1.0-12-amd64 #1 SMP PREEMPT_DYNAMIC Debian 6.1.52-1 (2023-09-07) x86_64 GNU/Linux
root@DS1: ~#
```

2. Les variables

▪ Créez une variable et utilisez-la :

```
root@DS1: ~#CONFIG_SHELL=/etc/profile
root@DS1: ~#ls -l $CONFIG_SHELL
-rw-r--r-- 1 root root 769 10 avril  2021 /etc/profile
root@DS1: ~#_
```

▪ **Créez** une **variable**, **affichez**-la, **affichez** toutes les **variables** (les **10 premières)** et **détruisez** une **variable** :

```
root@DS1: ~#A="bonjour Mr"
root@DS1: ~#echo $A
bonjour Mr
root@DS1: ~#set | head
A='bonjour Mr'
BASH=/bin/bash
BASHOPTS=checkwinsize:cmdhist:complete_fullquote:expand_aliases:extglob:extquote:force_fignore:globa
sciiranges:globskipdots:interactive_comments:login_shell:patsub_replacement:progcomp:promptvars:sour
cepath
BASH_ALIASES=()
BASH_ARGC=([0]="0")
BASH_ARGV=()
BASH_CMDS=()
BASH_COMPLETION_VERSINFO=([0]="2" [1]="11")
BASH_LINENO=()
BASH_LOADABLES_PATH=/usr/local/lib/bash:/usr/lib/bash:/opt/local/lib/bash:/usr/pkg/lib/bash:/opt/pkg
/lib/bash:.
root@DS1: ~#unset A
root@DS1: ~#set | grep A=
root@DS1: ~#
```

▪ Créez un **script interactif** qui saisit une variable et l'affiche :

```
root@DS1: ~#vim bonjour.sh
```

```
#!/bin/bash
printf "Votre nom ?"
read nom
echo "Bonjour Mr $nom"_
```

```
root@DS1: ~#sh bonjour.sh
Votre nom ?Lopez
Bonjour Mr Lopez
root@DS1: ~#_
```

▪ Créez un **script** qui **affiche** les **paramètres** :

```
root@DS1: ~#vim param.sh
```

```
#!/bin/bash
echo "Le nom du script          : $0"
echo "Le 1er parametre          : $1"
echo "Le 2eme parametre         : $2"
echo "Tous les parametres       : $*"
echo "Le nombre de parametres   : $#"_
```

```
root@DS1: ~#chmod +x param.sh
root@DS1: ~#./param.sh un deux trois
Le nom du script          : ./param.sh
Le 1er parametre          : un
Le 2eme parametre         : deux
Tous les parametres       : un deux trois
Le nombre de parametres : 3
root@DS1: ~#
```

3. Les structures de contrôle

3.1. Structure conditionnelle

▪ Écrivez un **script utilisant** une **alternative** :

```
root@DS1: ~#vim creer_rep.sh
```

```
#!/bin/bash
echo "Nom du répertoire a creer ? "
read nom_rep
if mkdir $nom_rep 2> /dev/null
then
        echo "Operation  reussie"
else
        echo "Echec"
        fi
```

```
root@DS1: ~#sh creer_rep.sh
Nom du répertoire a creer ?
sauve
Operation  reussie
```

```
root@DS1: ~#sh creer_rep.sh
Nom du répertoire a creer ?
sauve
Echec
root@DS1: ~#ls
avecdoublons      cal.txt       erreurs.log    param.sh          sansdoublons.txt  un_script.sh
avecdoublons.txt  creer_rep.sh  etudiants.txt  prenoms_tries     sauve
bonjour.sh        eleves.txt    notes.csv      prenoms_tries.txt  sui.txt
root@DS1: ~#_
```

▪ Mettez en œuvre une **alternative** avec utilisation de la **commande test** :

```
root@DS1: ~#vim heureux.sh
```

```
printf "Etes-vous heureux ? "
read reponse
if [ "$reponse" = "oui" ]
then
        echo "Bravo"
else
        echo "Mangez du chocolat"
fi_
```

```
root@DS1: ~#sh heureux.sh
Etes-vous heureux ? oui
Bravo
root@DS1: ~#sh heureux.sh
Etes-vous heureux ? non
Mangez du chocolat
root@DS1: ~#_
```

3.2. Boucle while.

▪ Créez un **script** qui **teste l'existence du fichier** « **flag** » toutes les **dix secondes**. Lancez le script en tâche de fond et au bout d'un moment, créez le fichier « flag ». Le script signale sa présence et se termine.

```
root@DS1: ~#vim flag_existe.sh_
```

```
while [ ! -f flag ]
do
sleep 10
done
echo "Le fichier flag existe"
~
```

```
root@DS1: ~#sh flag_existe.sh &
[1] 851
root@DS1: ~#Le fichier flag existe

[1]+  Fini                    sh flag_existe.sh
root@DS1: ~#
```

▪ Autre version du programme :

```
root@DS1: ~#vim flag_existebis.sh
```

```
while :
do
        sleep 10
        if test -f flag ;then
                break
        fi
done
echo "Le fichier flag existe"_
~
```

```
root@DS1: ~#rm flag
root@DS1: ~#sh flag_existebis.sh &
[1] 893
root@DS1: ~#touch flag
root@DS1: ~#Le fichier flag existe

[1]+  Fini                    sh flag_existebis.sh
root@DS1: ~#
```

3.3. Boucle for

▪ Ecrivez un **script** permettant **d'effectuer** un **décompte** de **secondes** :

```
root@DS1: ~#vim boucle_for.sh
```

```
for i in 5 4 3 2 1
do
        echo "=====> $i"
        sleep 1
done
echo "FEU!"
```

```
root@DS1: ~#sh boucle_for.sh
=====> 5
=====> 4
=====> 3
=====> 2
=====> 1
FEU!
root@DS1: ~#
```

3.4. Choix multiple

▪ Ecrivez un **script** permettant **d'écrire** un **menu** avec l'instruction « **case** » :

```
root@DS1: ~#vim menu.sh
```

```
#!/bin/bash

echo "1 - Afficher la date et l'heure"
echo "2 - Afficher la charge syseme"
echo "3 - Afficher la version du systeme"

echo -n "votre choix ? "
read choix

case "$choix" in
1)
        date
        ;;
2)
        update
        ;;
3)
        uname -a
        ;;
*)
        echo "choix incorrect"
        ;;
esac
```

```
root@DS1: ~#sh menu.sh
1 - Afficher la date et l'heure
2 - Afficher la charge syseme
3 - Afficher la version du systeme
votre choix ? 3
Linux DS1 6.1.0-12-amd64 #1 SMP PREEMPT_DYNAMIC Debian 6.1.52-1 (2023-09-07) x86_64 GNU/Linux
root@DS1: ~#_
```

3.5. La commande let

▪ Ecrivez un script utilisant une boucle **while** et mettant en œuvre **l'incrémentation d'une variable** via la **commande let** :

```
root@DS1: ~#vim plusun.sh
```

```
#!/bin/bash
i=0     # ou let i=0
while let 'i<5'
do
        echo Bonjour
        let i=i+1
done
```

```
root@DS1: ~#bash plusun.sh
Bonjour
Bonjour
Bonjour
Bonjour
Bonjour
root@DS1: ~#
```

Ou

```
root@DS1: ~#chmod +x plusun.sh
root@DS1: ~#plusun.sh
Bonjour
Bonjour
Bonjour
Bonjour
Bonjour
root@DS1: ~#_
```

3.6. Lecture d'un fichier et commande set

▪ Créez un fichier texte **users.txt** contenant quelques lignes comportant les champs suivants : **login, mot-de-passe, nom de l'utilisateur et nom des groupes secondaires**. Les champs sont séparés par un espace et chaque champ ne comporte pas d'espace

```
  GNU nano 7.2                                        users.txt *
kbeatini watson kevin 2SIO,SISR
acanonne navarone axel 2SIO,SISR
cmartinez malaucoccyx cedric 2SIO,SISR
vgosse toutafait victor 2SIO,SISR
jpichon pelican jonathan 2SIO,SISR
rfumey weed raphael 2SIO,SISR
vmanceau dakota vincent 2SIO,SISR
```

▪ Ecrivez le script **lecture.sh** qui **parcourt** le **fichier users.txt** ligne par ligne, **récupère** les **champs** de **chaque ligne dans les variables paramètres de position et les affiche.**

```
root@DS1: ~#vim lecture.sh
```

```
#!/bin/bash
cat users.txt | while true
do
        read ligne
        if [ "$ligne" = "" ] ; then break ; fi
        echo "lecture de la ligne --->" $ligne
        set $ligne
        login=$1
        passwd=$2
        nom=$3
        groupe=$4
        echo login=$login, mdp=$passwd, groupe=$groupe, nom=$nom
done
```

```
root@DS1: ~#sh lecture.sh
lecture de la ligne ---> kbeatini watson kevin 2SIO,SISR
login=kbeatini, mdp=watson, groupe=2SIO,SISR, nom=kevin
lecture de la ligne ---> acanonne navarone axel 2SIO,SISR
login=acanonne, mdp=navarone, groupe=2SIO,SISR, nom=axel
lecture de la ligne ---> cmartinez malaucoccyx cedric 2SIO,SISR
login=cmartinez, mdp=malaucoccyx, groupe=2SIO,SISR, nom=cedric
lecture de la ligne ---> vgosse toutafait victor 2SIO,SISR
login=vgosse, mdp=toutafait, groupe=2SIO,SISR, nom=victor
lecture de la ligne ---> jpichon pelican jonathan 2SIO,SISR
login=jpichon, mdp=pelican, groupe=2SIO,SISR, nom=jonathan
lecture de la ligne ---> rfumey weed raphael 2SIO,SISR
login=rfumey, mdp=weed, groupe=2SIO,SISR, nom=raphael
lecture de la ligne ---> vmanceau dakota vincent 2SIO,SISR
login=vmanceau, mdp=dakota, groupe=2SIO,SISR, nom=vincent
root@DS1: ~#
```

3.7. Commande shift et boucle for

▪ Ecrivez un **script** qui **reçoit** en **premier argument le nom d'un répertoire** (/test) puis, à partir du **deuxième argument,** un **nombre quelconque** de **noms** de **fichiers** (f1, f2…). Le premier argument ne subira pas le même traitement que les suivants : le nom du répertoire est sauvegardé dans une variable puis la commande **« shift »** fait sortir le nom du répertoire de la liste des arguments. Ceci permet de récupérer dans la variable **$*** uniquement la liste des fichiers qui fera l'objet ultérieurement d'une boucle for.

```
root@DS1: ~#vim decal.sh
```

```
#!/bin/bash
# Afichage des variables paramètres de position avant décalage avec shift
echo -e "Affichage avant shift\n"
echo "1er argument \$1 : $1"
echo "2eme argument \$2 : $2"
echo "3eme argument \$3 : $3"
echo "4eme argument \$4 : $4"
echo "Tous les arguments \$* : $*"
echo -e "Nombre d'argument \$# : $#\n"
# Sauvegarde du 1er argument et décalage des arguments avec la commande shift
rep=$1
shift
# Affichage des variables apres execution de la commande shift
echo -e "Afichage après utilisation de la commande shift\n"
echo "1er argument \$1 : $1"
echo "2eme argument \$2 : $2"
echo "3eme argument \$3 : $3"
echo "4eme argument \$4 : $4"
echo "Tous les arguments \$* : $*"
echo -e "Nombre d'arguments \$# : $#\n"
#Creation du répertoire et déplacement dans le répertoire créé
mkdir $rep
cd $rep_
~
#Céation des fichiers dans le répertoire créé
for fichier in $*
do
        touch $fichier
        echo "Fichier $fichier créé"
done
```

▪ Positionnez le **droit d'exécution** :

```
root@DS1: ~#chmod u+x decal.sh
root@DS1: ~#
```

▪ Exécutez le script :

```
root@DS1: ~#./decal.sh /test f1 f2 f3 f4 f5 f6
Affichage avant shift

1er argument $1 : /test
2eme argument $2 : f1
3eme argument $3 : f2
4eme argument $4 : f3
Tous les arguments $* : /test f1 f2 f3 f4 f5 f6
Nombre d'argument $# : 7

Afichage après utilisation de la commande shift

1er argument $1 : f1
2eme argument $2 : f2
3eme argument $3 : f3
4eme argument $4 : f4
Tous les arguments $* : f1 f2 f3 f4 f5 f6
Nombre d'arguments $# : 6

Fichier f1 créé
Fichier f2 créé
Fichier f3 créé
Fichier f4 créé
Fichier f5 créé
Fichier f6 créé
root@DS1: ~#
```

▪ Vérifiez la **création** du **répertoire** et des **fichiers** :

```
root@DS1: ~#ls -l /test
total 0
-rw-r--r-- 1 root root 0 17 mai   13:12 f1
-rw-r--r-- 1 root root 0 17 mai   13:12 f2
-rw-r--r-- 1 root root 0 17 mai   13:12 f3
-rw-r--r-- 1 root root 0 17 mai   13:12 f4
-rw-r--r-- 1 root root 0 17 mai   13:12 f5
-rw-r--r-- 1 root root 0 17 mai   13:12 f6
root@DS1: ~#_
```

4. Les sous-programmes

▪ Ecrivez un **script** utilisant une **fonction** de **présentation** avec **passage d'arguments** à la suite du **nom de la fonction** (cf. variables paramètres de position) :

```
root@DS1: ~#vim affiche.sh_
```

```
presentation ()
{
        for i
        do
                echo "==== $i"
        done
        echo
}

#Debut du programme

presentation "Bonjour" "ce matin" "nous etudions" "les structures de controle" "ainsi que" "les fonc
tions"
date
presentation "Soyez attentif à la syntaxe" "Bon courage"_
~
~
~
```

```
root@DS1: ~#sh affiche.sh
==== Bonjour
==== ce matin
==== nous etudions
==== les structures de controle
==== ainsi que
==== les fonctions

ven. 17 mai 2024 13:20:25 CEST
==== Soyez attentif à la syntaxe
==== Bon courage

root@DS1: ~#
```

▪ Ecrivez un **script** qui **affiche un menu**. La sortie du **programme** est **contrôlée** par une **fonction** qui **demande une confirmation**.

```
root@DS1: ~#vim menu2.sh
```

```
#!/bin/bash

confirmation ()
{
        printif "Etes-vous sur $* (y/n) ?"
        read reponse
        if [ $reponse = "y" ] ; then return 0 ; else return 1 ; fi_
}
~
```

```
while :
do
        clear
        echo "1 - afficher la date et l'heure"
        echo "2 - afficher la charge systeme"
        echo "3 - afficher la version du systeme"
        echo "99 - FIN "
        printf "Votre choix ? "; read choix
        case "$choix" in
        1) date ;;
        2) uptime ;;
        3) uname -a ;;
        99)
                if confirmation "de vouloir quitter" ; then
                        break
                fi
                ;;
        *) echo "choix incorrect" ;;
        esac
        sleep 3
done
exit 0
```

```
1 - afficher la date et l'heure
2 - afficher la charge systeme
3 - afficher la version du systeme
99 - FIN
Votre choix ? 99
Etes-vous sur de vouloir quitter (y/n) ?y
root@DS1: ~#_
```

```
1 - afficher la date et l'heure
2 - afficher la charge systeme
3 - afficher la version du systeme
99 - FIN
Votre choix ? 9
choix incorrect
```