

Module : Système et Scripting

Année Universitaire
2024-2025

Planification & Informations

- **Chapitre 1** : Introduction & Commandes de base
- **Chapitre 2** : Langage de programmation Shell
- **Chapitre 3** : Conditions & Boucles en Shell
- **Chapitre 4** : Sous-programmes en Shell

Système & Scripting

Chapitre III : Conditions & Boucles en Shell

Plan

1. Structure conditionnelle test
2. Structure conditionnelle if
3. Structure conditionnelle case
4. Structure itérative for
5. Structure itérative while, until
6. Exercices

Les structures conditionnelles test

La fonction **test** permet d'exprimer une condition ou plusieurs:

- Sur des fichiers
- Sur des nombres
- Sur des chaînes de caractères

Les structures conditionnelles test sur les fichiers

`test -e <nom_fichier>` : vrai si l'argument existe

`test -f <nom_fichier>` : vrai si l'argument est un fichier ordinaire

`test -d <nom_fichier>` : vrai si l'argument est un répertoire

`test -L <nom_fichier>` : vrai si l'argument est un lien symbolique

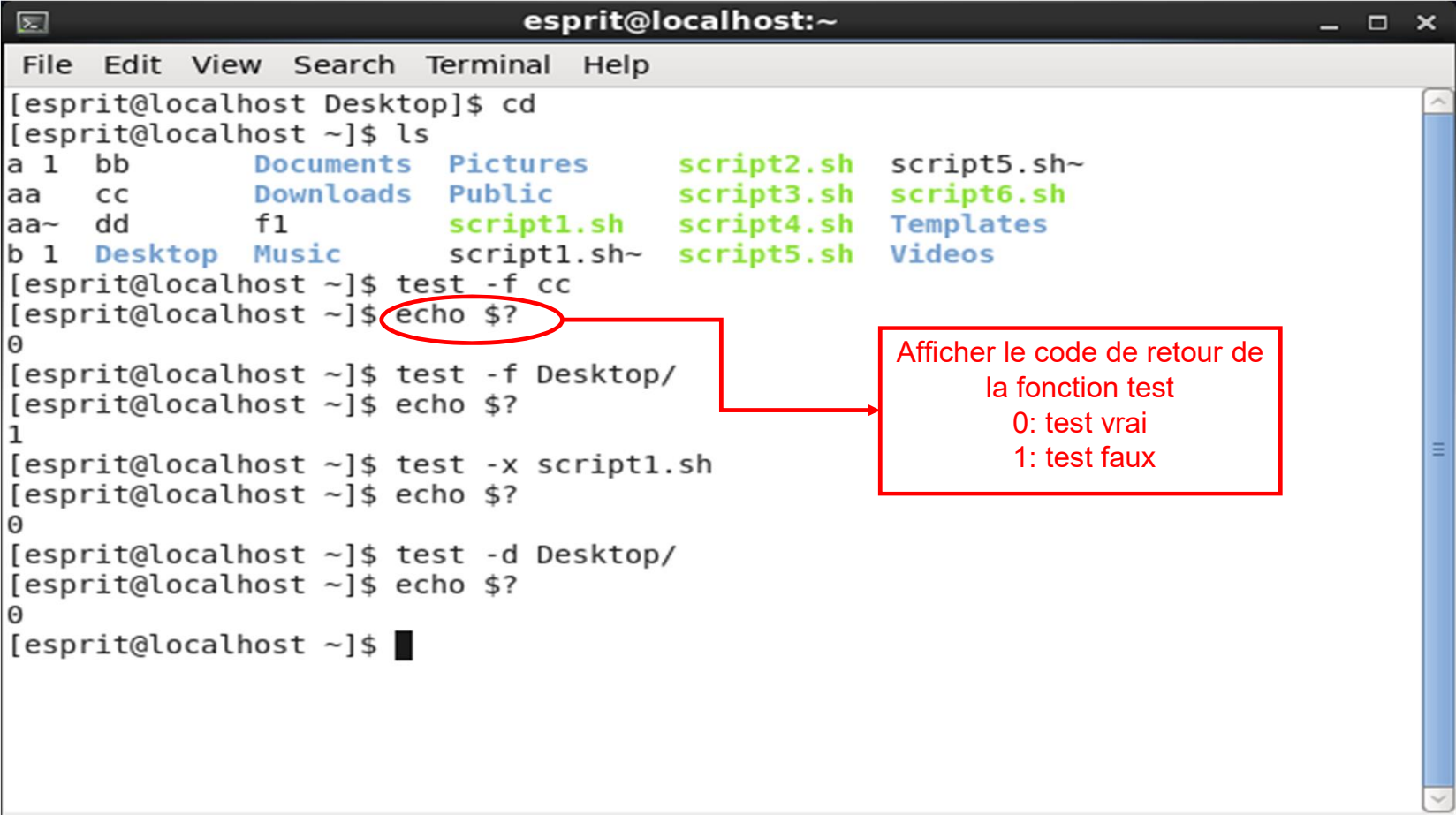
`test -r <nom_fichier>` : vrai si on a le droit de lire le fichier

`test -x <nom_fichier>` : vrai si on a le droit d'exécuter le fichier

`test -w <nom_fichier>` : vrai si on a le droit d'écrire dans le fichier

`test -s <nom_fichier>` : vrai si l'argument existe et non vide

Les structures conditionnelles test sur les fichiers



```
esprit@localhost:~  
File Edit View Search Terminal Help  
[esprit@localhost Desktop]$ cd  
[esprit@localhost ~]$ ls  
a 1  bb      Documents Pictures  script2.sh  script5.sh~  
aa cc      Downloads Public    script3.sh  script6.sh  
aa~ dd      f1       script1.sh  script4.sh  Templates  
b 1  Desktop Music  script1.sh~ script5.sh  Videos  
[esprit@localhost ~]$ test -f cc  
[esprit@localhost ~]$ echo $?  
0  
[esprit@localhost ~]$ test -f Desktop/  
[esprit@localhost ~]$ echo $?  
1  
[esprit@localhost ~]$ test -x script1.sh  
[esprit@localhost ~]$ echo $?  
0  
[esprit@localhost ~]$ test -d Desktop/  
[esprit@localhost ~]$ echo $?  
0  
[esprit@localhost ~]$
```

Afficher le code de retour de la fonction test
0: test vrai
1: test faux

Les structures conditionnelles test sur les nombres

Pour les expressions arithmétiques, les opérateurs sont:

- eq , -ne, -lt, -le, -gt , -ge

équivalent à égale, différent, inférieur strictement, inférieur ou égale, supérieur strictement, supérieur ou égale

test "i1" -eq "i2": Teste si i1 est égal à i2

test "i1" -ne "i2": Teste si i1 n'est pas égal à i2

test "i1" -gt "i2": Teste si i1 est plus grand que i2

test "i1" -lt "i2": Teste si i1 est inférieur à i2

test "i1" -ge "i2": Teste si i1 est plus grand ou égal à i2

test "i1" -le "i2": Teste si i1 est inférieur ou égal à i2

Les structures conditionnelles test sur les nombres



```
esprit@localhost:~  
File Edit View Search Terminal Help  
[esprit@localhost Desktop]$ cd  
[esprit@localhost ~]$ v1=50  
[esprit@localhost ~]$ v2=30  
[esprit@localhost ~]$ test $v1 -eq $v2  
[esprit@localhost ~]$ echo $?  
1  
[esprit@localhost ~]$ test $v1 -gt $v2  
[esprit@localhost ~]$ echo $?  
0  
[esprit@localhost ~]$ test $v1 -lt $v2  
[esprit@localhost ~]$ echo $?  
1  
[esprit@localhost ~]$
```

Les structures conditionnelles test sur les chaînes de caractères

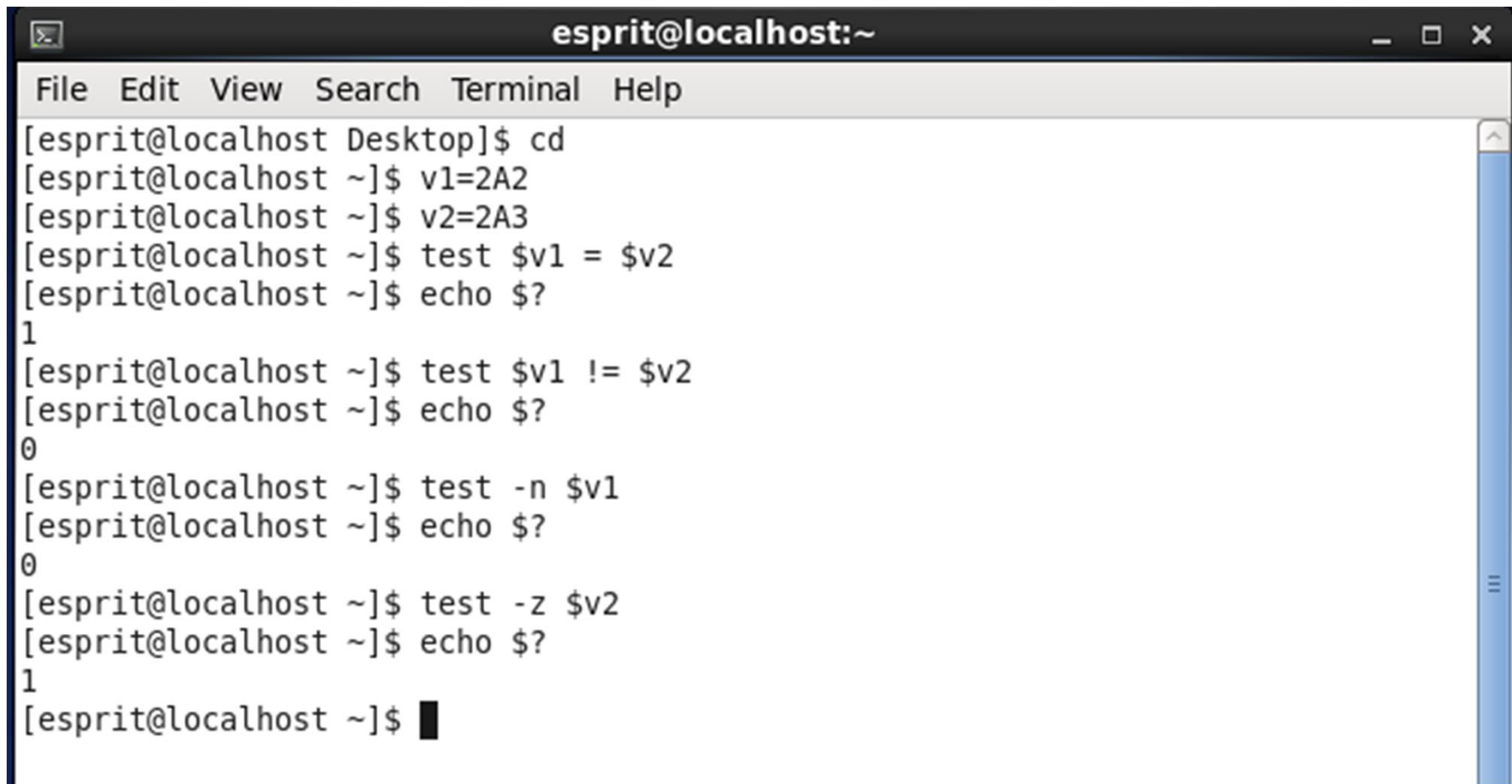
`test -n "string"` : Teste si la longueur de la chaîne string est différente de zéro

`test -z "string"`: Teste si la chaîne string est égale à zéro

`test "s1" = "s2"`: Teste si la chaîne s1 est égale à s2

`test "s1" != "s2"`: Teste si la chaîne s1 n'est pas égale à s2

Les structures conditionnelles test sur les chaînes de caractères



```
esprit@localhost:~  
File Edit View Search Terminal Help  
[esprit@localhost Desktop]$ cd  
[esprit@localhost ~]$ v1=2A2  
[esprit@localhost ~]$ v2=2A3  
[esprit@localhost ~]$ test $v1 = $v2  
[esprit@localhost ~]$ echo $?  
1  
[esprit@localhost ~]$ test $v1 != $v2  
[esprit@localhost ~]$ echo $?  
0  
[esprit@localhost ~]$ test -n $v1  
[esprit@localhost ~]$ echo $?  
0  
[esprit@localhost ~]$ test -z $v2  
[esprit@localhost ~]$ echo $?  
1  
[esprit@localhost ~]$
```

Les structures conditionnelles test vs []

test -e fichier
test \$v1 -eq \$v2
test \$chaine1 = \$chaine2



[-e fichier **]**
[\$v1 -eq \$v2 **]**
[\$chaine1 = \$chaine2 **]**

Combinaison de conditions

! : Condition Not **[! -d /etc/group]**

-a : Condition AND **[-f script.sh -a -x script.sh]**

-o : Condition OR **[-d script.sh -o -x script.sh]**

\(\) : pour traiter un groupe d'expressions

[-w script.sh -a \(-e fich -o -e fich7 \)]

Effectuer plusieurs tests à la fois

OR = || `test -f file || echo « le fichier file n'existe pas »`

`[-f file] || echo « le fichier file n'existe pas »`

AND = && `test -f file && echo « le fichier file existe »`

Les structures conditionnelles if

Syntaxe 1

```
if condition
  then
    Traitement 1
  else
    Traitement 2
fi
```

Syntaxe 2

```
if condition 1
  then
    Traitement 1
  else if condition 2
    then
      Traitement 2
  else if condition 3
    then
      traitement 3
    fi
  fi
fi
```

Syntaxe 3

```
if condition 1
  then
    Traitement 1
  if condition 2
    then
      Traitement 2
  else Traitement 3
fi
```

...

Syntaxe 4

```
if condition ; then
  Traitement
fi
```

Les structures conditionnelles if



The image shows a Gedit editor window titled `*script1.sh (~) - gedit` and a terminal window titled `esprit@localhost:~`.

The Gedit window contains the following script:

```
#!/bin/bash
ls
echo "Choisir un nom de répertoire ou fichier"
read NAME
if [ -f $NAME ]
then echo $NAME ":fichier ordinaire"
elif [ -d $NAME ]
then echo $NAME ":répertoire"
else echo $NAME ":type non traité"
fi
```

The terminal window shows the execution of the script:

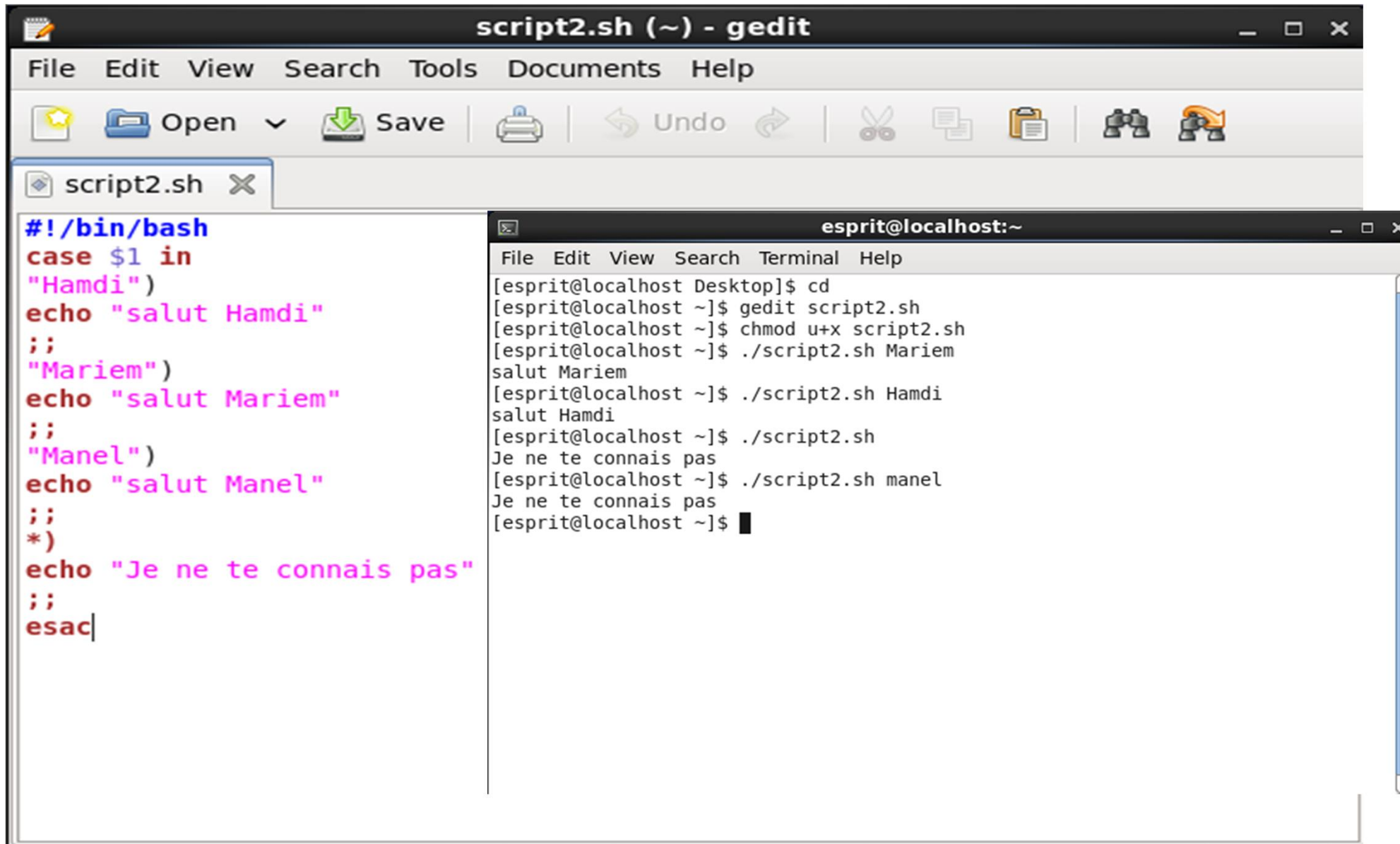
```
esprit@localhost Desktop]$ cd
esprit@localhost ~]$ gedit script1.sh
esprit@localhost ~]$ ./script1.sh
a 1 aa~ bb dd Documents fl Pictures script1.sh Templates
aa b 1 cc Desktop Downloads Music Public script1.sh~ Videos
Choisir un nom de répertoire ou fichier
bb
bb :fichier ordinaire
esprit@localhost ~]$ ./script1.sh
a 1 aa~ bb dd Documents fl Pictures script1.sh Templates
aa b 1 cc Desktop Downloads Music Public script1.sh~ Videos
Choisir un nom de répertoire ou fichier
Downloads
Downloads :répertoire
esprit@localhost ~]$ ./script1.sh
a 1 aa~ bb dd Documents fl Pictures script1.sh Templates
aa b 1 cc Desktop Downloads Music Public script1.sh~ Videos
Choisir un nom de répertoire ou fichier
music
music :type non traité
esprit@localhost ~]$
```

The status bar at the bottom of the Gedit window shows `sh`, `Tab Width: 8`, `Ln 3, Col 47`, and `INS`.

Les structures conditionnelles case

```
case $variable in
  expr1) commandes ;;
  expr2) commandes ;;
  .
  .
  .
  .
  *) par_défaut ;;
esac
```

Les structures conditionnelles case



The image shows a Gedit editor window titled 'script2.sh (~) - gedit' and a terminal window titled 'esprit@localhost:~'. The Gedit window contains a shell script using a 'case' statement to greet different names. The terminal window shows the script being executed with various inputs, demonstrating its behavior.

```
#!/bin/bash
case $1 in
"Hamdi")
echo "salut Hamdi"
;;
"Mariem")
echo "salut Mariem"
;;
"Manel")
echo "salut Manel"
;;
*)
echo "Je ne te connais pas"
;;
esac
```

```
File Edit View Search Terminal Help
[esprit@localhost Desktop]$ cd
[esprit@localhost ~]$ gedit script2.sh
[esprit@localhost ~]$ chmod u+x script2.sh
[esprit@localhost ~]$ ./script2.sh Mariem
salut Mariem
[esprit@localhost ~]$ ./script2.sh Hamdi
salut Hamdi
[esprit@localhost ~]$ ./script2.sh
Je ne te connais pas
[esprit@localhost ~]$ ./script2.sh manel
Je ne te connais pas
[esprit@localhost ~]$
```

Les structures itératives for

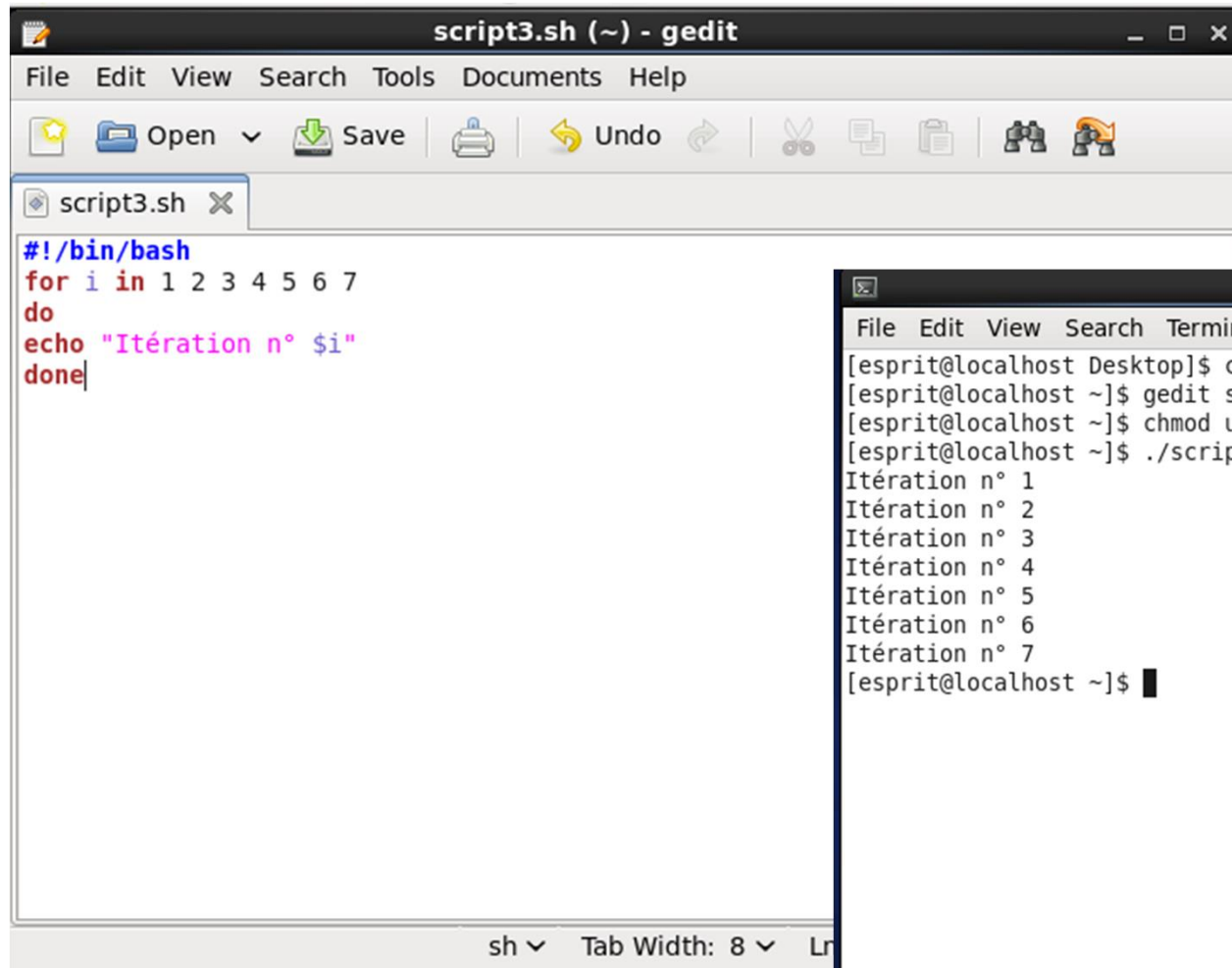
for variable [**in** liste]

do

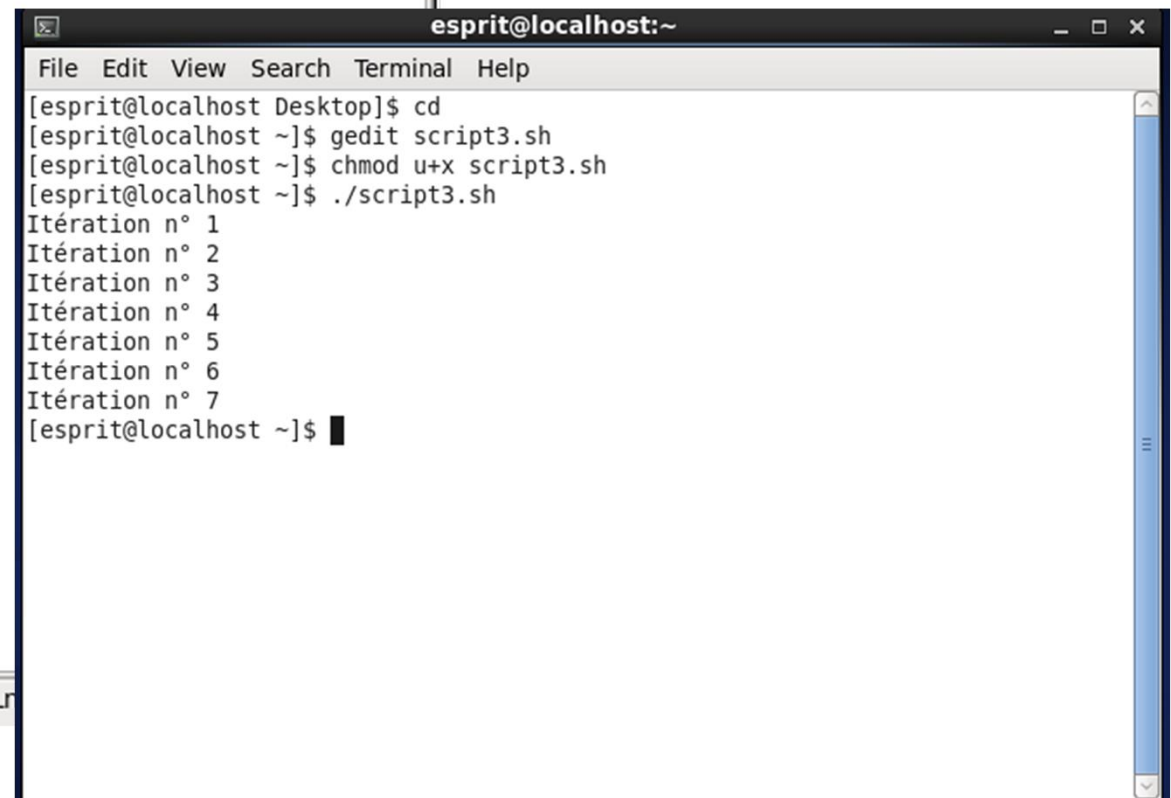
commandes (*utilisant \$variable*)

done

Les structures itératives for

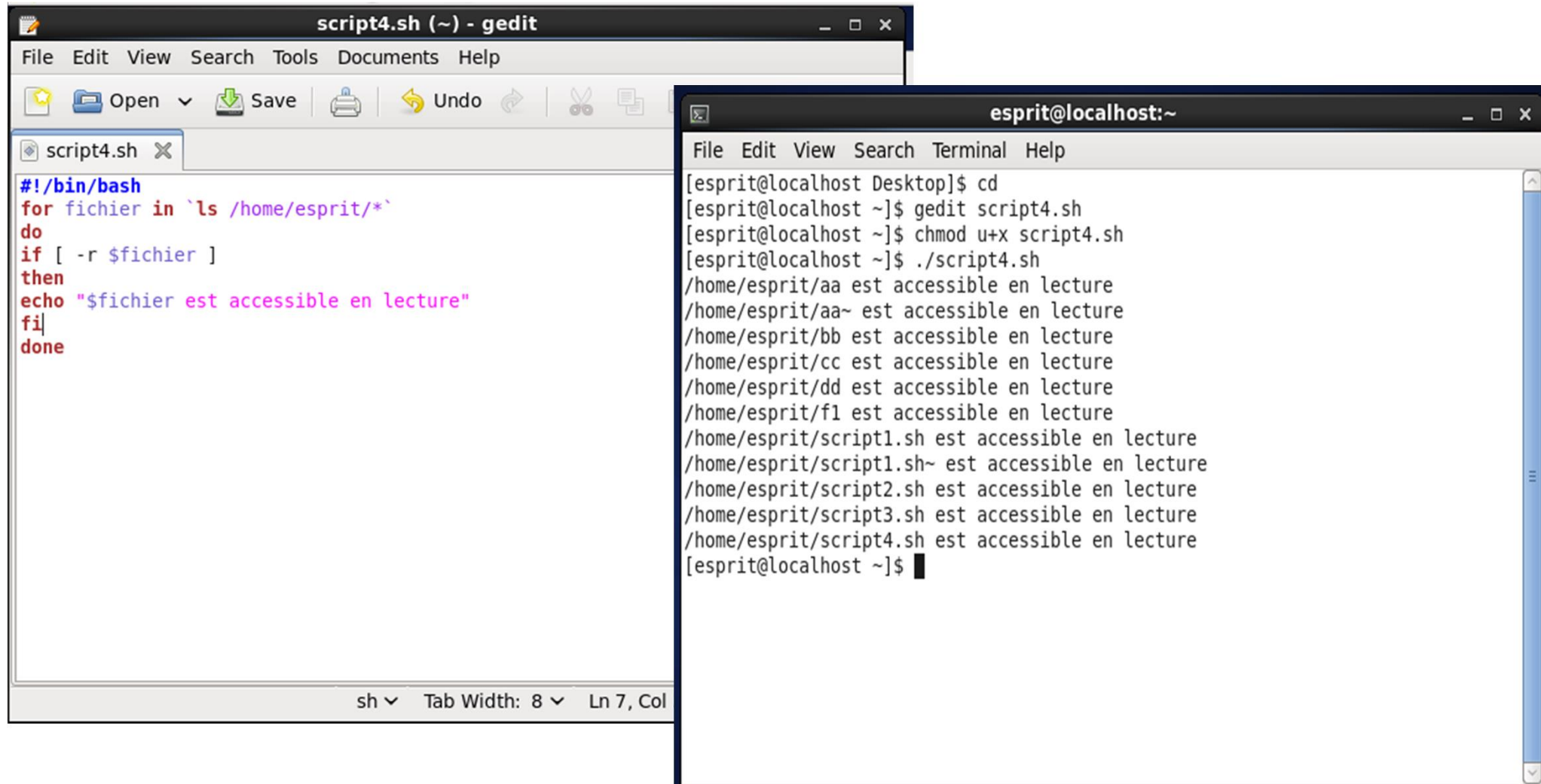


```
script3.sh (~) - gedit
File Edit View Search Tools Documents Help
★ Open Save Undo
script3.sh
#!/bin/bash
for i in 1 2 3 4 5 6 7
do
echo "Itération n° $i"
done
```



```
esprit@localhost:~
File Edit View Search Terminal Help
[esprit@localhost Desktop]$ cd
[esprit@localhost ~]$ gedit script3.sh
[esprit@localhost ~]$ chmod u+x script3.sh
[esprit@localhost ~]$ ./script3.sh
Itération n° 1
Itération n° 2
Itération n° 3
Itération n° 4
Itération n° 5
Itération n° 6
Itération n° 7
[esprit@localhost ~]$
```

Les structures itératives for



The image shows two overlapping windows. The background window is a Gedit text editor titled 'script4.sh (~) - gedit'. It contains a shell script with a 'for' loop that iterates over files in the directory '/home/esprit/'. The script checks if each file is readable and prints a message. The foreground window is a terminal titled 'esprit@localhost:~'. It shows the execution of the script, with the output of the 'for' loop displayed line by line.

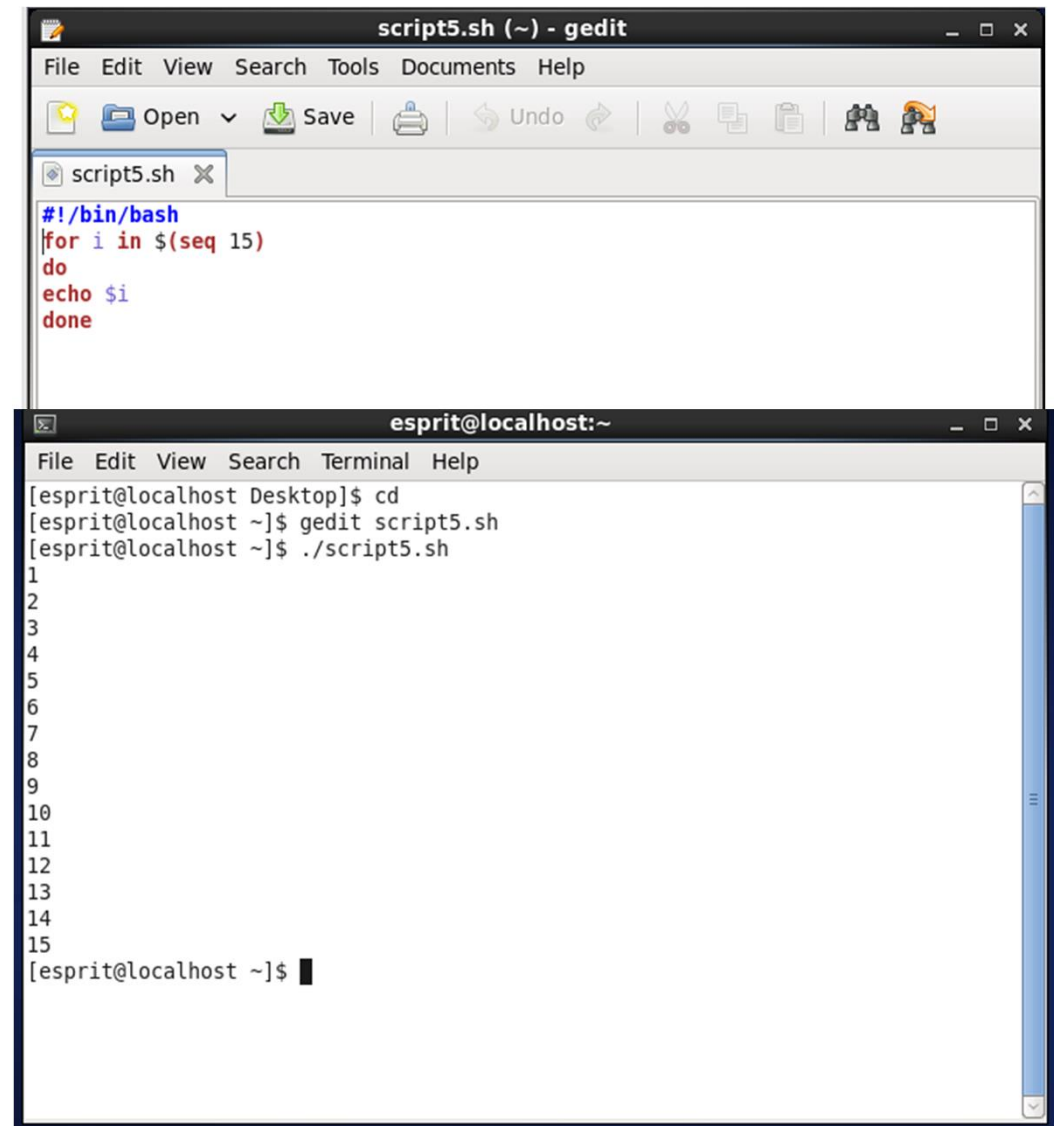
```
#!/bin/bash
for fichier in `ls /home/esprit/*`
do
if [ -r $fichier ]
then
echo "$fichier est accessible en lecture"
fi
done
```

```
[esprit@localhost Desktop]$ cd
[esprit@localhost ~]$ gedit script4.sh
[esprit@localhost ~]$ chmod u+x script4.sh
[esprit@localhost ~]$ ./script4.sh
/home/esprit/aa est accessible en lecture
/home/esprit/aa~ est accessible en lecture
/home/esprit/bb est accessible en lecture
/home/esprit/cc est accessible en lecture
/home/esprit/dd est accessible en lecture
/home/esprit/fl est accessible en lecture
/home/esprit/script1.sh est accessible en lecture
/home/esprit/script1.sh~ est accessible en lecture
/home/esprit/script2.sh est accessible en lecture
/home/esprit/script3.sh est accessible en lecture
/home/esprit/script4.sh est accessible en lecture
[esprit@localhost ~]$
```

Commande seq

```
for i in 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15
do
echo $i
done
```

Très pénible ! Pour itérer par exemple 250 fois, vous devez saisir tout cela au clavier ! Heureusement, **il y a un « raccourci »**, la **commande seq**, qui affiche une séquence de nombres allant de 1 jusqu'au maximum indiqué



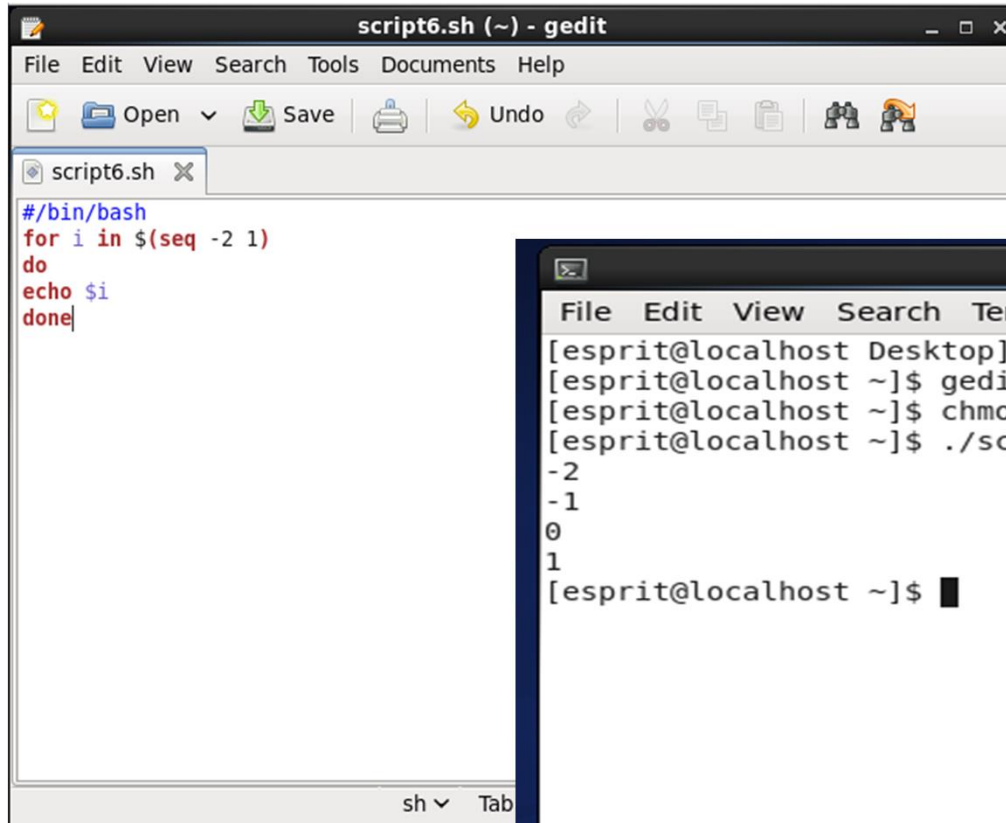
The image shows two windows from a Linux desktop environment. The top window is a Gedit text editor titled 'script5.sh (~) - gedit'. It contains a shell script with the following content:

```
#!/bin/bash
for i in $(seq 15)
do
echo $i
done
```

The bottom window is a terminal titled 'esprit@localhost:~'. It shows the execution of the script:

```
[esprit@localhost Desktop]$ cd
[esprit@localhost ~]$ gedit script5.sh
[esprit@localhost ~]$ ./script5.sh
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
[esprit@localhost ~]$
```


Commande seq



A screenshot of a gedit editor window titled "script6.sh (~) - gedit". The window has a menu bar with "File", "Edit", "View", "Search", "Tools", "Documents", and "Help". Below the menu bar is a toolbar with icons for "Open", "Save", "Print", "Undo", "Redo", "Cut", "Copy", "Paste", "Find", and "Run". The main text area contains the following script:

```
#!/bin/bash
for i in $(seq -2 1)
do
echo $i
done
```

At the bottom of the window, there is a status bar showing "sh" and "Tab".



A screenshot of a terminal window titled "esprit@localhost:~". The window has a menu bar with "File", "Edit", "View", "Search", "Terminal", and "Help". The terminal shows the following commands and output:

```
[esprit@localhost Desktop]$ cd
[esprit@localhost ~]$ gedit script6.sh
[esprit@localhost ~]$ chmod u+x script6.sh
[esprit@localhost ~]$ ./script6.sh
-2
-1
0
1
[esprit@localhost ~]$
```

Les structures itératives while, until

while liste-commandes

do

commandes

done

La répétition se poursuit TANT QUE la dernière commande de la liste est vraie
(renvoie un code de retour nul)

until liste-commandes

do

commandes

done

La répétition se poursuit JUSQU'A CE QUE la dernière commande de la liste
devienne vraie

Les structures itératives while, until

while

```
#!/bin/bash
num=0
while [ "$num" -le 10 ]; do
    echo -n "Entrez un nombre <10):"
    read num
done
echo "Vous avez entré un nombre>10"
```

```
esprit@esprit:~/Desktop$ ./while
Entrez un nombre <10):2
Entrez un nombre <10):6
Entrez un nombre <10):12
Vous avez entré un nombre>10
```

until

```
#!/bin/bash
num=0
until [ "$num" -gt 10 ]; do
    echo -n "Entrez un nombre<10):"
    read num
done
echo "Vous avez entré un nombre>10"
```

```
esprit@esprit:~/Desktop$ ./until
Entrez un nombre<10):3
Entrez un nombre<10):8
Entrez un nombre<10):13
Vous avez entré un nombre>10
```

Exercices

Exercice 1

Créer un script **Shell** nommé "**change**" qui affichera la date de dernière modification d'un fichier puis la modifiera avec l'heure actuelle et enfin réaffichera la date de dernière modification du fichier.

Cette procédure acceptera 1 paramètre qui sera le nom du fichier.

Lorsque vous exécuterez "**change mon_fichier**", le 8 octobre à 15 heures 12 vous obtiendrez le résultat:

avant : -r--r--r-- 1 user group 40 Feb 3 2013 mon_fichier

après : -r--r--r-- 1 user group 40 Oct 8 15:12 mon_fichier

Exercices

Correction exercice1

```
#!/bin/bash

# Vérifier si le fichier a été fourni en argument
if [ -z "$1" ]; then
    echo "Veuillez fournir un nom de fichier."
else if [ ! -f "$1" ]; then # Vérifier si le fichier existe
    echo "Le fichier '$1' n'existe pas."
else
    # Afficher la date de dernière modification avant la modification
    echo "avant :"
    ls -l "$1"
    # Modifier la date de dernière modification avec l'heure actuelle
    touch "$1"
    # Afficher la date de dernière modification après la modification
    echo "après :"
    ls -l "$1"
fi
fi
```

```
esprit@esprit:~/Desktop$ ./chap3_exemple1
Veuillez fournir un nom de fichier.
esprit@esprit:~/Desktop$ ./chap3_exemple1 ddd
Le fichier 'ddd' n'existe pas.
esprit@esprit:~/Desktop$ ./chap3_exemple1 2A3
Le fichier '2A3' n'existe pas.
esprit@esprit:~/Desktop$ ./chap3_exemple1 info.txt
avant :
-rw-rw-r-- 1 esprit esprit 85 23:36 10 جانفي info.txt
après :
-rw-rw-r-- 1 esprit esprit 85 19:13 28 جانفي info.txt
```

Exercices

Exercice 2

1. Créer un script permettant d'afficher la liste des fichiers du répertoire personnel accessibles en lecture.
2. Créer un script permettant d'afficher la liste des fichiers du répertoire personnel accessibles en écriture.

Correction Exercice 2

```
#!/bin/bash

echo "Fichiers accessibles en lecture dans le répertoire personnel:"
# Boucle pour parcourir les fichiers du répertoire personnel
for fichier in $HOME/*; do
    if [ -f "$fichier" ] && [ -r "$fichier" ]; then
        echo "$fichier"
    fi
done

echo "Fichiers accessibles en écriture dans le répertoire personnel:"
# Boucle pour parcourir les fichiers du répertoire /etc
for fichier in $HOME/*; do
    if [ -f "$fichier" ] && [ -w "$fichier" ]; then
        echo "$fichier"
    fi
done
```

```
esprit@esprit:~/Desktop$ ./chap3_exemple2
Fichiers accessibles en lecture dans le répertoire personnel:
/home/esprit/file
/home/esprit/lienpp
/home/esprit/script3
/home/esprit/script4
Fichiers accessibles en écriture dans le répertoire personnel:
/home/esprit/file
/home/esprit/lienpp
/home/esprit/script3
/home/esprit/script4
```

Exercices

Exercice 3

Créer un script nommé "table" permettant d'afficher des tables de multiplication.
"table 5 10" aura pour résultat l'affichage:

0 x 5 = 0
1 x 5 = 5
2 x 5 = 10
3 x 5 = 15.....

```
#!/bin/bash

# Vérifier que deux arguments sont fournis
if [ $# -ne 2 ]; then
    echo "donner le nombre et la limite de multiplication"
else
    # Boucle pour afficher la table de multiplication
    for i in $(seq 1 $2); do
        echo "$1 x $i =$(( $1 * $i ))"
    done
fi
```

```
esprit@esprit:~/Desktop$ ./chap3_exemple3
donner le nombre et la limite de multiplication
esprit@esprit:~/Desktop$ ./chap3_exemple3 5 10
5 x 1 =5
5 x 2 =10
5 x 3 =15
5 x 4 =20
5 x 5 =25
5 x 6 =30
5 x 7 =35
5 x 8 =40
5 x 9 =45
5 x 10 =50
```