



IUS
INSTITUT
UNIVERSITAIRE
DES SCIENCES

Faculté des Sciences et Technologie

(FST)

Niveau : L3-FST

Systèmes

Soumis au chargé de cours : Ismaël SAINT AMOUR

Préparé par : Jameson DOMINIQUE

Date : 06 Février 2025

Systèmes d'exploitation Linux

Introduction aux Scripts Shell, Utilisation de Variables, Boucles et conditions.

TD 8

Objectif :

Ce TD te permettra de te familiariser avec l'écriture de scripts Shell de base et l'utilisation des variables.

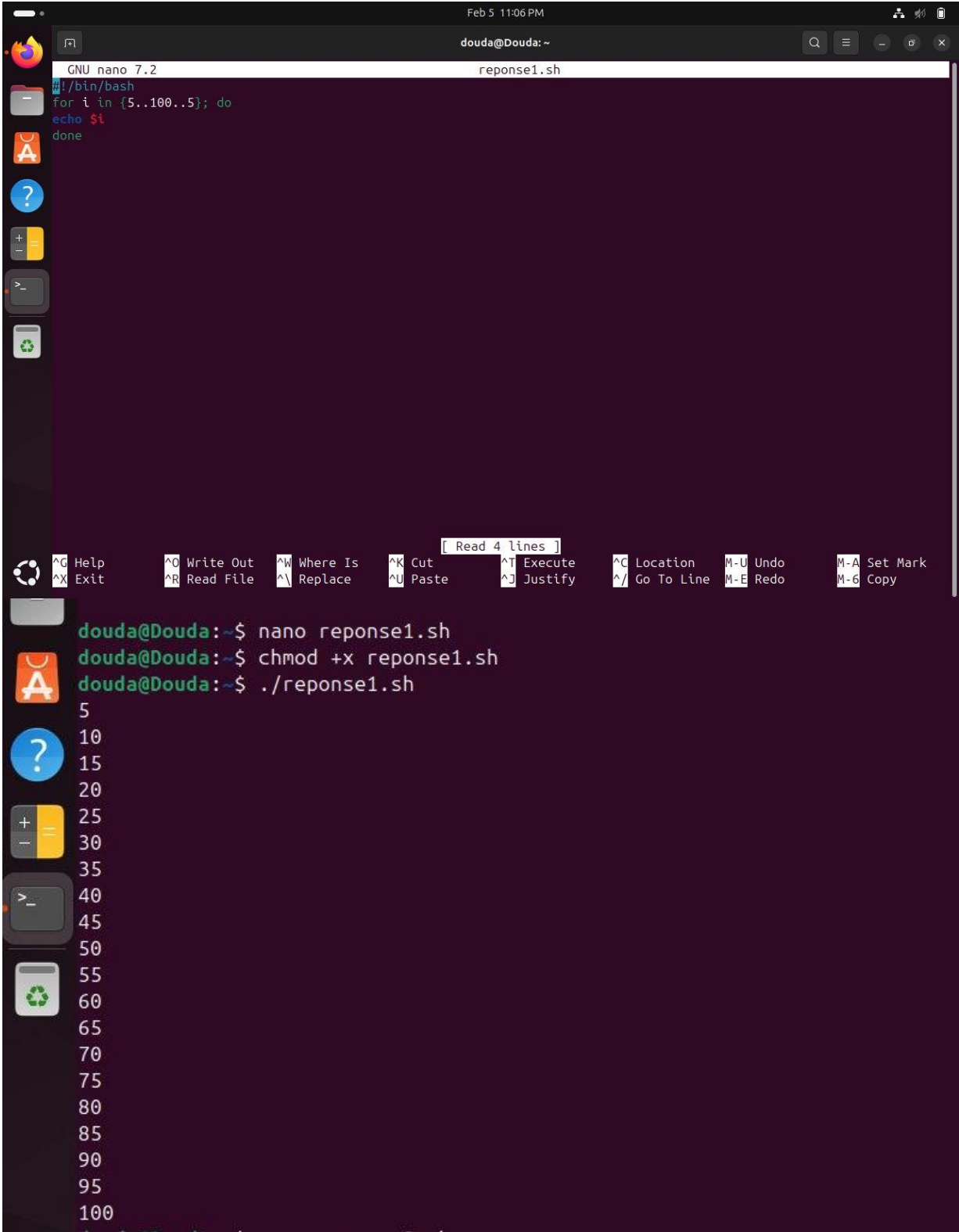
1. Comprendre la structure de base d'un script shell.
2. Manipuler des variables dans un script.
3. Utiliser des conditions (`if`, `case`).
4. Implémenter des boucles (`for`, `while`).
5. Automatiser des tâches simples avec des scripts shell.

Matériel Nécessaire :

- ♦ Un ordinateur avec Linux installé ou une machine virtuelle avec une distribution Linux (comme Ubuntu).
- ♦ Accès au terminal.
- ♦ Accès à Internet.

Travaux Dirigés

1. Créez un script qui affiche les multiples de 5 entre 5 et 100.



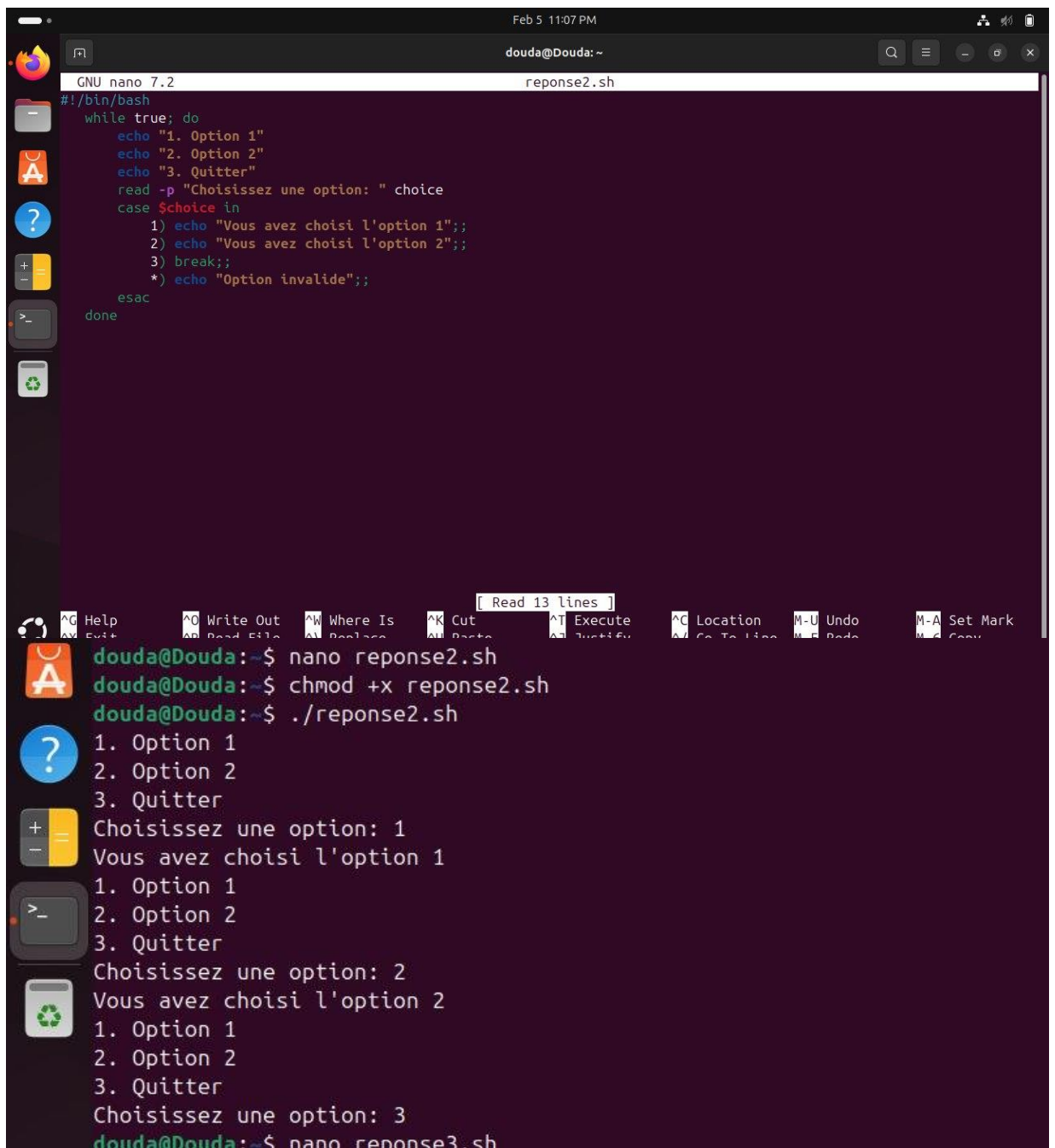
The screenshot shows a terminal window with the following content:

```
GNU nano 7.2 reponse1.sh
#!/bin/bash
for i in {5..100..5}; do
echo $i
done

douda@douda:~$ nano reponse1.sh
douda@douda:~$ chmod +x reponse1.sh
douda@douda:~$ ./reponse1.sh
5
10
15
20
25
30
35
40
45
50
55
60
65
70
75
80
85
90
95
100
```

The terminal window has a dark purple background. The nano editor's status bar at the bottom shows various keyboard shortcuts for editing and file operations. The script 'reponse1.sh' is a simple bash script that uses a 'for' loop to iterate over the range {5..100..5} and prints each value on a new line.

2. Créez un script avec une boucle qui affiche un menu interactif à l'utilisateur.



The screenshot shows a terminal window with a dark background. At the top, the title bar indicates the date and time as 'Feb 5 11:07 PM' and the user as 'douda@douda: ~'. The terminal is running GNU nano 7.2, editing a file named 'reponse2.sh'. The script content is as follows:

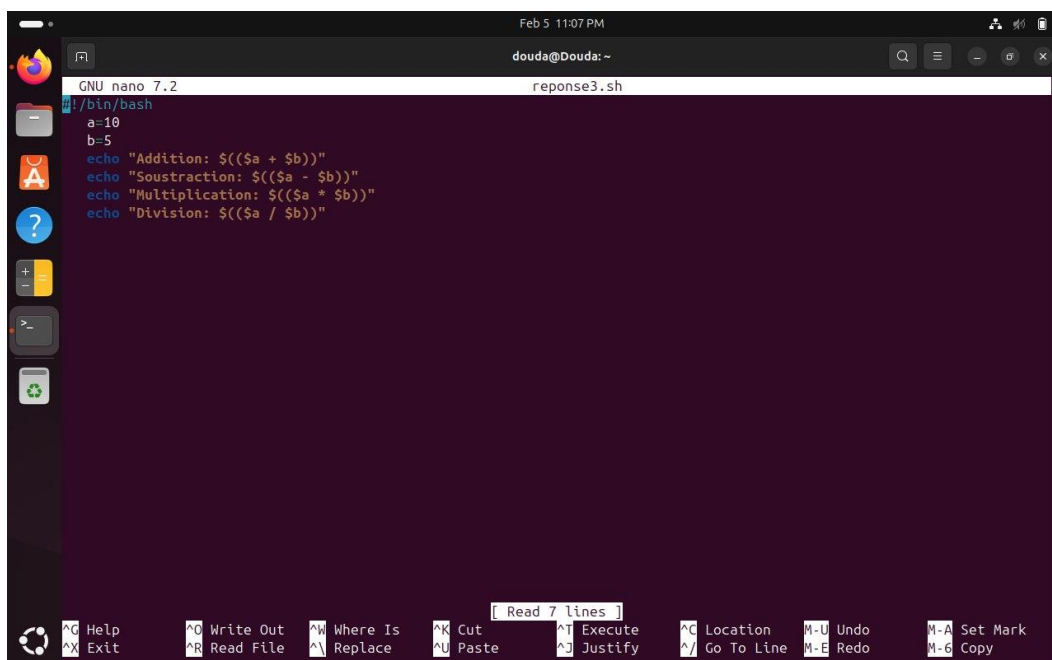
```
#!/bin/bash
while true; do
  echo "1. Option 1"
  echo "2. Option 2"
  echo "3. Quitter"
  read -p "Choisissez une option: " choice
  case $choice in
    1) echo "Vous avez choisi l'option 1";;
    2) echo "Vous avez choisi l'option 2";;
    3) break;;
    *) echo "Option invalide";;
  esac
done
```

Below the nano editor, the terminal shows the user's commands and the script's output:

```
douda@douda:~$ nano reponse2.sh
douda@douda:~$ chmod +x reponse2.sh
douda@douda:~$ ./reponse2.sh
1. Option 1
2. Option 2
3. Quitter
Choisissez une option: 1
Vous avez choisi l'option 1
1. Option 1
2. Option 2
3. Quitter
Choisissez une option: 2
Vous avez choisi l'option 2
1. Option 1
2. Option 2
3. Quitter
Choisissez une option: 3
douda@douda:~$ nano reponse3.sh
```

The nano editor's status bar at the bottom shows 'Read 13 lines' and various keyboard shortcuts like 'Help', 'Write Out', 'Where Is', 'Cut', 'Execute', 'Location', 'Undo', and 'Set Mark'.

3. Créez un script qui permet d'effectuer des opérations sur les variables.



The screenshot shows a terminal window with a dark background. At the top, the title bar indicates the date and time as 'Feb 5 11:07 PM' and the user as 'douda@douda: ~'. The terminal is running GNU nano 7.2, editing a file named 'reponse3.sh'. The script content is as follows:

```
#!/bin/bash
a=10
b=5
echo "Addition: $((a + b))"
echo "Soustraction: $((a - b))"
echo "Multiplication: $((a * b))"
echo "Division: $((a / b))"
```

The nano editor's status bar at the bottom shows 'Read 7 lines' and various keyboard shortcuts like 'Help', 'Write Out', 'Where Is', 'Cut', 'Execute', 'Location', 'Undo', and 'Set Mark'.

```
douda@douda:~$ nano reponse3.sh
douda@douda:~$ chmod +x reponse3.sh
douda@douda:~$ ./reponse3.sh
Addition: 15
Soustraction: 5
Multiplication: 50
Division: 2
douda@douda:~$
```

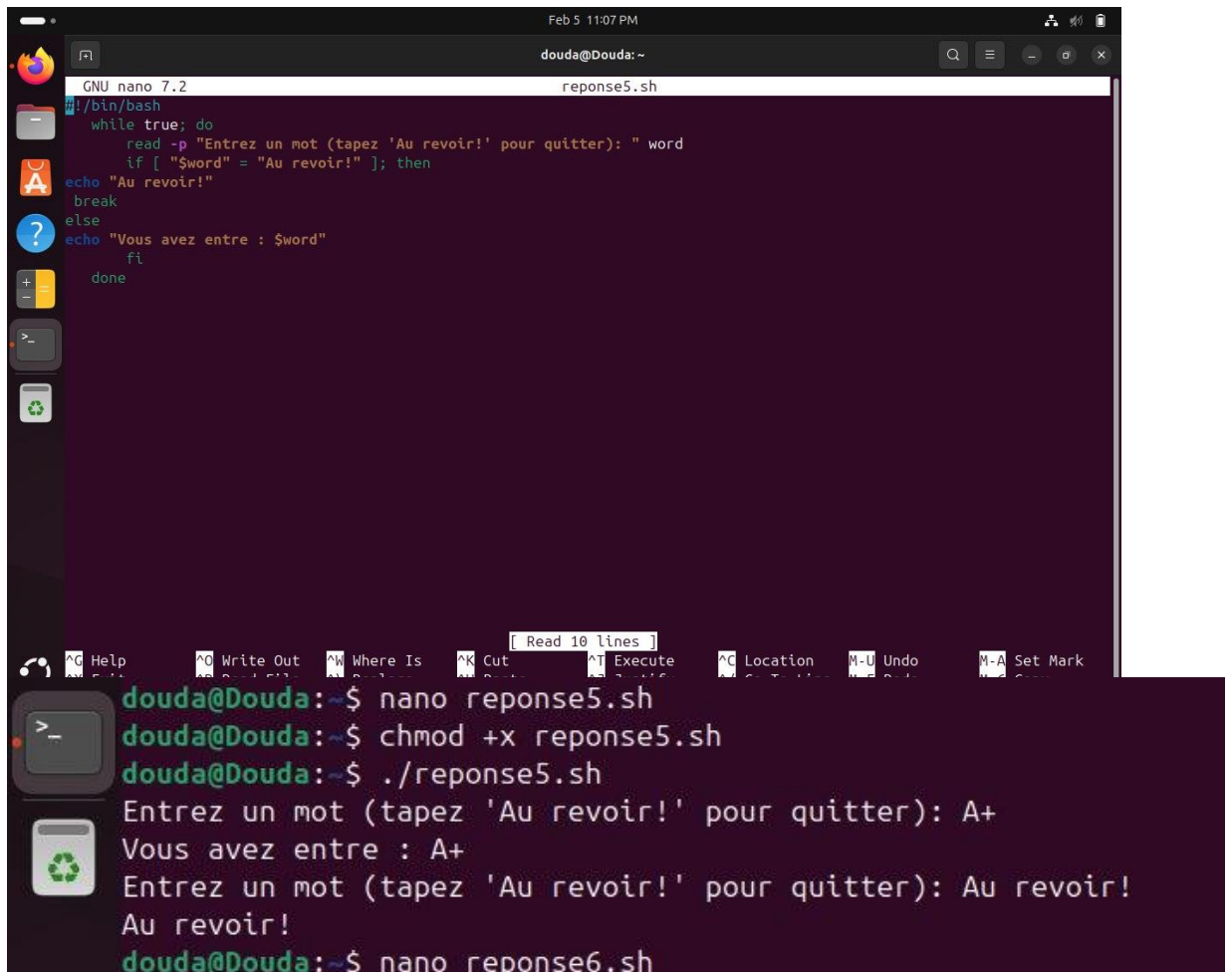
4. Créez un script qui utilise une boucle `until` pour demander à l'utilisateur de fournir un fichier existant et un mot à rechercher.

```
GNU nano 7.2 reponse4.sh
#!/bin/bash
until [ -f "$file" ]; do
    read -p "Entrez le nom d'un fichier existant: " file
    if [ ! -f "$fichier" ]; then
        echo "Le fichier n'existe pas. Veuillez reessayer."
    fi
done

read -p "Entrez un mot à rechercher: " word
if grep -q "$word" "$file"; then
    echo "Le mot '$word' a ete trouve dans le fichier."
else
    echo "Le mot '$word' n'a pas ete trouve dans le fichier."
fi
done

douda@douda:~$ nano reponse4.sh
douda@douda:~$ chmod +x reponse4.sh
douda@douda:~$ ./reponse4.sh
Entrez le nom d'un fichier existant: reponse1
Le fichier n'existe pas. Veuillez reessayer.
Entrez le nom d'un fichier existant: reponse2
Le fichier n'existe pas. Veuillez reessayer.
Entrez le nom d'un fichier existant: douda
Le fichier n'existe pas. Veuillez reessayer.
```

5. Créez un script qui demande un mot à l'utilisateur jusqu'à ce qu'il tape "Au revoir!".



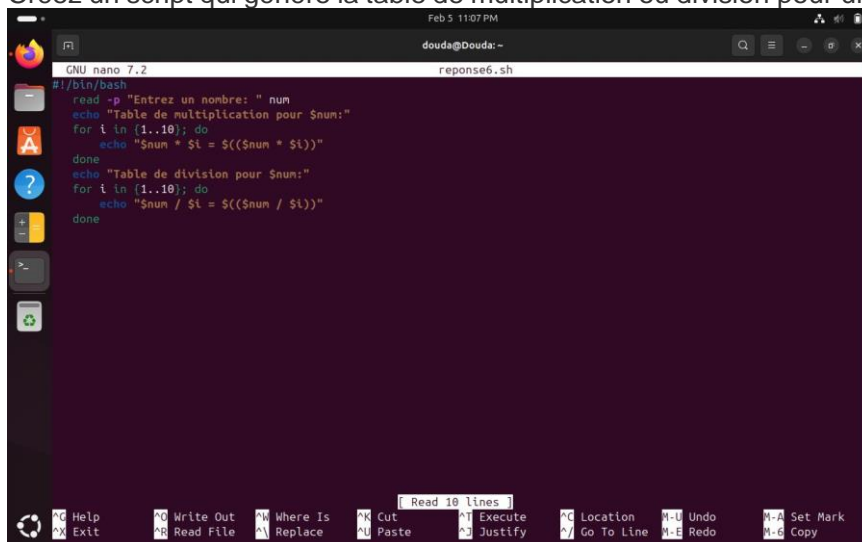
The screenshot shows a terminal window with a dark background. At the top, the title bar indicates the time as 'Feb 5 11:07 PM' and the user as 'douda@Douda: ~'. The terminal is running GNU nano 7.2, editing a file named 'reponse5.sh'. The script content is as follows:

```
#!/bin/bash
while true; do
  read -p "Entrez un mot (tapez 'Au revoir!' pour quitter): " word
  if [ "$word" = "Au revoir!" ]; then
    echo "Au revoir!"
    break
  else
    echo "Vous avez entre : $word"
  fi
done
```

Below the nano editor, the terminal shows the following commands and their outputs:

```
douda@Douda:~$ nano reponse5.sh
douda@Douda:~$ chmod +x reponse5.sh
douda@Douda:~$ ./reponse5.sh
Entrez un mot (tapez 'Au revoir!' pour quitter): A+
Vous avez entre : A+
Entrez un mot (tapez 'Au revoir!' pour quitter): Au revoir!
Au revoir!
douda@Douda:~$ nano reponse6.sh
```

6. Créez un script qui génère la table de multiplication ou division pour un nombre donné.



The screenshot shows a terminal window with a dark background. At the top, the title bar indicates the time as 'Feb 5 11:07 PM' and the user as 'douda@Douda: ~'. The terminal is running GNU nano 7.2, editing a file named 'reponse6.sh'. The script content is as follows:

```
#!/bin/bash
read -p "Entrez un nombre: " num
echo "Table de multiplication pour $num:"
for i in {1..10}; do
  echo "$num * $i = $((num * i))"
done
echo "Table de division pour $num:"
for i in {1..10}; do
  echo "$num / $i = $((num / i))"
done
```



```
douda@Douda:~$ nano reponse6.sh
douda@Douda:~$ chmod +x reponse6.sh
douda@Douda:~$ ./reponse6.sh
Entrez un nombre: 4
Table de multiplication pour 4:
4 * 1 = 4
4 * 2 = 8
4 * 3 = 12
4 * 4 = 16
4 * 5 = 20
4 * 6 = 24
4 * 7 = 28
4 * 8 = 32
4 * 9 = 36
4 * 10 = 40
Table de division pour 4:
4 / 1 = 4
4 / 2 = 2
4 / 3 = 1
4 / 4 = 1
4 / 5 = 0
4 / 6 = 0
4 / 7 = 0
4 / 8 = 0
4 / 9 = 0
4 / 10 = 0
douda@Douda:~$ nano reponse7.sh
```

7. Créez un Script pour Générer un Récit

```
GNU nano 7.2 reponse7.sh
#!/bin/bash
read -p "Entrez un personnage: " character
read -p "Entrez un lieu: " place
read -p "Entrez une action: " action
echo "Il était une fois $character qui vivait à $place. Un jour, $character décida de $action."
```

Feb 5 11:08 PM

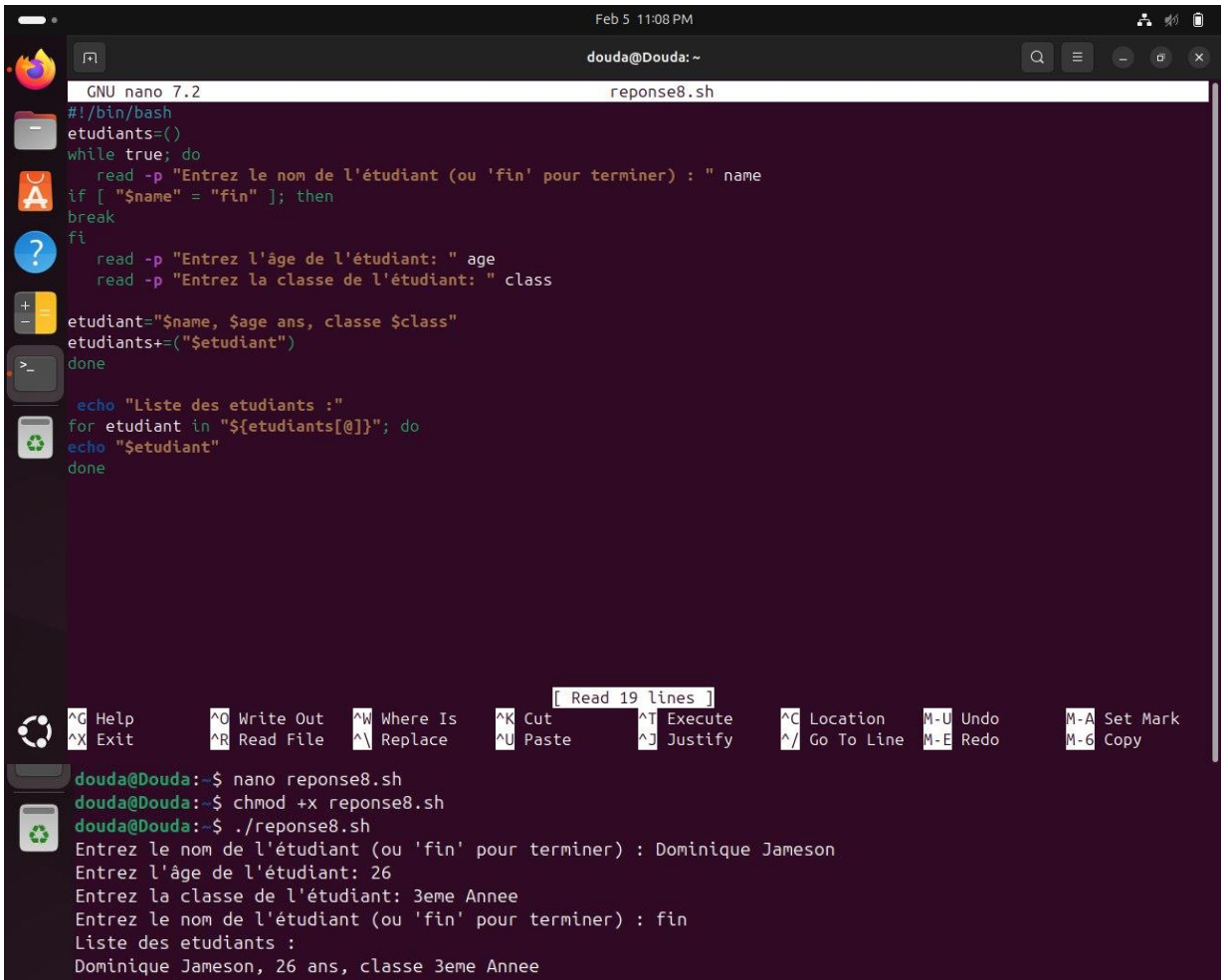
douda@Douda: ~

[Read 5 lines]

Help Exit Write Out Read File Where Is Replace Cut Paste Execute Justify Location Go To Line Undo Redo Set Mark Copy

```
douda@Douda:~$ nano reponse7.sh
douda@Douda:~$ chmod +x reponse7.sh
douda@Douda:~$ ./reponse7.sh
Entrez un personnage: Jameson Dominique
Entrez un lieu: Jacmel
Entrez une action: Conduire
Il était une fois Jameson Dominique qui vivait à Jacmel. Un jour, Jameson Dominique décida de Conduire.
douda@Douda:~$ nano reponse8.sh
```

8. Créez un Script Collecter les Informations d'étudiant

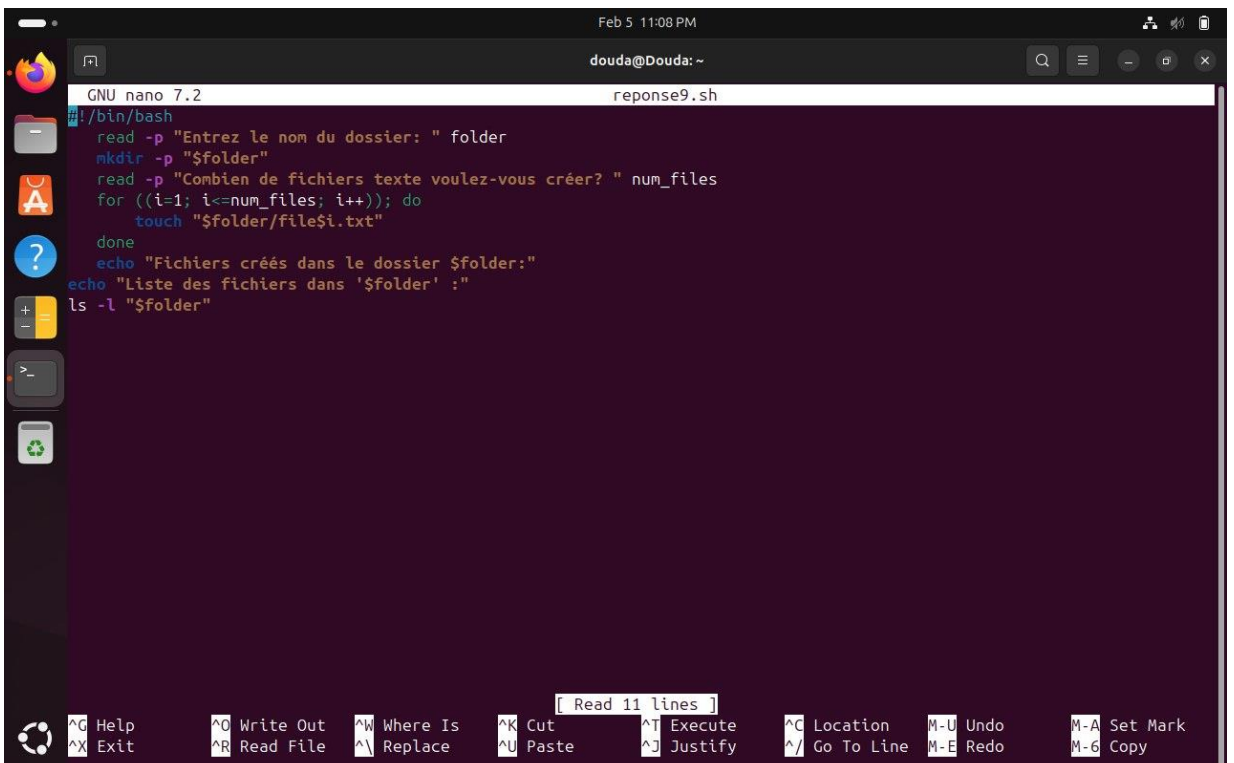


```
GNU nano 7.2 reponse8.sh
#!/bin/bash
etudiants=()
while true; do
  read -p "Entrez le nom de l'étudiant (ou 'fin' pour terminer) : " name
  if [ "$name" = "fin" ]; then
    break
  fi
  read -p "Entrez l'âge de l'étudiant: " age
  read -p "Entrez la classe de l'étudiant: " class
  etudiant="$name, $age ans, classe $class"
  etudiants+=("$etudiant")
done

echo "Liste des etudiants :"
for etudiant in "${etudiants[@]}; do
  echo "$etudiant"
done

douda@douda:~$ nano reponse8.sh
douda@douda:~$ chmod +x reponse8.sh
douda@douda:~$ ./reponse8.sh
Entrez le nom de l'étudiant (ou 'fin' pour terminer) : Dominique Jameson
Entrez l'âge de l'étudiant: 26
Entrez la classe de l'étudiant: 3eme Annee
Entrez le nom de l'étudiant (ou 'fin' pour terminer) : fin
Liste des etudiants :
Dominique Jameson, 26 ans, classe 3eme Annee
```

9. Créez un script qui demande à l'utilisateur de fournir le nom du dossier, demande à l'utilisateur combien de fichiers texte il souhaite créer, crée les fichiers texte et les ajoute au dossier, liste tous les fichiers créés.



```
GNU nano 7.2 reponse9.sh
#!/bin/bash
read -p "Entrez le nom du dossier: " folder
mkdir -p "$folder"
read -p "Combien de fichiers texte voulez-vous créer? " num_files
for ((i=1; i<=num_files; i++)); do
  touch "$folder/file$i.txt"
done
echo "Fichiers créés dans le dossier $folder:"
echo "Liste des fichiers dans '$folder' :"
ls -l "$folder"
```



```
douda@douda:~$ nano reponse9.sh
douda@douda:~$ chmod +x reponse9.sh
douda@douda:~$ ./reponse9.sh
Entrez le nom du dossier: Douda
Combien de fichiers texte voulez-vous créer? 3
Fichiers créés dans le dossier Douda:
Liste des fichiers dans 'Douda' :
total 0
-rw-rw-r-- 1 douda douda 0 Feb  5 23:21 file1.txt
-rw-rw-r-- 1 douda douda 0 Feb  5 23:21 file2.txt
-rw-rw-r-- 1 douda douda 0 Feb  5 23:21 file3.txt
douda@douda:~$
```

Conclusion

En conclusion, les scripts Bash présentés dans ces réponses couvrent une variété de tâches courantes en programmation shell, allant de la manipulation de boucles et de conditions à la gestion de fichiers et d'interactions utilisateur. Voici un résumé des points clés :

1. Boucles et conditions : Les scripts utilisent des boucles (for, while, until) et des conditions (if, case) pour gérer des flux de contrôle complexes, comme la répétition d'actions jusqu'à ce qu'une condition soit remplie.
2. Interactions utilisateur : Plusieurs scripts demandent à l'utilisateur de fournir des informations via read, ce qui permet de créer des programmes interactifs et dynamiques.
3. Manipulation de fichiers : Les scripts montrent comment créer, lire et manipuler des fichiers, ainsi que comment vérifier l'existence d'un fichier avant d'effectuer des opérations.
4. Opérations sur les variables : Les scripts illustrent comment effectuer des opérations arithmétiques de base (addition, soustraction, multiplication, division) sur des variables.
5. Génération de contenu : Certains scripts génèrent du contenu dynamique, comme des tables de multiplication ou des récits personnalisés, en fonction des entrées utilisateur.
6. Gestion de dossiers et fichiers : Un script montre comment créer des dossiers, générer des fichiers texte, et lister leur contenu, ce qui est utile pour automatiser des tâches de gestion de fichiers.