EXPERIMENT NO.: 2 **DATE:**

AIM: To implement the following programs using shell script:

1)To Find Difference of Largest And Smallest Digit Of a Number.

2)Menu Driven Program

i)To Find Which User Has Logged In

ii)Display Process Executed By User

iii)Display Number Of Files In Directory

3)Calculate Simple Intrest

4)Prime Number Upto N

5)generate pattern

1

2 2

3 3 3

4 4 4 4

THEORY:

A shell is a command-line interpreter and typical operations performed by shell scripts include file manipulation, program execution, and printing text.

To Run Shell Script program, do the following:

1. Save the file with "filename.sh" having the shell script code by creating a Directory and saving this file under that Directory Name say XYZ.
2. Go to cmd/ Linux shell and say "cd XYZ" and hit Enter.
3. Now say ./filename.sh to Run the file.

IF-ELSE Statement:

• The if...else...fi statement is the next form of control statement that allows Shell to execute statements in a controlled way and make the right choice.
• If the resulting value is true, given statement(s) are executed.
• If the expression is false, then no statement will be executed.
• Syntax:

```
if [ expression ] then
        Statement(s) to be executed if expression is true else
        Statement(s) to be executed if expression is not true
    fi
```

WHILE Loop:

• The while loop enables you to execute a set of commands repeatedly until some condition occurs.
• It is usually used when you need to manipulate the value of a variable repeatedly.
• If the resulting value is true, given statement(s) are executed.
• If command is false then no statement will be executed and the program will jump to the next line after the done statement.
• Syntax:

```
while command do
                Statement(s) to be executed if command is true

Done
```

CASE Condition:

• The basic syntax of the case...esac statement is to give an expression to evaluate and to execute several different statements based on the value of the expression.
• Here the string word is compared against every pattern until a match is found.
• The statement(s) following the matching pattern executes.

• If nothing matches, a default condition will be used.
• When statement(s) part executes, the command ;; indicates that the program flow should jump to the end of the entire case statement. This is similar to break in the C programming language.
• Syntax:

```
case word in
                pattern1)
                        Statement(s) to be executed if pattern1 matches
                        ;;
                pattern2)
                        Statement(s) to be executed if pattern2 matches
                        ;;
                *)
                        Default condition to be executed
                        ;;
        Esac
```

FOR Loop:

• The for loop operate on lists of items.
• It repeats a set of commands for every item in a list.
• Here var is the name of a variable and word 1 to word N are sequences of characters separated by spaces (words).
• Syntax:

```
for var in word 1 word 2 ...word n
            do
                        Statement to be executed

done
```

1)To Find Difference of Largest And Smallest Digit Of a Number.

```
echo "Enter a number"
read num
max=0
min=9
temp=$num

while [ $temp -gt 0 ]
do
  digit=$((temp % 10))
  if [ $digit -gt $max ]; then
    max=$digit
  fi
  if [ $digit -lt $min ]; then
    min=$digit
  fi
  temp=$((temp / 10))
done
echo "max: $max"
echo "min: $min"

diff=$((max - min))
echo "Difference between largest and smallest digit is: $diff"
```

```
audumber@Acer-Nitro-V:/mnt/c/Audumber/shells$ sh diff.sh
Enter a number
456
max: 6
min: 4
Difference between largest and smallest digit is: 2
```

2)Menu Driven Program

1)To Find Which User Has Logged In

2)Display Process Executed By User

3)Display Number Of Files In Directory

```
echo "Menu"
echo "(1) Print user"
echo "(2) Print process"
echo "(3) Print number of files in directory"

read -p "Enter your choice: " choice

case $choice in
 1)
  echo "Current user: $(whoami)"
  ;;
 2)
  echo "Running processes:"
  ps
  ;;
 3)
```

```
audumber@Acer-Nitro-V:/mnt/c/Audumber/shells$ sh menu.sh
Menu
(1) Print user
(2) Print process
(3) Print number of files in directory
Enter your choice: 1
Current user: audumber
audumber@Acer-Nitro-V:/mnt/c/Audumber/shells$ sh menu.sh
Menu
(1) Print user
(2) Print process
(3) Print number of files in directory
Enter your choice: 2
Running processes:
  PID TTY          TIME CMD
    9 pts/0    00:00:00 bash
   62 pts/0    00:00:00 sh
   63 pts/0    00:00:00 ps
audumber@Acer-Nitro-V:/mnt/c/Audumber/shells$ sh menu.sh
Menu
(1) Print user
(2) Print process
(3) Print number of files in directory
Enter your choice: 3
Enter directory path: /
Number of items in '/': 25
```

```
  read -p "Enter directory path: " dir
  if [ -d "$dir" ]; then
    count=$(ls -1 "$dir" | wc -l)
    echo "Number of items in '$dir': $count"
  else
    echo "Directory does not exist."
  fi
  ;;
 *)
  echo "Invalid choice."
  ;;
esac
```

3)Calculate Simple Intrest

```
#!/bin/bash

echo "Simple Interest Calculator"

read -p "Enter principal amount: " principal
read -p "Enter rate of interest (in %): " rate
read -p "Enter time (in years): " time

simple_interest=$(( principal * rate * time / 100 ))

echo "Simple Interest = $simple_interest"
```

```
audumber@Acer-Nitro-V:/mnt/c/Audumber/shells$ bash simpleintrest.sh
Simple Interest Calculator
Enter principal amount: 100
Enter rate of interest (in %): 3
Enter time (in years): 4
Simple Interest = 12
```

4)Prime Number Upto N

```
#!/bin/bash

read -p "Enter a number (n): " n

echo "Prime numbers up to $n are:"

for (( num=2; num<=n; num++ ))
do
 is_prime=1

 for (( i=2; i*i<=num; i++ ))
 do
  if (( num % i == 0 )); then
   is_prime=0
   break
  fi
 done
```

```
audumber@Acer-Nitro-V:/mnt/c/Audumber/shells$ bash primenumber.sh
Enter a number (n): 6
Prime numbers up to 6 are:
2
3
5
```

```
 if (( is_prime == 1 )); then
  echo "$num"
 fi
done
```

5)Generate Pattern

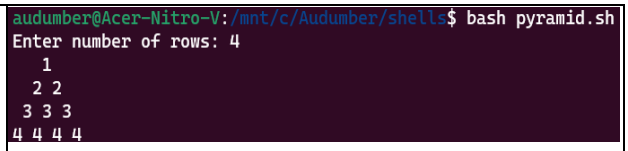
1

2 2

3 3 3

4 4 4 4

```
#!/bin/bash

read -p "Enter number of rows: " n

for (( i=1; i<=n; i++ ))
do

 for (( j=i; j<n; j++ ))
 do
  echo -n " "
 done

 for (( k=1; k<=i; k++ ))
 do
  echo -n "$i "
 done

 echo
done
```



CONCLUSION:

The Shell Script programs were executed successfully without any errors.