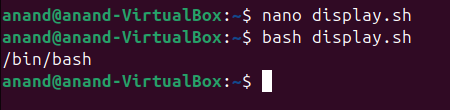
1. Display your current shell on the terminal using **echo**​ command.

$ nano display.sh

$ echo $SHELL

$ bash display.sh

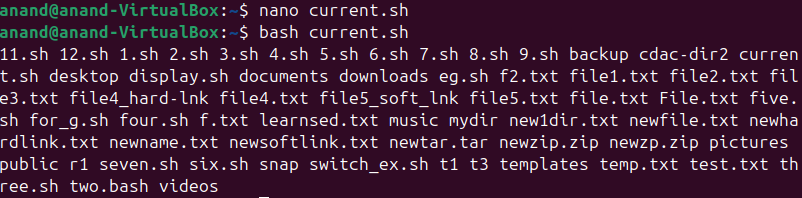


1. Display files and directories in the present working directory using **echo** command.

$ nano current.sh

$ echo $(ls)

$ bash. Current.sh



3. create 4 files with **.txt**​ extension and 4 files with **.c**​ extension - display only **.txt** ​extension files using **echo** ​command in present working directory.

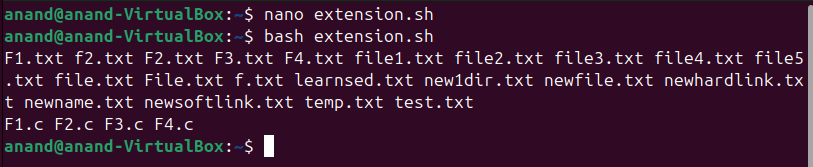
- display only **.c** ​extension files using **echo** ​command in present working directory.(Hint: use **touch**​ command to create files)

$ nano extension.sh

$ touch F1.txt F2.txt F3.txt F4.txt

$ touch F1.c F2.c F3.c F4.c

$ bash extension.sh



1. Display all environment variables on the terminal using command and verify a few of them by printing them using **echo**​ command. (Hint: use **printenv** command)

$ nano variable.sh

$ Printenv

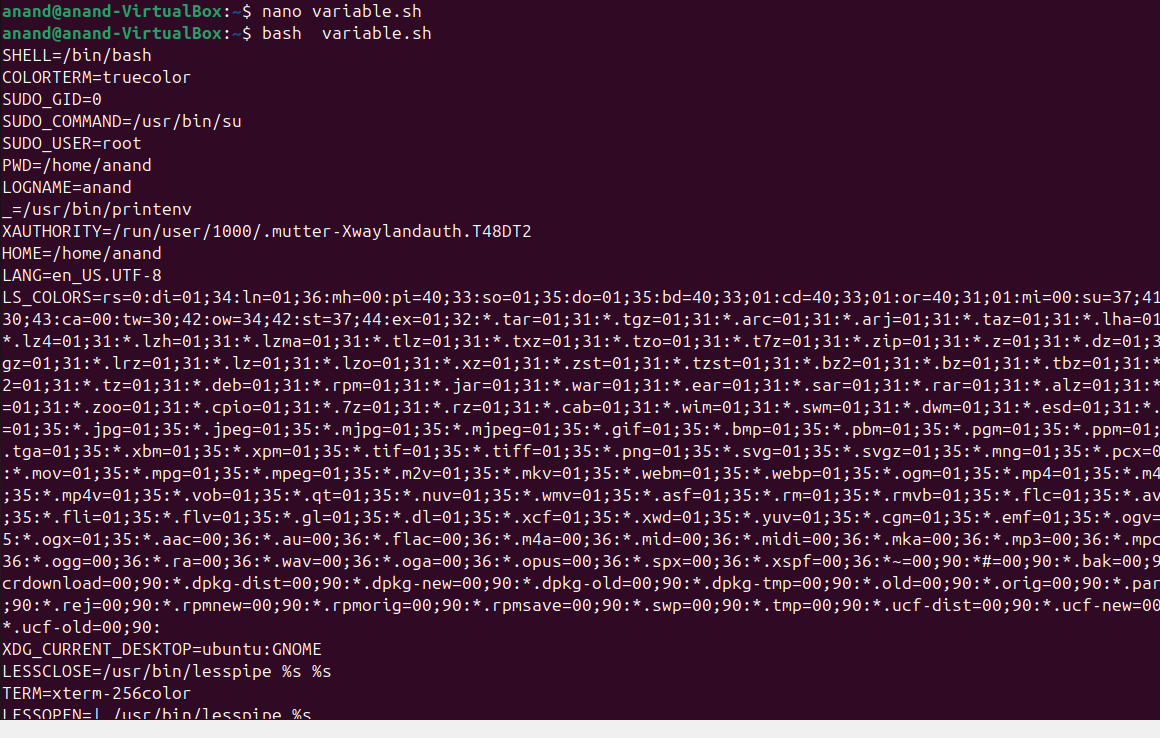
$ echo $SHELL

$ echo $USER

$ echo $PWD

$ echo $PATH

$ bash variable.sh



**Part 1**​ **(Shell scripting)**

**Note1:**​ For the following assignments, accept only integer values from user.If user enters a value other than integer display error message and terminate the program. **Note2:**​ If you want to perform the following arithmetic operations with Floating point values make use of the Basic **calculator** ​in your script.

(**Hint:**​ use **bc**​ command for Basic calculator)

$ nano integer.sh

$ is\_integer() {

$ if ! [ [ “$1” =~ ^[0-9]+$ ] ]

$ then

$ echo “Error:Please entre to intger’

$ exit 1

$ fi

$ bash integer.sh

1. Write a shell script to Print prime numbers from 1 to **n**​. Read the value of **n** from the user.
2. Write a script to check if a given number is even or odd.

$ nano even.sh

$ read -p “enter to num:” a

$ if [ $((a % 2)) = 0 ]

$ then

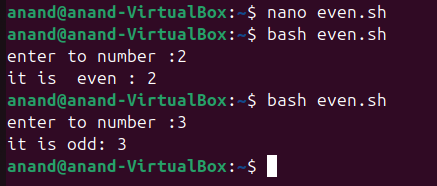
$ echo “it is even :$a’

$ else

$ echo “it is odd:$a’

$ fi

$ bash even.sh



1. Write a shell script to convert a decimal number to binary number.

$ nano binary.sh

$ read -p “enter to decimal “ decimal

$ decimal=$1

$ binary=””

$ while[ $decimal -gt 0 ]

$ do

$ remainder=$((decimal % 2 ))

$ binary=$remainder$binary

$ decimal=$((decimal / 2))

$ done

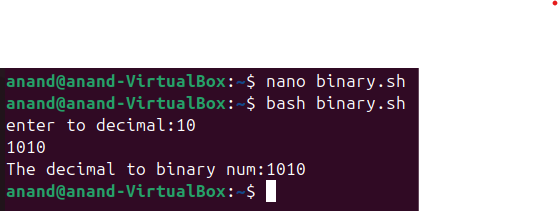
$ if [ -z $decimal ]

$ then

$ $binary=0

$ fi

$echo “decimal to binary :” $binary



4. Write a script to **swap**​ 2 numbers using an intermediate variable.

Example:

input : 12 output: 21

input : 213 output: 312

input : 125634 output:436521

$ nano swap.sh

$ read -p “enter to number:” a

$ read -p “enter to number:” b

$ temp=$a

$ a=$b

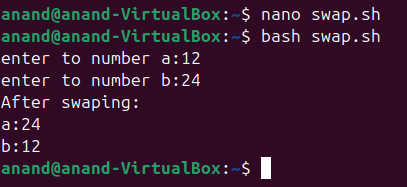
$ b=$temp

$ echo “After swaping:”

$ echo “a:$a”

$ echo “b:$b”

$ bash swap.sh



1. Write a script to **swap**​ 2 numbers without using an intermediate variable.

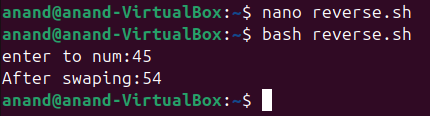
$ nano revers.sh

$ read -p “enter to number:” num

$ reverse=$(echo “$num” | rev )

$ echo “After swaping : $reverse”

$ bash revers.sh



1. Write a script to reverse a number using a while loop.

$ nano REVERS.sh

$ read -p “enter to number “ num

$ original\_num=$num

$ reverse=0

$ while [ $num -gt 0 ]

$ do

$ remainder=$((num % 10 ))

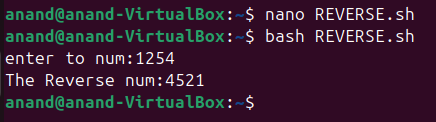
$ reverse=$((reverse \* 10 + remainder))

$ num=$((num / 10 ))

$ done

$echo “the reverse number:” $reverse

$ bash REVERS.sh



1. print multiplication table of integer using while loop.

$ nano multiplction.sh

$ read -p “enter to num :” num

$ i=1

$ while [ $i -le 10 ]

$ do

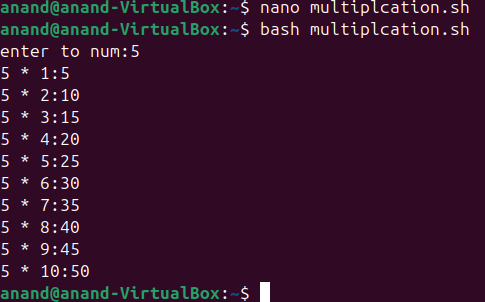
$ result=$((num \* i))

$ echo “$num \* i :“ $result

$ i=$((i + 1))

$ done

$ bash multiplication.sh



1. Get year as an input from user and find whether year is leap year or not.

$ nano leap.sh

$ read -p “enter to year” year

$ if [ $((year % 4 )) -ne 0 ]

$ then

$ echo “ $year it is not leap year”

$elief [ $((year % 100)) -ne 0 ]

$ then

$ echo “$year it is leap year”

$ elief [ $((year % 400)) -eq 0 ]

$ then

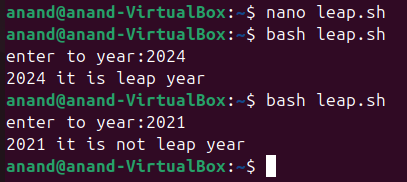
$ echo “$year it is leap year”

$ else

$ echo”$year is not a leap year”

$ fi

$ bash leap.sh



1. Write a script to read the number of rows to be displayed in the pattern and print following pattern using for loop:

$ nano pattern.sh

$ read -p “enter to num:” rows

$ num=1

$ for (( i=1; i<=rows; i++));do

$ for (( j=1; j<=I; j++));do

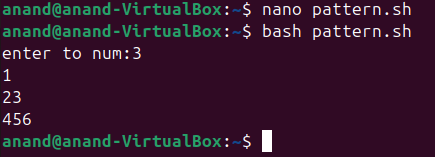
$ echo -n “$num”

$ num=$((num + 1))

$ done

$ echo

$ done



10. Write a script using **case**​ condition to do the following

- Display “Press any key of keyboard and then press enter key”

- If the given input is a number, display “The input is digit.” message - If the given input is lowercase letter then display “The input is lowercase Letter.” message

- If the given input is uppercase letter then display “The input is Uppercase letter.” message

**echo "Press any key of keyword and then press enter key:"**

**read input**

**case $input in**

**[0-9])**

**echo "The input is digit"**

**;;**

**[a-z])**

**echo "The input is lowercase letter"**

**;;**

**[A-Z])**

**echo "The input is uppercase letter"**

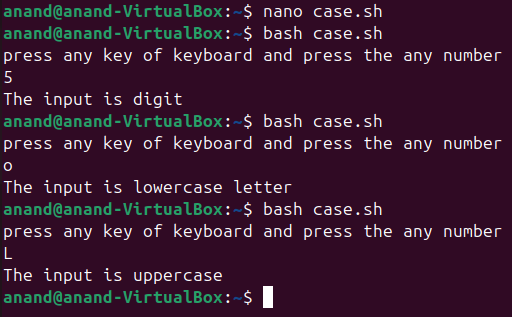
**;;**

**\*)**

**echo "Invalid "**

**;;**

**esac**

****

11.Write a for loop to display the outputs of **Date,pwd, df** ​commands. **df** command displays system disk usage details.

(**Hint:**​ give these commands as input to for loop)

$ nano command.sh

$ for command in “date” “pwd” “df”

$ do

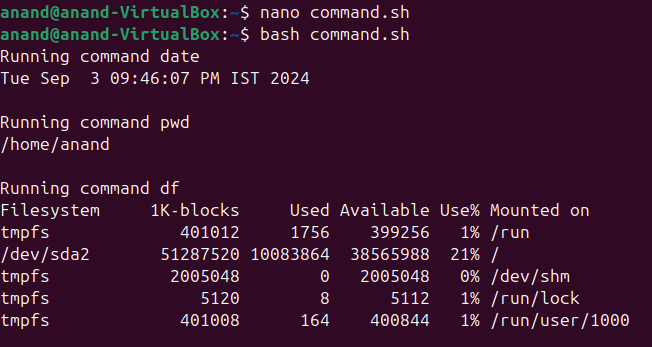
$ echo “Running command:”$command

$ $command

$ echo

$ done

$ bash command.sh



12.Write a script to take filename as argument and display whether the file exists or not **Note:**​ If the file exists in current working directory just give filename as argument, If not give absolute path of that file as argument.

**$ nano twelve.sh**

**$ read -p “enter to filename:” fname**

**$ if [ -f $fname ]**

**$ then**

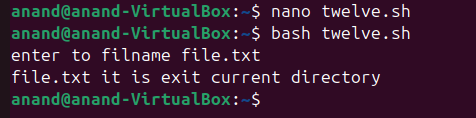
**$ echo “ $fname is exit in current directory”**

**$else**

**$ echo “ $fname is not exit in current directory”**

**$fi**

**$bash twelve.sh**



13.Write a script to take the directory name as argument and display whether the directory exists or not.

**Note:**​ If the directory exists in the current working directory just give filename as argument, If not give absolute path of that file as argument.

14.Read a file and display the contents of the file line by line using a for loop and pass the file as command line argument to the script.

15.Read a file and display the contents of the file line by line using a while loop and pass the file as command line argument to the script.

17.Write a shell script to read array elements as command line arguments assign the arguments to array and do the following

- Display the length of the array.

- Display all elements and their index values.

Example :

let the array elements are as follows

arr[0]=”zero”, arr[1]=”one”, arr[2]=”two”,

output should be :

length of the array : 3

index 0 element is “zero”

index 1 element is “one”

index 2 element is “two”

18. Write a single shell script to do the following

- read two numbers as input from the terminal.

- write Add,Sub and Mul functions to perform addition,subtraction and multiplication between two integers.

- Display the results(**Note:** ​To display float results using a basic calculator.)

**# Prompt the user to enter two numbers**

**echo "Enter the first number:"**

**read num1**

**echo "Enter the second number:"**

**read num2**

**# Prompt the user to choose an operation**

**echo "Choose an operation (+, -, \*):"**

**read operation**

**# Use a case statement to perform the chosen operation**

**case $operation in**

**+)**

**result=$((num1 + num2))**

**echo "The result of $num1 + $num2 is: $result"**

**;;**

**-)**

**result=$((num1 - num2))**

**echo "The result of $num1 - $num2 is: $result"**

**;;**

**\\*)**

**result=$((num1 \* num2))**

**echo "The result of $num1 \* $num2 is: $result"**

**;;**

**\*)**

**echo "Invalid operation. Please choose +, -, \*, or /."**

**;;**

**esac**

