1. **Practice Basic Shell Commands like:- ls, cd, du, pwd, man, cat, more, less, head, tail, mkdir, cp, mv, rm, touch, grep, sort, wc, cut, echo…**

**ls -** used to display a list of content of a directory.

**cd -** used to change the current directory.

**du** - used to check the information of disk usage of files and directories on a system.

**pwd -** used to display the location of the current working directory.

**man -** used to display the user manual of any command that we can run on the terminal.

**cat** - used as a filter, to filter a file, it is used inside pipes.

**more** - reads files and displays the text one screen at a time.

**less** - used for filtering and viewing text files one screen page at a time.

**head** - prints the first 10 lines of the specified files.

**tail** - prints the last few number of lines of a certain file, then terminates.

**mkdir** - used to create a new directory under any directory.

**cp** - used to copy a file or directory.

**mv** - used to move a file or a directory form one location to another location.

**rm** - used to remove a file.

**touch** - Create a new file or update its timestamp.

**grep** - To search for matching patterns in a file.

**sort** - used to sort files in alphabetical order.

**wc** - used to count the lines, words, and characters in a file.

**cut** - used to select a specific column of a file.

**echo** - for displaying lines of text or string which are passed as arguments on the command line

1. **Write a Shell program to check the given number is even or odd.**

echo -n "Enter a number:"

read n

echo -n "RESULT: "

if [ `expr $n % 2` == 0 ]

then

echo "$n is even"

else

echo "$n is Odd"

fi

**OUTPUT**



1. **Write a Shell program to check a leap year.**

echo -n "Enter a year:"

read year\_checker

if [ `expr $year\_checker % 4` -eq 0 ]

then

echo "$year\_checker is a leap year"

else

echo "$year\_checker is not a leap year"

fi

**OUTPUT**



1. **Write a Shell program to find the area and circumference of a circle.**

echo "Enter the radious of the circle"

read r

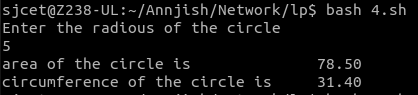
area=$(echo "3.14\*$r\*$r" | bc )

circum=$(echo "3.14\*2\*$r" | bc)

echo "area of the circle is " $area

echo "circumference of the circle is " $circum

**OUTPUT**

****

1. **Write a Shell program to check the given number and its reverse are same.**

read -p "Enter a number: " number

temp=$number

while [ $temp -ne 0 ]

do

reverse=$reverse$((temp%10))

temp=$((temp/10))

done

echo "Reverse of $number is $reverse"

**OUTPUT**

****

1. **Write a Shell program to check the given string is palindrome or not.**

echo "Enter a String"

read input

reverse=""

len=${#input}

for (( i=$len-1; i>=0; i-- ))

do

reverse="$reverse${input:$i:1}"

done

if [ $input == $reverse ]

then

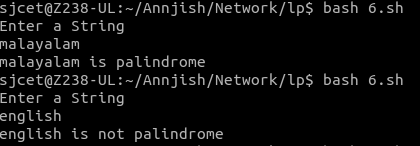
echo "$input is palindrome"

else

echo "$input is not palindrome"

fi

**OUTPUT**



1. **Write a Shell program to find the sum of odd and even numbers from a set of numbers.**

echo "Enter the Numbers"

read num

rev=0

even=0

odd=0

while [ $num -gt 0 ]

do

tmp=$(( $num % 10 ))

if(( $tmp % 2 == 0 ))

then

even=$(( $even + $tmp ))

else

odd=$(( $odd + $tmp ))

fi

rev=$(( $rev \* 10 + $tmp ))

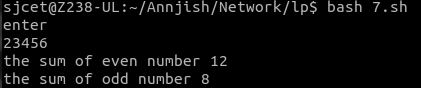
num=$(( $num / 10 ))

done

echo the sum of even number $even

echo the sum of odd number $odd

**OUTPUT**



1. **Write a Shell program to find the roots of a quadratic equation.**

echo Enter the coefficient of x^2:

read a

echo Enter the coefficient of x:

read b

echo Enter the constant term:

read c

f=`echo "-($b)" |bc`

p=`expr 2 \\* $a`

if [ $a -ne 0 ]

then

d=`echo \( \( $b \\* $b \) - \( 4 \\* $a \\* $c \) \) | bc`

if [ $d -lt 0 ]

then

x=`echo "-($d)" | bc`

s=`echo "scale=2; sqrt ( $x )" | bc`

echo The first root is:

echo "($f + $s i) / $p"

echo The second root is:

echo "($f - $s i) / $p"

elif [ $d -eq 0 ]

then

res=`expr $f / $p`

echo The root is: $res

else

s=`echo "scale=2; sqrt( $d )" | bc`

res1=`echo "scale=2; ( $f + $s) / ( $p )"|bc`

res2=`echo "scale=2; ( $f - $s) / ( $p )"|bc`

echo The first root is: $res1

echo The second root is: $res2

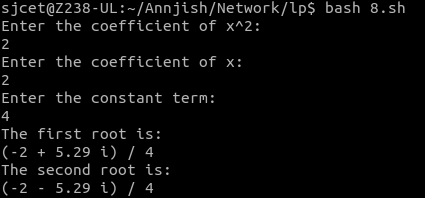
fi

else

echo Coefficient of x^2 can not be 0.

fi

**OUTPUT**



1. **Write a Shell program to check the given integer is Armstrong number or not.**

echo "Enter the number"

read n

function ams

{

t=$n

s=0

b=0

c=10

while [ $n -gt $b ]

do

r=$((n % c))

i=$((r \* r \* r))

s=$((s + i))

n=$((n / c))

done

if [ $s == $t ]

then

echo "Amstrong number"

else

echo "Not an Armstrong number"

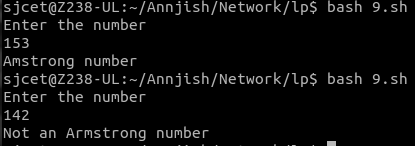
fi

}

result=`ams $n`

echo "$result"

**OUTPUT**



1. **Write a Shell program to check the given integer is prime or not.**

echo -e "Enter Number : "

read n

while [ $n -gt 2 ]

do

for((i=2; i<=$n/2; i++))

do

ans=$(( n%i ))

if [ $ans -eq 0 ]

then

echo "$n is not a prime number."

exit 0

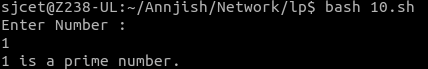
fi

done

done

echo "$n is a prime number."

**OUTPUT**



1. **Write a Shell program to generate prime numbers between 1 and 50.**

echo "Enter a limit"

read limit

echo "prime numbers upto $limit are :"

echo "1"

i=2

while [ $i -le $limit ]

do

flag=1

j=2

while [ $j -lt $i ]

do

rem=$(( $i % $j ))

if [ $rem -eq 0 ]

then

flag=0

break

fi

j=$(( $j+1 ))

done

if [ $flag -eq 1 ]

then

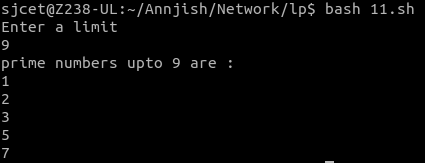
echo "$i"

fi

i=$(( $i+1 ))

done

**OUTPUT**



1. **Write a Shell program to find the sum of square of individual digits of a number.**

echo "Enter a number: "

read number

sum=0

while [ $number -ne 0 ]

do

digit=$((number % 10))

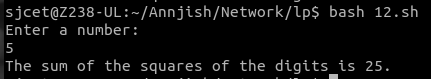
sum=$((sum + digit \* digit))

number=$((number / 10))

done

echo "The sum of the squares of the digits is $sum."

**OUTPUT**



1. **Write a Shell program to count the number of vowels in a line of text.**

echo "Enter a string to find the number of Vowels "

read st

len=`expr $st | wc -c`

len=`expr $len - 1`

count=0

while [ $len -gt 0 ]

do

ch=`expr $st | cut -c $len`

case $ch in

[aeiou,AEIOU]) count=`expr $count + 1` ;;

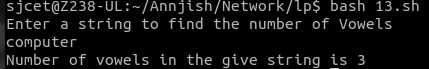
esac

len=`expr $len - 1`

done

echo "Number of vowels in the give string is $count"

**OUTPUT**



1. **Write a Shell program to display student grades.**

echo '\tStudent Mark List'

echo -----------------------------------

echo Enter the Student name

read name

echo Enter the Register number

read rno

echo Enter the Mark1

read m1

echo Enter the Mark2

read m2

echo Enter the Mark3

read m3

echo Enter the Mark4

read m4

echo Enter the Mark5

read m5

tot=$(expr $m1 + $m2 + $m3 + $m4 + $m5)

avg=$(expr $tot / 5)

echo -----------------------------------

echo '/t Student Mark List'

echo -----------------------------------

echo "Student Name : $name"

echo "Register Number : $rno"

echo "Mark1 : $m1"

echo "Mark2 : $m2"

echo "Mark3 : $m3"

echo "Mark4 : $m4"

echo "Mark5 : $m5"

echo "Total : $tot"

echo "Average : $avg"

if [ $m1 -ge 35 ] && [ $m2 -ge 35 ] && [ $m3 -ge 35 ] && [ $m4 -ge 35 ] && [ $m5 -ge 35 ]

then

echo "Result : Pass"

if [ $avg -ge 90 ]

then

echo "Grade : S"

elif [ $avg -ge 80 ]

then

echo "Grade : A"

elif [ $avg -ge 70 ]

then

echo "Grade : B"

elif [ $avg -ge 60 ]

then

echo "Grade : C"

elif [ $avg -ge 50 ]

then

echo "Grade : D"

elif [ $avg -ge 35 ]

then

echo "Grade : E"

fi

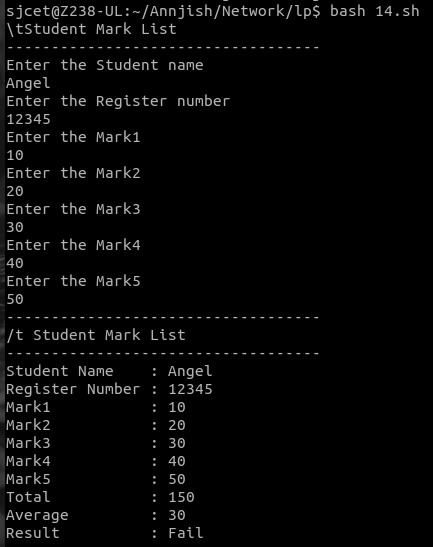
else

echo "Result : Fail"

fi

echo -----------------------------------

**OUTPUT**



1. **Write a Shell program to find the smallest and largest numbers from a set of numbers.**

echo "Enter a list of numbers separated by spaces: "

read numbers

IFS=' ' read -ra nums <<< "$numbers"

min=${nums[0]}

max=${nums[0]}

for num in "${nums[@]}"

do

if (( num < min )); then

min=$num

fi

if (( num > max )); then

max=$num

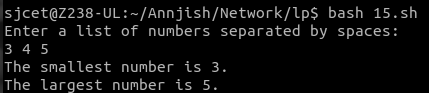
fi

done

echo "The smallest number is $min."

echo "The largest number is $max."

**OUTPUT**



1. **Write a Shell program to find the smallest digit from a number.**

echo "Enter a number: "

read num

min=${num:0:1}

for (( i=1; i<${#num}; i++ ))

do

digit=${num:$i:1}

if (( digit < min )); then

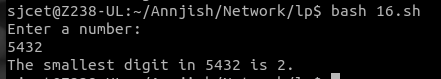
min=$digit

fi

done

echo "The smallest digit in $num is $min."

**OUTPUT**



1. **Write a Shell program to find the sum of all numbers between 50 and 100, which are divisible by 3 and not divisible by 5.**

echo "Enter a number: "

read num

min=${num:0:1}

for (( i=1; i<${#num}; i++ ))

do

digit=${num:$i:1}

if (( digit < min )); then

min=$digit

fi

done

echo "The smallest digit in $num is $min."

**OUTPUT**



1. **Write a Shell program to find the second highest number from a set of numbers.**

echo "Enter a set of numbers separated by spaces: "

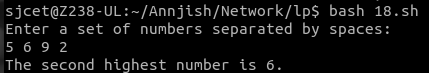
read numbers

arr=($numbers)

sorted\_arr=($(echo "${arr[@]}" | tr " " "\n" | sort -rn))

echo "The second highest number is ${sorted\_arr[1]}."

**OUTPUT**



1. **Write a Shell program to find the sum of digits of a number using function.**

sum\_of\_digits() {

num=$1

sum=0

while [ $num -gt 0 ]

do

digit=$((num % 10))

sum=$((sum + digit))

num=$((num / 10))

done

echo $sum

}

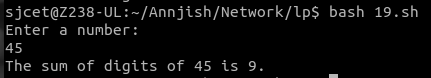
echo "Enter a number: "

read num

result=$(sum\_of\_digits $num)

echo "The sum of digits of $num is $result."

**OUTPUT**



1. **Write a Shell program to print the reverse of a number using function.**

reverse\_number() {

num=$1

rev=0

while [ $num -gt 0 ]

do

digit=$((num % 10))

rev=$((rev \* 10 + digit))

num=$((num / 10))

done

echo $rev

}

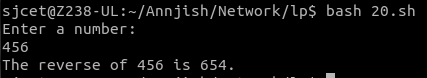
echo "Enter a number: "

read num

result=$(reverse\_number $num)

echo "The reverse of $num is $result."

**OUTPUT**



1. **Write a Shell program to find the factorial of a number using for loop.**

echo "Enter a number: "

read num

factorial=1

for (( i=1; i<=$num; i++ ))

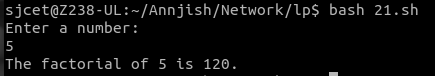
do

factorial=$((factorial \* i))

done

echo "The factorial of $num is $factorial."

**OUTPUT**



1. **Write a Shell program to generate Fibonacci series.**

echo "Enter the number of terms to generate: "

read num

a=0

b=1

echo -n "$a $b"

for (( i=3; i<=$num; i++ ))

do

c=$((a + b))

echo -n " $c"

a=$b

b=$c

done

emy\_list=("string1" "string2" "string3" "string4")

my\_list=($(echo "${my\_list[@]}" | tr ' ' '\n' | tac | tr '\n' ' '))

for i in "${!my\_list[@]}"

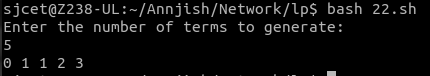
do

my\_list[$i]=`echo ${my\_list[$i]} | rev`

done

echo "${my\_list[@]}"cho

**OUTPUT**



1. **Write a shell script, which receives two filenames as arguments. It checks whether the two files contents are same or not. If they are same then second file is deleted.**

if [ $# -ne 2 ]; then

echo "Usage: $0 file1 file2"

exit 1

fi

if cmp -s "$1" "$2"; then

echo "The contents of $1 and $2 are the same. Deleting $2..."

rm "$2"

else

echo "The contents of $1 and $2 are different."

fi

**OUTPUT**



1. **Write a Menu driven Shell script that Lists current directory, Prints Working Directory, displays Date and displays Users logged in**

while true

do

clear

echo "Menu:"

echo "1. List current directory"

echo "2. Print working directory"

echo "3. Display date"

echo "4. Display users logged in"

echo "5. Exit"

read -p "Enter your choice: " choice

case $choice in

1)

ls -l

read -p "Press enter to continue"

;;

2)

pwd

read -p "Press enter to continue"

;;

3)

date

read -p "Press enter to continue"

;;

4)

who

read -p "Press enter to continue"

;;

5)

exit 0

;;

\*)

echo "Invalid choice. Press enter to try again"

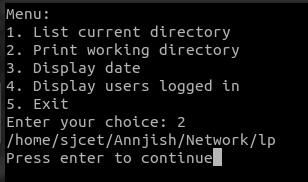
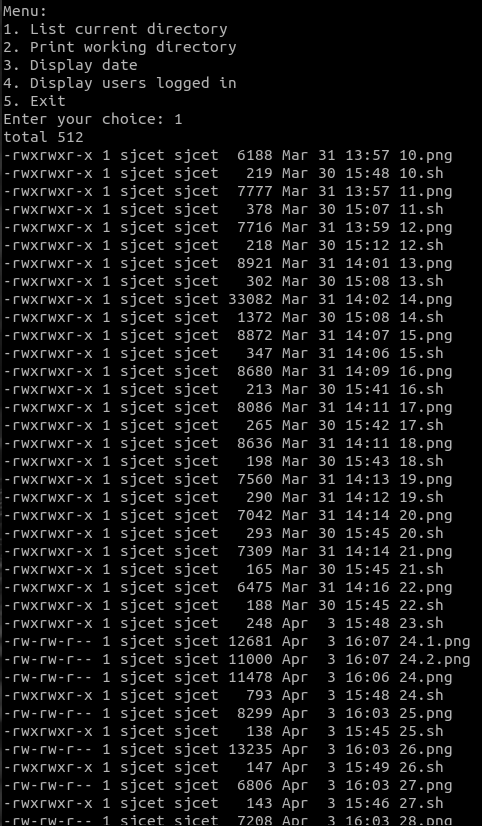
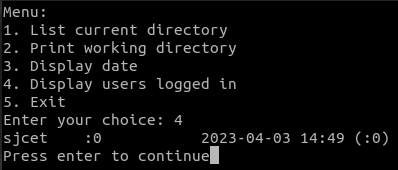
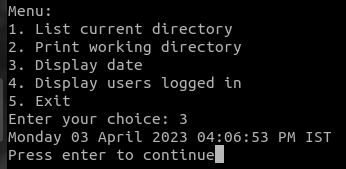
read

;;

esac

done

**OUTPUT**



1. **Shell script to check executable rights for all files in the current directory, if a file does not have the execute permission then make it executable.**

find . -type f | while read file; do

if [ ! -x "$file" ]; then

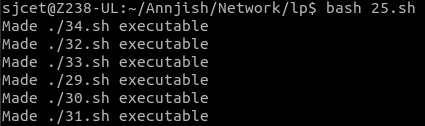
chmod +x "$file"

echo "Made $file executable"

fi

done

**OUTPUT**



1. **Write a Shell program to generate all combinations of 1, 2, and 3 using loop.**

for i in 1 2 3

do

for j in 1 2 3

do

for k in 1 2 3

do

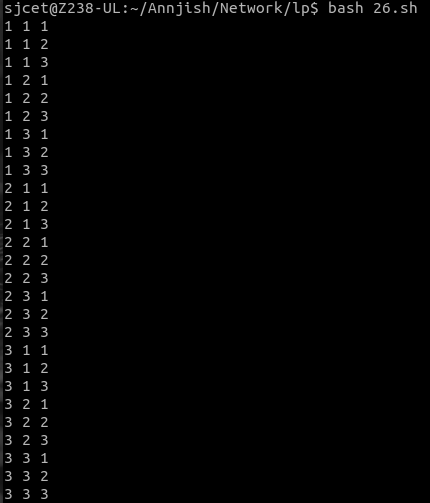
echo "$i $j $k"

done

done

done

**OUTPUT**



1. **Write a Shell program to create the number series.**

1

2 3

4 5 6

7 8 9 10

count=1

rows=4

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

do

for ((j=1; j<=i; j++))

do

echo -n "$count "

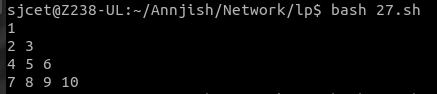
count=$((count+1))

done

echo ""

done

**OUTPUT**



1. **Write a Shell program to create Pascal’s triangle.**

function binom {

if [ $2 -eq 0 ] || [ $2 -eq $1 ]; then

echo 1

else

echo $(( $(binom $(($1-1)) $(($2-1))) + $(binom $(($1-1)) $2) ))

fi

}

echo "Enter the number of rows in Pascal's triangle: "

read rows

for (( i=0; i<$rows; i++ )); do

for (( j=0; j<=$i; j++ )); do

val=$(binom $i $j)

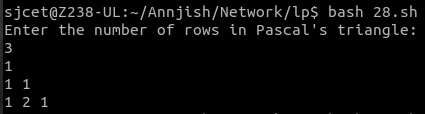
echo -n "$val "

done

echo ""

done

**OUTPUT**



1. **Write a Decimal to Binary Conversion Shell Script**

echo "Enter a decimal number: "

read decimal

binary=""

while [ $decimal -gt 0 ]; do

remainder=$((decimal % 2))

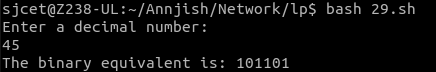
binary="$remainder$binary"

decimal=$((decimal / 2))

done

echo "The binary equivalent is: $binary"

**OUTPUT**



1. **Write a Shell Script to Check Whether a String is Palindrome or not**

echo "Enter a string: "

read string

reverse=$(echo $string | rev)

if [ "$string" == "$reverse" ]; then

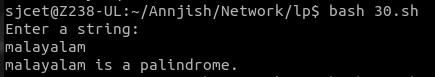
echo "$string is a palindrome."

else

echo "$string is not a palindrome."

fi

**OUTPUT**



1. **Write a shell script to find out the unique words in a file and also count the occurrence of each of these words.**

echo "Enter the file name: "

read file

if [ ! -f "$file" ]; then

echo "File not found."

exit 1

fi

contents=$(tr '[:upper:]' '[:lower:]' < $file | sed 's/[^a-z0-9]/ /g')

words=($contents)

declare -A count

for word in "${words[@]}"; do

if [ -n "$word" ]; then

((count[$word]++))

fi

done

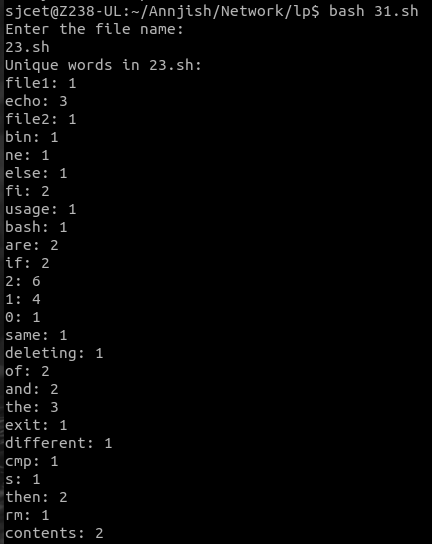
echo "Unique words in $file:"

for word in "${!count[@]}"; do

echo "$word: ${count[$word]}"

done

**OUTPUT**



1. **Write a shell script to get the total count of the word “Linux” in all the “.txt” files and also across files present in subdirectories.**

search\_dir="."

files=$(find "$search\_dir" -type f -name "\*.txt")

count=0

for file in $files; do

occurrences=$(grep -o "Linux" "$file" | wc -l)

count=$((count + occurrences))

done

echo "Total count of 'Linux' in all .txt files: $count"

**OUTPUT**



1. **Write a shell script to validate password strength. Here are a few assumptions for the password string.**

Length – minimum of 8 characters.

Contain both alphabet and number.

Include both the small and capital case letters.

read -p "Enter your password: " password

if [[ ${#password} -lt 8 ]]; then

echo "Password length must be at least 8 characters."

exit 1

fi

if ! [[ "$password" =~ [A-Za-z]+[0-9]+ ]]; then

echo "Password must contain both alphabet and number."

exit 1

Fi

if ! [[ "$password" =~ [a-z]+ ]] || ! [[ "$password" =~ [A-Z]+ ]]; then

echo "Password must include both small and capital case letters."

exit 1

fi

echo "Password is valid."

**OUTPUT**



1. **Write a shell script to print the count of files and subdirectories in the specified directory.**

if [ $# -eq 0 ]; then

echo "Usage: $0 directory"

exit 1

fi

directory=$1

if [ ! -d $directory ]; then

echo "Error: $directory is not a directory"

exit 1

fi

num\_files=$(find $directory -maxdepth 1 -type f | wc -l)

num\_dirs=$(find $directory -maxdepth 1 -type d | wc -l)

echo "Number of files in $directory: $num\_files"

echo "Number of directories in $directory: $((num\_dirs - 1))"

**OUTPUT**



1. **Write a shell script to reverse the list of strings and reverse each string further in the list.**

my\_list=("string1" "string2" "string3" "string4")

my\_list=($(echo "${my\_list[@]}" | tr ' ' '\n' | tac | tr '\n' ' '))

for i in "${!my\_list[@]}"

do

my\_list[$i]=`echo ${my\_list[$i]} | rev`

done

echo "${my\_list[@]}"

**OUTPUT**

****