**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…**

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

**CODE**

**echo "---- EVEN OR ODD IN SHELL SCRIPT -----"**

**echo -n "Enter a number:"**

**read n**

**echo -n "RESULT: "**

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

**then**

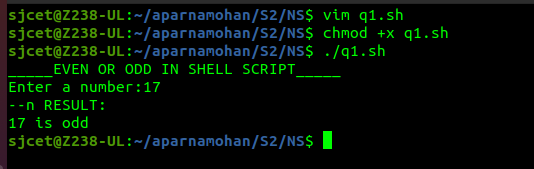
**echo "$n is even"**

**else**

**echo "$n is Odd"**

**fi**

**OUTPUT**

****

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

**CODE**

**echo "LEAP YEAR SHELL SCRIPT "**

**echo -n "Enter a year :"**

**read year\_checker**

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

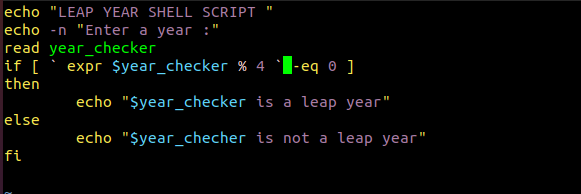
**then**

**echo "$year\_checker is a leap year"**

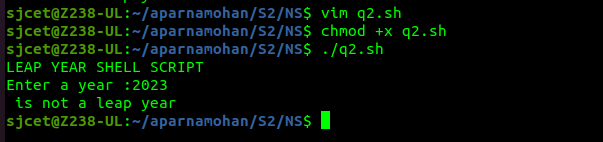
**else**

**echo "$year\_checher is not a leap year"**

**fi**

****

**OUTPUT**

****

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

**CODE**

**echo "Enter the radious of the circle"**

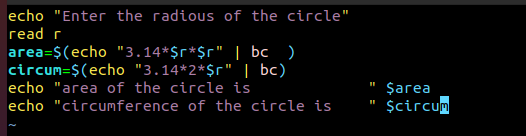
**read r**

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

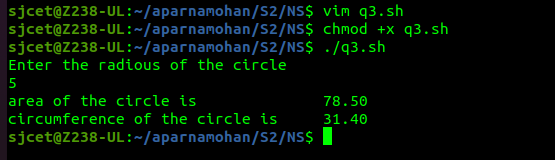
**circum=$(echo "3.14\*2\*$r" | bc)**

**echo "area of the circle is " $area**

**echo "circumference of the circle is " $circum**

****

**OUTPUT**

****

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

**CODE**

**echo enter n**

**read n**

**num=0**

**while [ $n -gt 0 ]**

**do**

**num=$(expr $num \\* 10)**

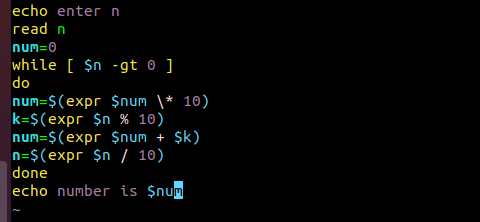
**k=$(expr $n % 10)**

**num=$(expr $num + $k)**

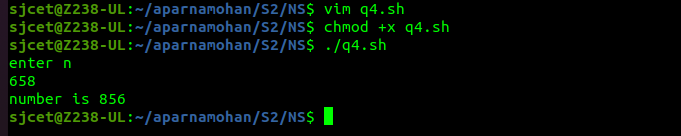
**n=$(expr $n / 10)**

**done**

**echo number is $num**

****

**OUTPUT**

****

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

**CODE**

**echo "input your string without space"**

**read vstr**

**for i in $(seq 0 ${#vstr})**

**do**

**rvstr=${vstr:$i:1}${rvstr}**

**done**

**echo "Input string was :" $vstr**

**echo "After reversng string is :" $rvstr**

**if [ "$vstr" = "$rvstr" ]**

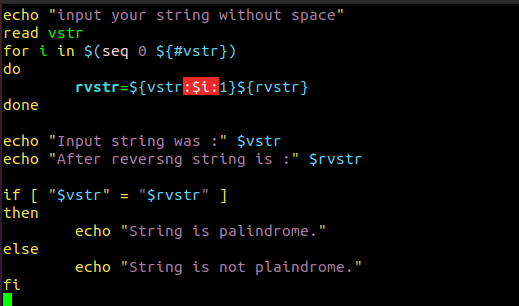
**then**

**echo "String is palindrome."**

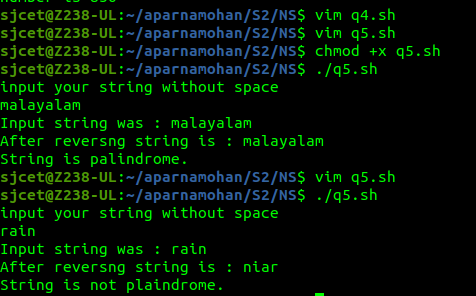
**else**

**echo "String is not plaindrome."**

**fi**

****

**OUTPUT**

****

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

**CODE**

**echo "enter"**

**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**

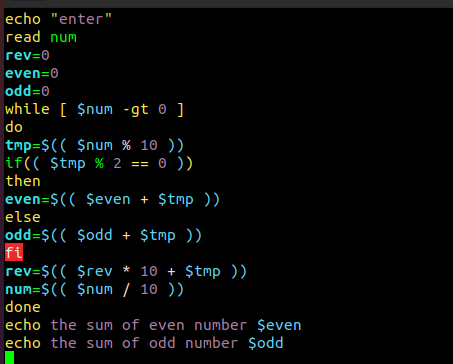
**rev=$(( $rev \* 10 + $tmp ))**

**num=$(( $num / 10 ))**

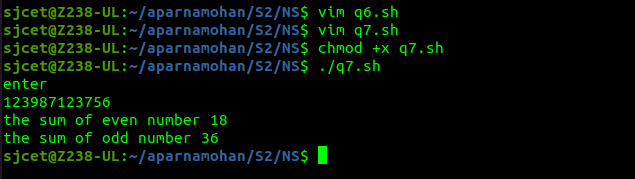
**done**

**echo the sum of even number $even**

**echo the sum of odd number $odd**

****

**OUTPUT**

****

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

**CODE**

**echo "Enter the coefficients of the quadratic equation (a, b, c): "**

**read a b c**

**# Calculate the discriminant**

**discriminant=$((b\*b - 4\*a\*c))**

**# Check if the discriminant is negative (no real roots)**

**if [ $discriminant -lt 0 ]**

**then**

**echo "The quadratic equation has no real roots."**

**else**

**# Calculate the roots**

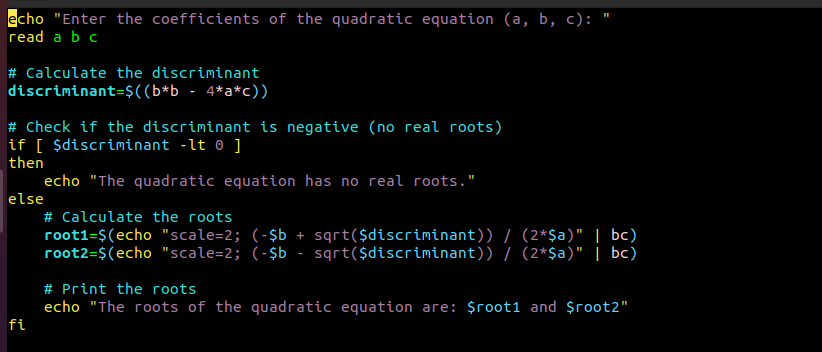
**root1=$(echo "scale=2; (-$b + sqrt($discriminant)) / (2\*$a)" | bc)**

**root2=$(echo "scale=2; (-$b - sqrt($discriminant)) / (2\*$a)" | bc)**

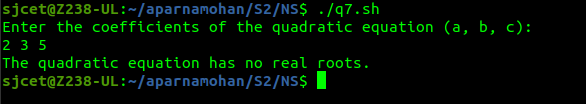
**# Print the roots**

**echo "The roots of the quadratic equation are: $root1 and $root2"**

**fi**

****

**OUTPUT**

****

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

**CODE**

**echo "Enter an integer: "**

**read number**

**# Count the number of digits in the number**

**count=${#number}**

**# Initialize the sum to 0**

**sum=0**

**# Loop through the digits of the number and calculate the sum**

**for (( i=0; i<count; i++ ))**

**do**

**digit=${number:i:1}**

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

**done**

**# Check if the number is an Armstrong number**

**if [ "$sum" -eq "$number" ]**

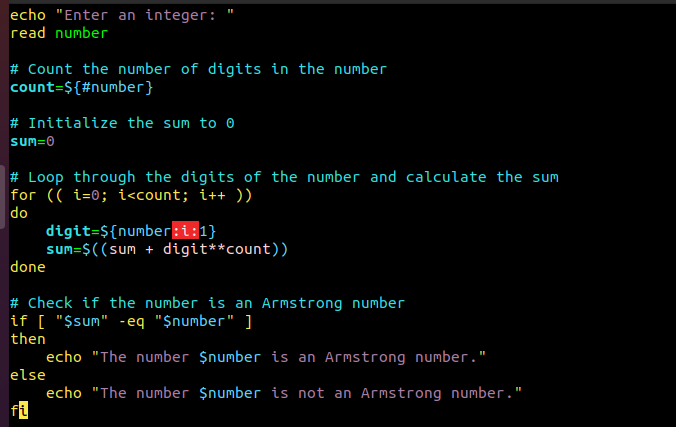
**then**

**echo "The number $number is an Armstrong number."**

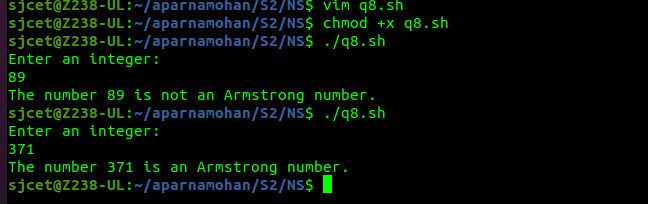
**else**

**echo "The number $number is not an Armstrong number."**

**fi**

****

**OUTPUT**

****

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

**CODE**

**echo "Enter an integer: "**

**read number**

**# Initialize the flag variable to 1**

**flag=1**

**# Check if the number is prime**

**for (( i=2; i<=number/2; i++ ))**

**do**

**if [ $((number%i)) -eq 0 ]**

**then**

**flag=0**

**break**

**fi**

**done**

**# Output the result**

**if [ $number -eq 1 ]**

**then**

**echo "1 is neither prime nor composite."**

**elif [ $flag -eq 1 ]**

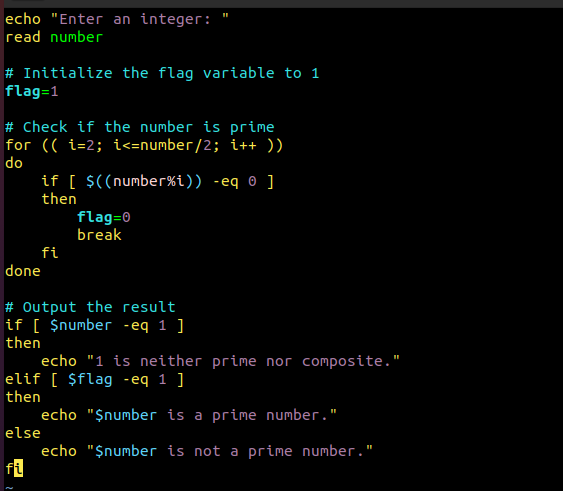
**then**

**echo "$number is a prime number."**

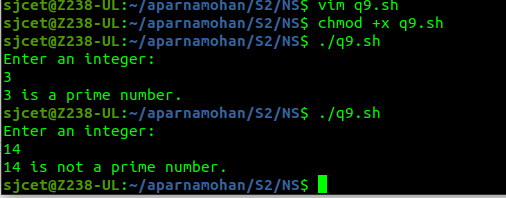
**else**

**echo "$number is not a prime number."**

**fi**

****

**OUTPUT**

****

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

**CODE**

**echo "Prime numbers between 1 and 50 are:"**

**# Check each number between 1 and 50 for primality**

**for (( number=2; number<=50; number++ ))**

**do**

**flag=1**

**for (( i=2; i<=number/2; i++ ))**

**do**

**if [ $((number%i)) -eq 0 ]**

**then**

**flag=0**

**break**

**fi**

**done**

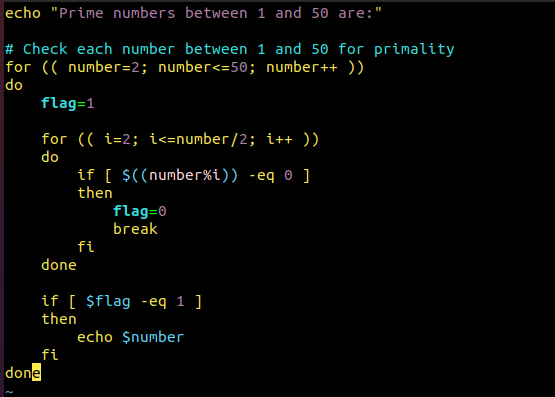
**if [ $flag -eq 1 ]**

**then**

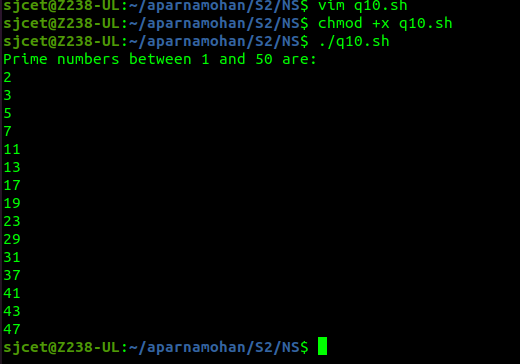
**echo $number**

**fi**

**done**

****

**OUTPUT**

****

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

**CODE**

**echo "Enter a number: "**

**read number**

**# Initialize the sum to 0**

**sum=0**

**# Loop through the digits of the number and calculate the sum of their squares**

**while [ $number -ne 0 ]**

**do**

**digit=$((number % 10))**

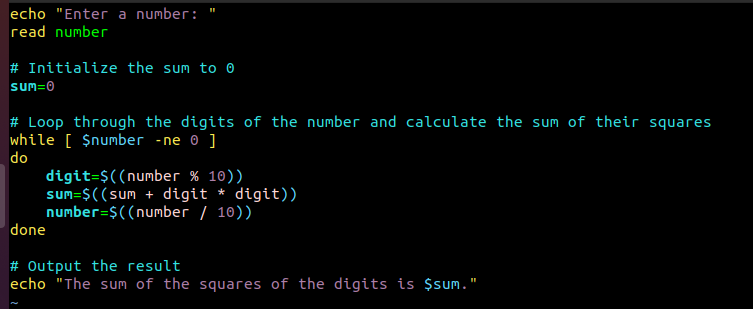
**sum=$((sum + digit \* digit))**

**number=$((number / 10))**

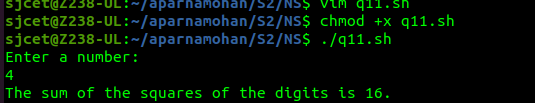
**done**

**# Output the result**

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

****

**OUTPUT**

****

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

**CODE**

**echo "Enter a line of text: "**

**read line**

**# Initialize the vowel count to 0**

**count=0**

**# Loop through each character of the line and check if it is a vowel**

**for (( i=0; i<${#line}; i++ ))**

**do**

**char=${line:$i:1}**

**if [[ $char == [aeiouAEIOU] ]]**

**then**

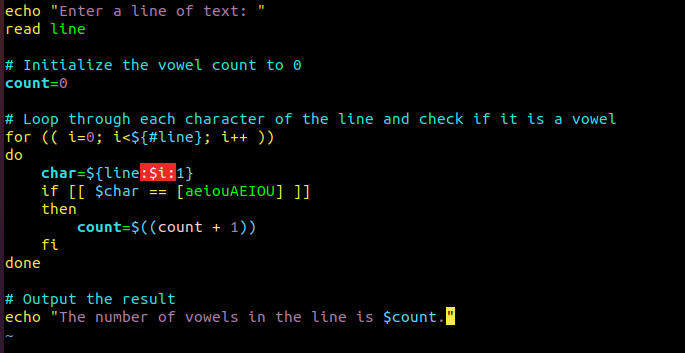
**count=$((count + 1))**

**fi**

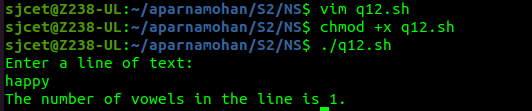
**done**

**# Output the result**

**echo "The number of vowels in the line is $count."**

****

**OUTPUT**

****

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

**CODE**

**declare -A grades=(**

**[Alice]=90**

**[Bob]=80**

**[Charlie]=70**

**[David]=60**

**[Emma]=50**

**)**

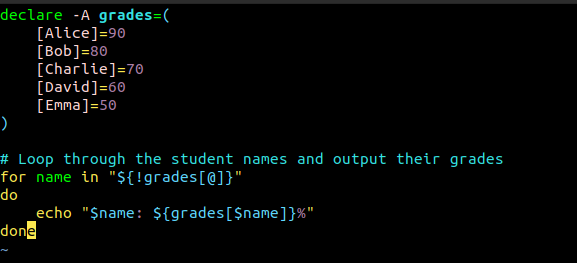
**# Loop through the student names and output their grades**

**for name in "${!grades[@]}"**

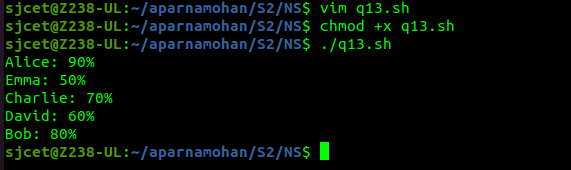
**do**

**echo "$name: ${grades[$name]}%"**

**done**

****

**OUTPUT**

****

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

**CODE**

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

**read numbers**

**# Convert the input string to an array of numbers**

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

**# Initialize the min and max variables to the first number in the array**

**min=${nums[0]}**

**max=${nums[0]}**

**# Loop through the remaining numbers in the array and update min and max as needed**

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

**do**

**if (( num < min )); then**

**min=$num**

**fi**

**if (( num > max )); then**

**max=$num**

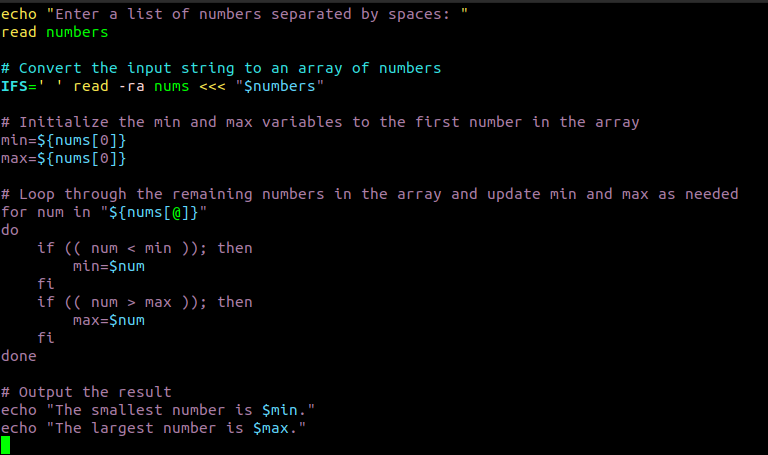
**fi**

**done**

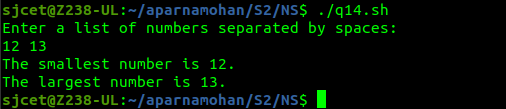
**# Output the result**

**echo "The smallest number is $min."**

**echo "The largest number is $max."**

****

**OUTPUT**

****

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

**CODE**

**echo "Enter a number: "**

**read num**

**# Initialize the min variable to the first digit of the number**

**min=${num:0:1}**

**# Loop through the remaining digits of the number and update min as needed**

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

**do**

**digit=${num:$i:1}**

**if (( digit < min )); then**

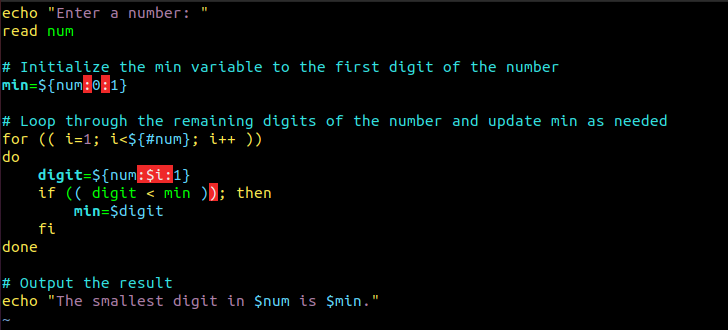
**min=$digit**

**fi**

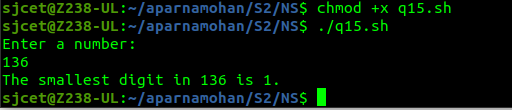
**done**

**# Output the result**

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

****

**OUTPUT**

****

**17. 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.**

**CODE**

**sum=0**

**# Loop through the numbers between 50 and 100**

**for (( num=50; num<=100; num++ ))**

**do**

**# Check if the number is divisible by 3 and not divisible by 5**

**if (( num % 3 == 0 && num % 5 != 0 )); then**

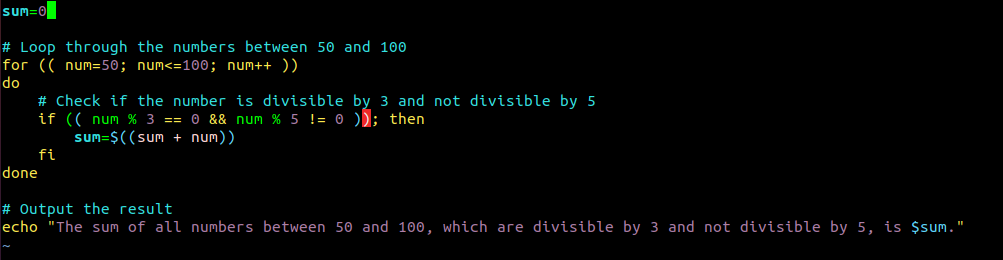
**sum=$((sum + num))**

**fi**

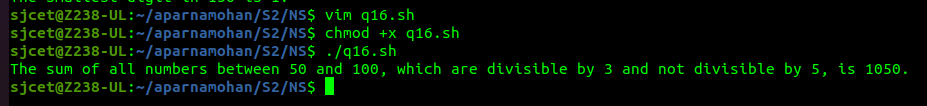
**done**

**# Output the result**

**echo "The sum of all numbers between 50 and 100, which are divisible by 3 and not divisible by 5, is $sum."**

****

**OUTPUT**

****

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

**CODE**

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

**read numbers**

**# Convert the space-separated string to an array**

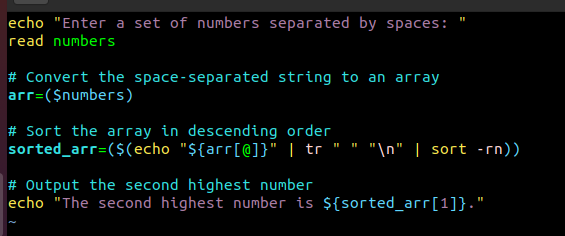
**arr=($numbers)**

**# Sort the array in descending order**

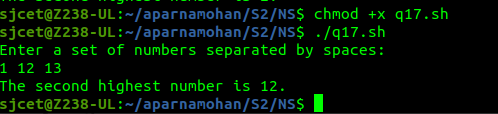
**sorted\_arr=($(echo "${arr[@]}" | tr " " "\n" | sort -rn))**

**# Output the second highest number**

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

****

**OUTPUT**

****

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

**CODE**

**# Define the function to calculate the sum of digits**

**sum\_of\_digits() {**

**num=$1**

**sum=0**

**while [ $num -gt 0 ]**

**do**

**digit=$((num % 10))**

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

**num=$((num / 10))**

**done**

**echo $sum**

**}**

**# Prompt the user to enter a number**

**echo "Enter a number: "**

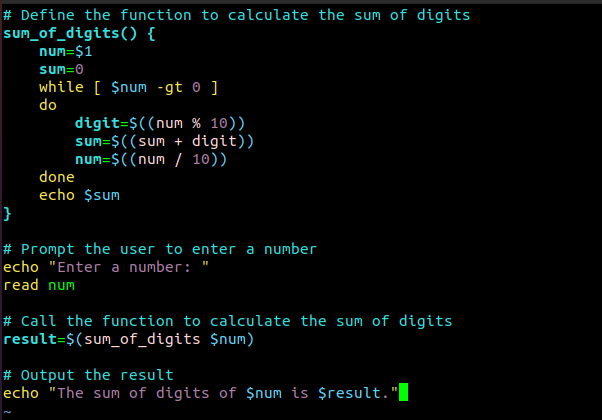
**read num**

**# Call the function to calculate the sum of digits**

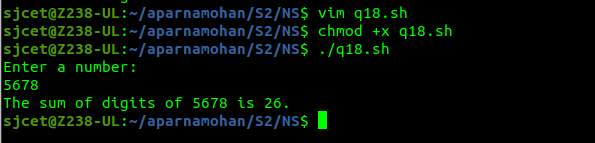
**result=$(sum\_of\_digits $num)**

**# Output the result**

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

****

**OUTPUT**

****

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

**CODE**

**# Define the function to reverse a number**

**reverse\_number() {**

**num=$1**

**rev=0**

**while [ $num -gt 0 ]**

**do**

**digit=$((num % 10))**

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

**num=$((num / 10))**

**done**

**echo $rev**

**}**

**# Prompt the user to enter a number**

**echo "Enter a number: "**

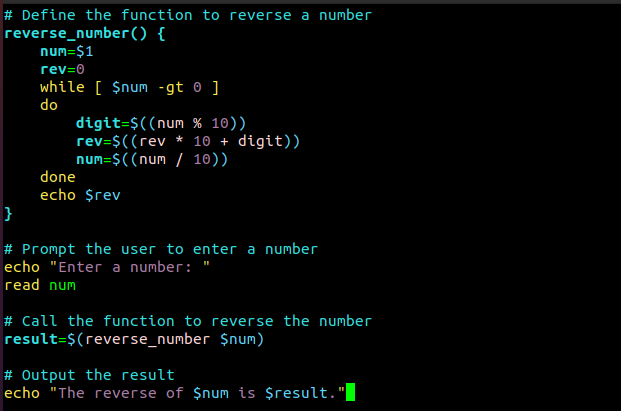
**read num**

**# Call the function to reverse the number**

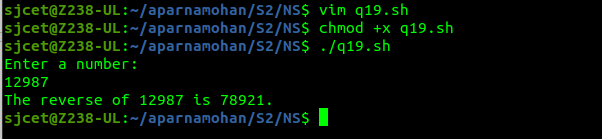
**result=$(reverse\_number $num)**

**# Output the result**

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

****

**OUTPUT**

****

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

**CODE**

**# Prompt the user to enter a number**

**echo "Enter a number: "**

**read num**

**# Initialize the factorial to 1**

**factorial=1**

**# Calculate the factorial using a for loop**

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

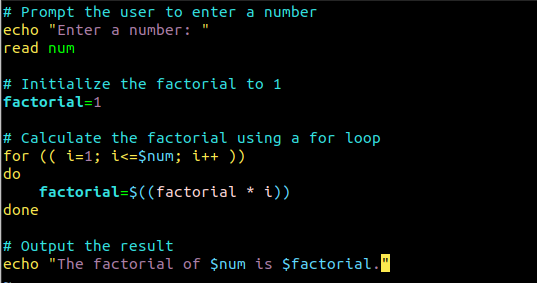
**do**

**factorial=$((factorial \* i))**

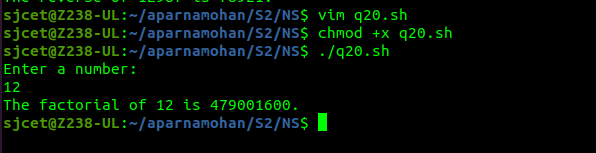
**done**

**# Output the result**

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

****

**OUTPUT**

****

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

**CODE**

**# Prompt the user to enter the number of terms to generate**

**echo "Enter the number of terms to generate: "**

**read num**

**# Initialize the first two terms of the series**

**a=0**

**b=1**

**# Output the first two terms**

**echo -n "$a $b"**

**# Generate the rest of the series using a loop**

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

**do**

**# Calculate the next term**

**c=$((a + b))**

**# Output the next term**

**echo -n " $c"**

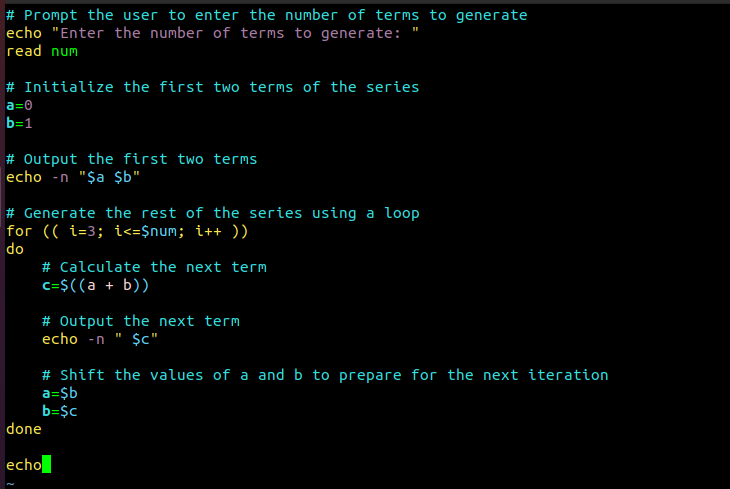
**# Shift the values of a and b to prepare for the next iteration**

**a=$b**

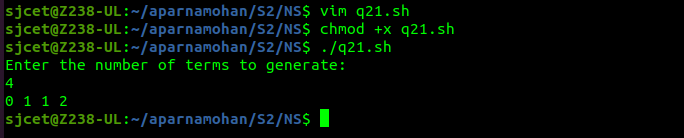
**b=$c**

**done**

**echo**

****

**OUTPUT**

****

**23. 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.**

**CODE**

**if [ $# -ne 2 ]; then**

**echo "Usage: $0 file1 file2"**

**exit 1**

**fi**

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

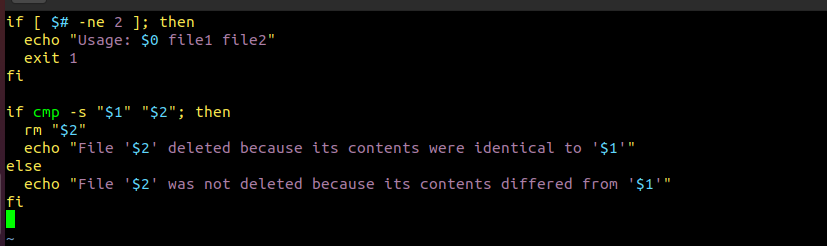
**rm "$2"**

**echo "File '$2' deleted because its contents were identical to '$1'"**

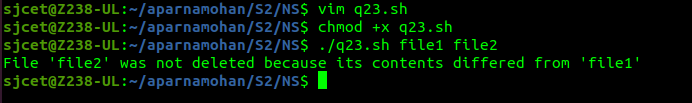
**else**

**echo "File '$2' was not deleted because its contents differed from '$1'"**

**fi**

****

**OUTPUT**

****

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

**CODE**

**#!/bin/bash**

**while true; do**

**clear**

**echo "======================="**

**echo " MAIN MENU "**

**echo "======================="**

**echo "1. List current directory"**

**echo "2. Print working directory"**

**echo "3. Display date"**

**echo "4. Display users logged in"**

**echo "5. Exit"**

**echo -n "Enter your choice: "**

**read choice**

**case $choice in**

**1)**

**ls -la**

**echo "Press enter to continue"**

**read**

**;;**

**2)**

**pwd**

**echo "Press enter to continue"**

**read**

**;;**

**3)**

**date**

**echo "Press enter to continue"**

**read**

**;;**

**4)**

**who**

**echo "Press enter to continue"**

**read**

**;;**

**5)**

**echo "Exiting..."**

**exit 0**

**;;**

**\*)**

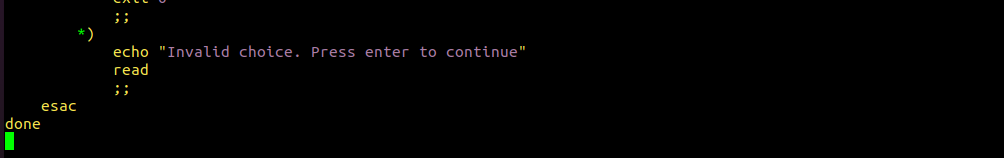
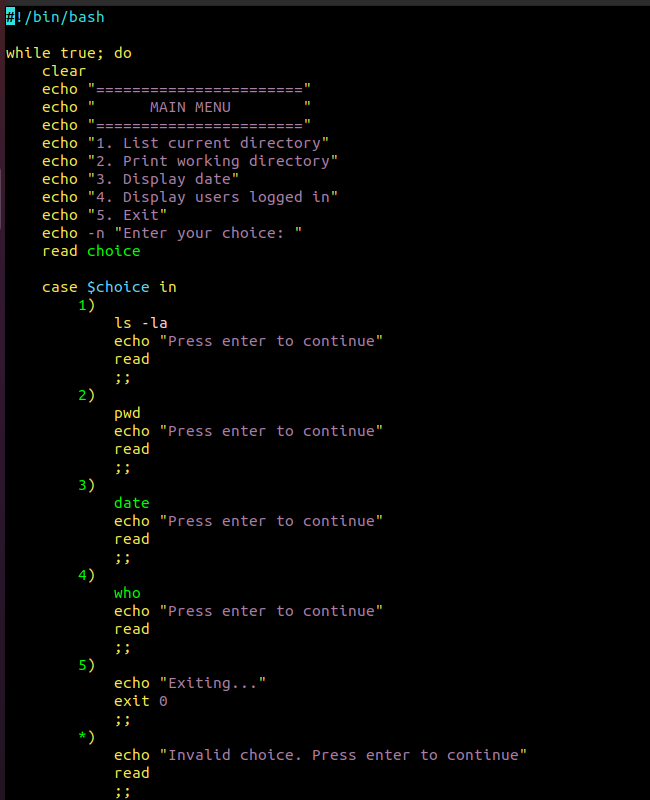
**echo "Invalid choice. Press enter to continue"**

**read**

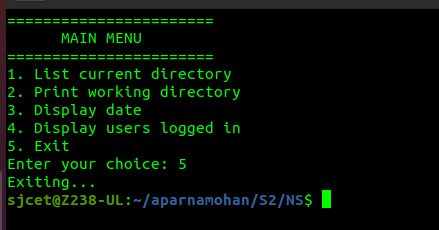
**;;**

**esac**

**done**

****

**OUTPUT**

****

**25.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.**

**CODE**

**#!/bin/bash**

**# Loop through all files in the current directory**

**for file in \*; do**

**# Check if the file is executable**

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

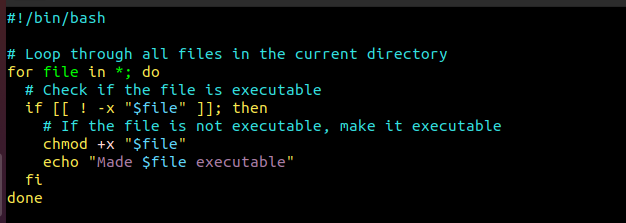
**# If the file is not executable, make it executable**

**chmod +x "$file"**

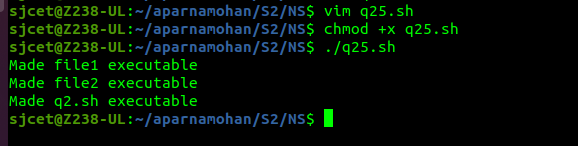
**echo "Made $file executable"**

**fi**

**done**

****

**OUTPUT**

****

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

**CODE**

**#!/bin/bash**

**for i in 1 2 3; do**

**for j in 1 2 3; do**

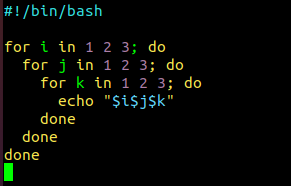
**for k in 1 2 3; do**

**echo "$i$j$k"**

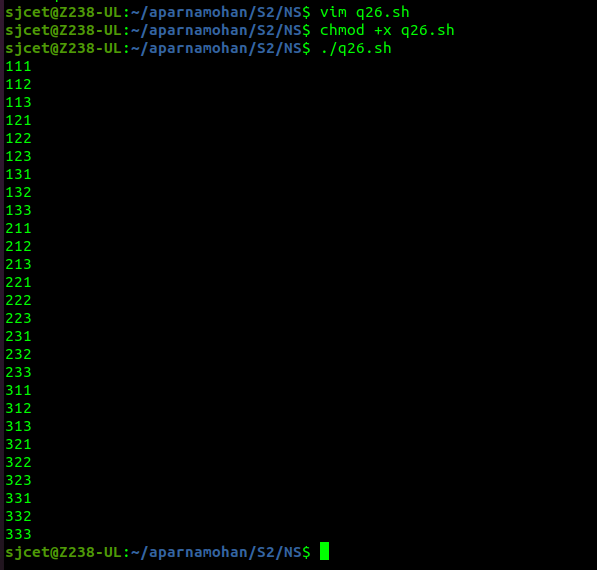
**done**

**done**

**done**

****

**OUTPUT**

****

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

**1**

**2 3**

**4 5 6**

**7\_\_\_ 8 9 10**

**CODE**

**#!/bin/bash**

**rows=4**

**current=1**

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

**do**

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

**do**

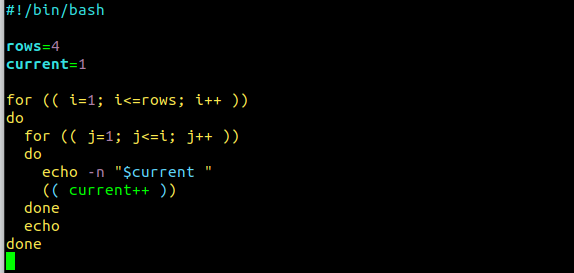
**echo -n "$current "**

**(( current++ ))**

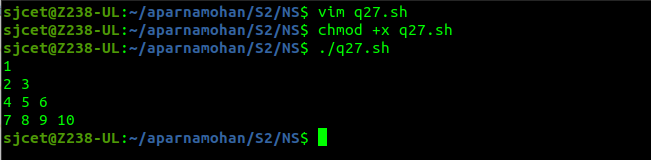
**done**

**echo**

**done**

****

**OUTPUT**

****

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

**CODE**

**#!/bin/bash**

**# Function to calculate the binomial coefficient**

**function binom {**

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

**echo 1**

**else**

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

**fi**

**}**

**# Get the number of rows from the user**

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

**read rows**

**# Loop through each row**

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

**# Loop through each element in the row**

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

**# Calculate the binomial coefficient and print**

**val=$(binom $i $j)**

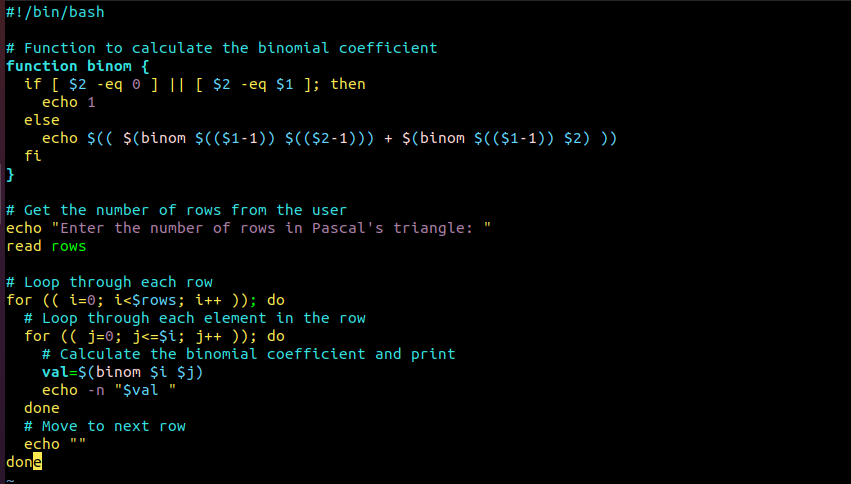
**echo -n "$val "**

**done**

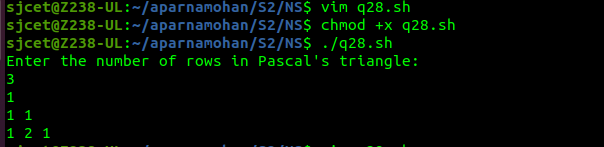
**# Move to next row**

**echo ""**

**done**

****

**OUTPUT**

****

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

**CODE**

**#!/bin/bash**

**# Prompt user for decimal input**

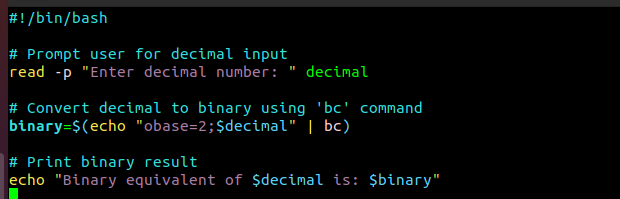
**read -p "Enter decimal number: " decimal**

**# Convert decimal to binary using 'bc' command**

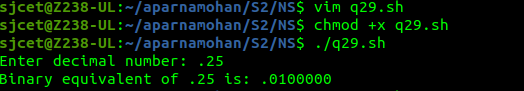
**binary=$(echo "obase=2;$decimal" | bc)**

**# Print binary result**

**echo "Binary equivalent of $decimal is: $binary"**

****

**OUTPUT**

****

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

**CODE**

**#!/bin/bash**

**echo "Enter a string: "**

**read string**

**# Reverse the string**

**reverse=$(echo $string | rev)**

**# Check if the string is equal to its reverse**

**if [ "$string" == "$reverse" ]**

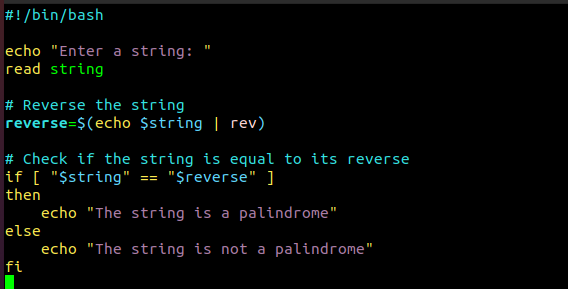
**then**

**echo "The string is a palindrome"**

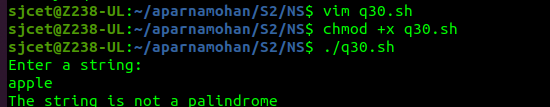
**else**

**echo "The string is not a palindrome"**

**fi**

****

**OUTPUT**

****

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

**CODE**

**#!/bin/bash**

**# Prompt the user for the file name**

**echo "Enter the file name: "**

**read file**

**# Check if the file exists**

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

**echo "File not found."**

**exit 1**

**fi**

**# Convert the contents of the file to lowercase and replace all non-alphanumeric characters with spaces**

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

**# Create an array of words from the file contents**

**words=($contents)**

**# Loop through the array of words and count their occurrences**

**declare -A count**

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

**if [ -n "$word" ]; then**

**((count[$word]++))**

**fi**

**done**

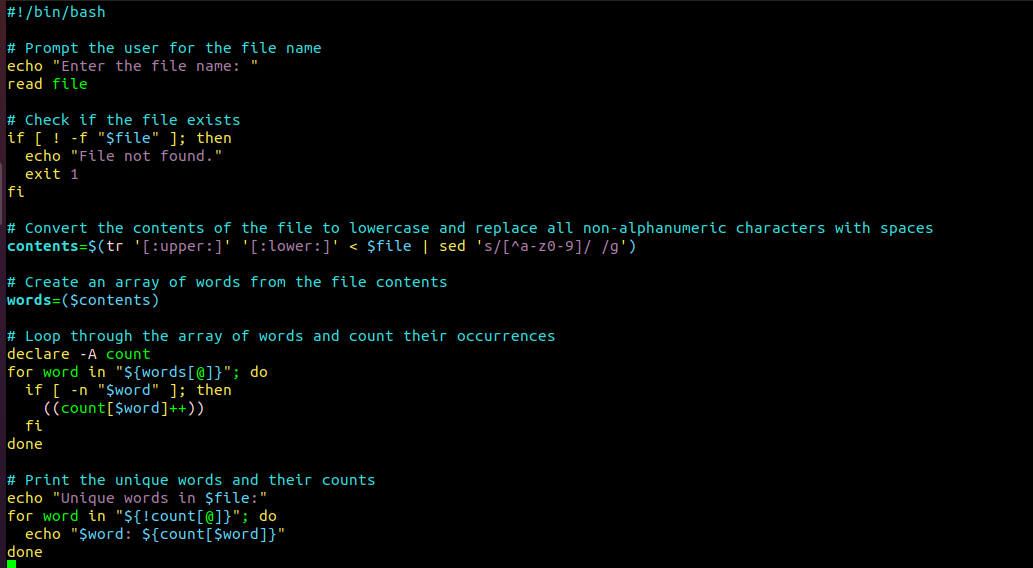
**# Print the unique words and their counts**

**echo "Unique words in $file:"**

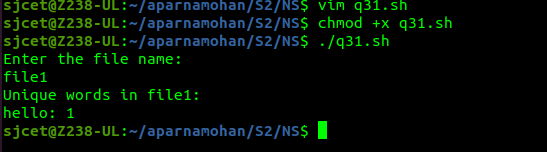
**for word in "${!count[@]}"; do**

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

**done**

****

**OUTPUT**

****

**32. 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.**

**CODE**

**#!/bin/bash**

**# Set the search directory**

**search\_dir="."**

**# Find all ".txt" files in the search directory and its subdirectories**

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

**# Initialize the count**

**count=0**

**# Loop through each file and count the occurrences of "Linux"**

**for file in $files; do**

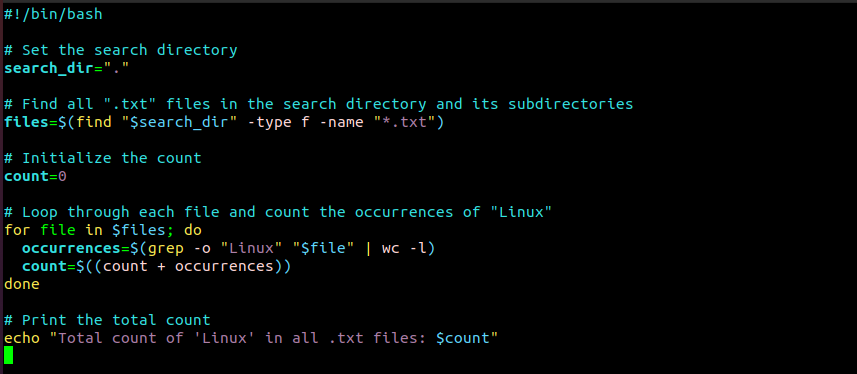
**occurrences=$(grep -o "Linux" "$file" | wc -l)**

**count=$((count + occurrences))**

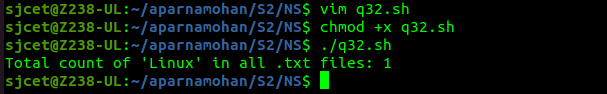
**done**

**# Print the total count**

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

****

**OUTPUT**

****

**33. 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.**

**CODE**

**#!/bin/bash**

**read -p "Enter your password: " password**

**# Check if password is at least 8 characters long**

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

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

**exit 1**

**fi**

**# Check if password contains both alphabet and number**

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

**echo "Password must contain both alphabet and number."**

**exit 1**

**fi**

**# Check if password includes both small and capital case letters**

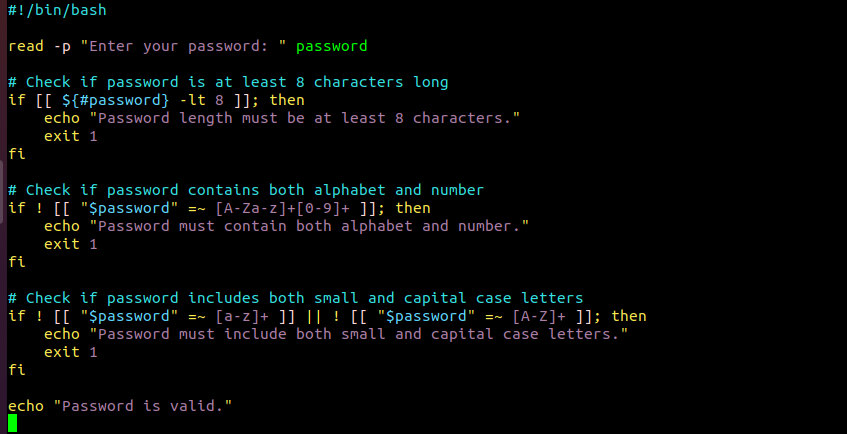
**if ! [[ "$password" =~ [a-z]+ ]] || ! [[ "$password" =~ [A-Z]+ ]]; then**

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

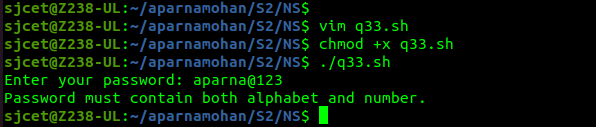
**exit 1**

**fi**

**echo "Password is valid."**

****

**OUTPUT**

****

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

**CODE**

**echo "Enter directory path: "**

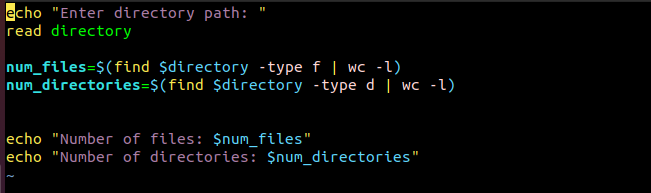
**read directory**

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

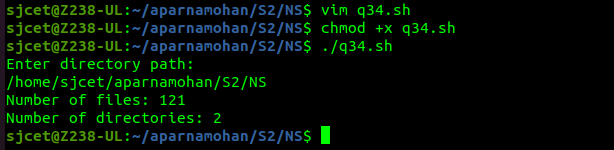
**num\_directories=$(find $directory -type d | wc -l)**

**echo "Number of files: $num\_files"**

**echo "Number of directories: $num\_directories"**

****

**OUTPUT**

****

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

**CODE**

**#!/bin/bash**

**# Define a list of strings**

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

**# Reverse the order of the list**

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

**# Reverse each string in the list**

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

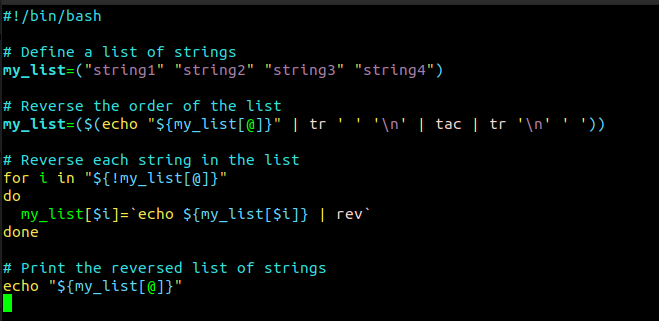
**do**

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

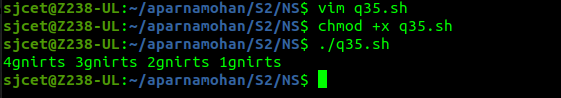
**done**

**# Print the reversed list of strings**

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

****

**OUTPUT**

****