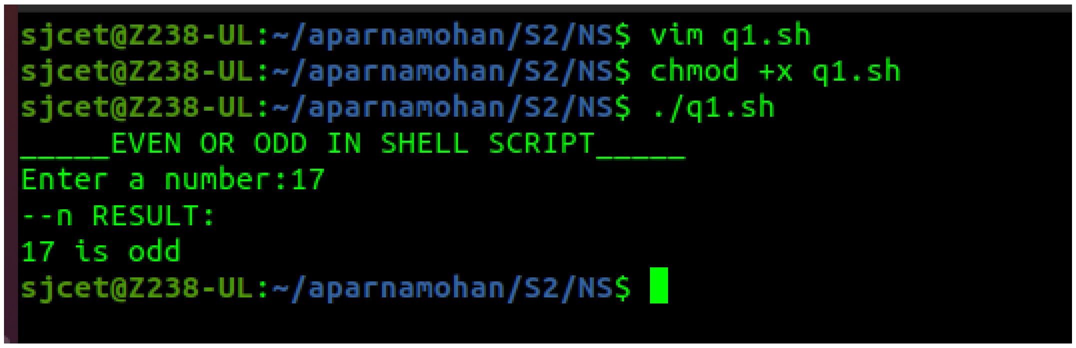
1. Practice Basic Shell Commands like:- ls, cd, du, pwd, man, cat, more, less, head, tail, mkdir,cp, mv, rm, touch, grep, sort, wc, cut, echo…

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

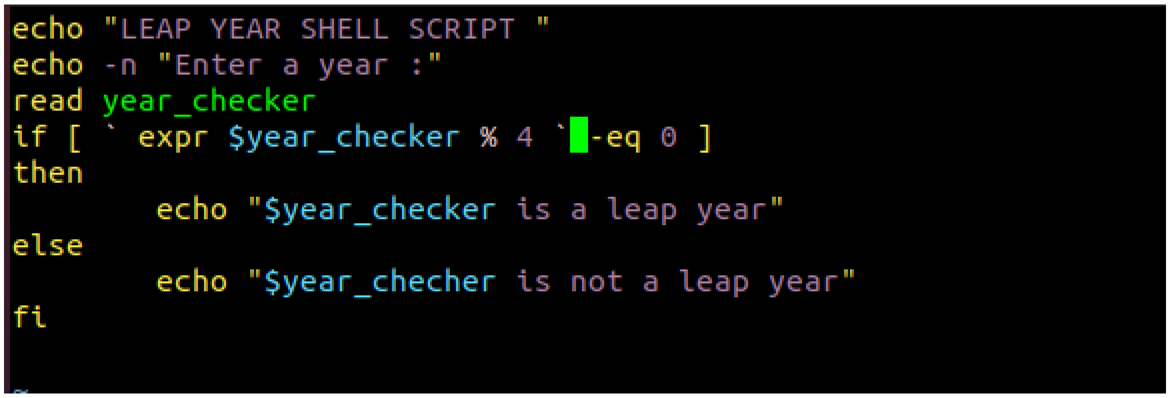


```
sjcet@Z238-UL:~/aparnamohan/S2/NS$ vim q1.sh
sjcet@Z238-UL:~/aparnamohan/S2/NS$ chmod +x q1.sh
sjcet@Z238-UL:~/aparnamohan/S2/NS$ ./q1.sh
_____EVEN OR ODD IN SHELL SCRIPT_____
Enter a number:17
--n RESULT:
17 is odd
sjcet@Z238-UL:~/aparnamohan/S2/NS$
```

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

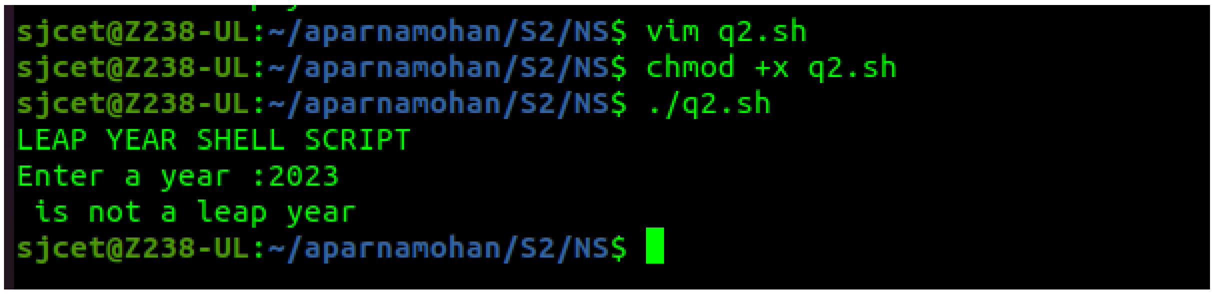
CODE

```
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
```



```
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

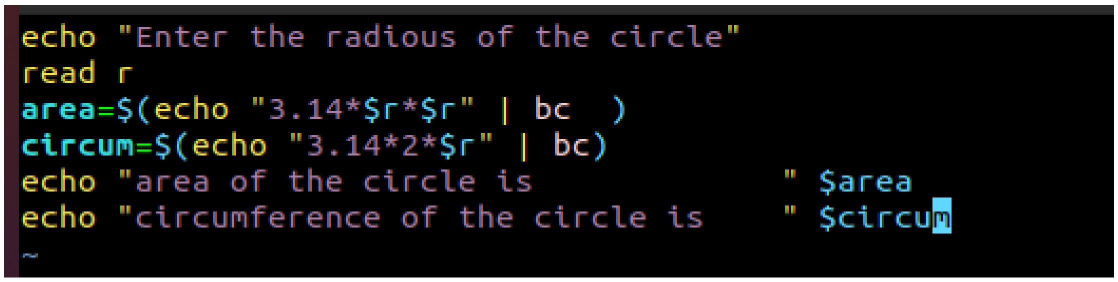


```
sjcet@Z238-UL:~/aparnamohan/S2/NS$ vim q2.sh
sjcet@Z238-UL:~/aparnamohan/S2/NS$ chmod +x q2.sh
sjcet@Z238-UL:~/aparnamohan/S2/NS$ ./q2.sh
LEAP YEAR SHELL SCRIPT
Enter a year :2023
 is not a leap year
sjcet@Z238-UL:~/aparnamohan/S2/NS$
```

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

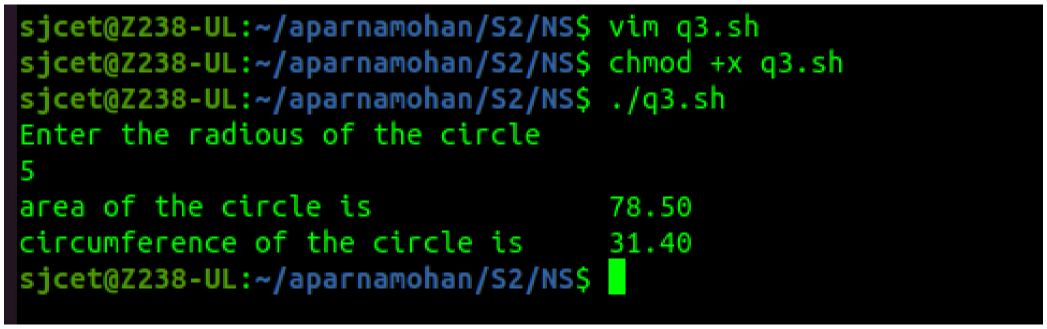
CODE

```
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
```



```
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



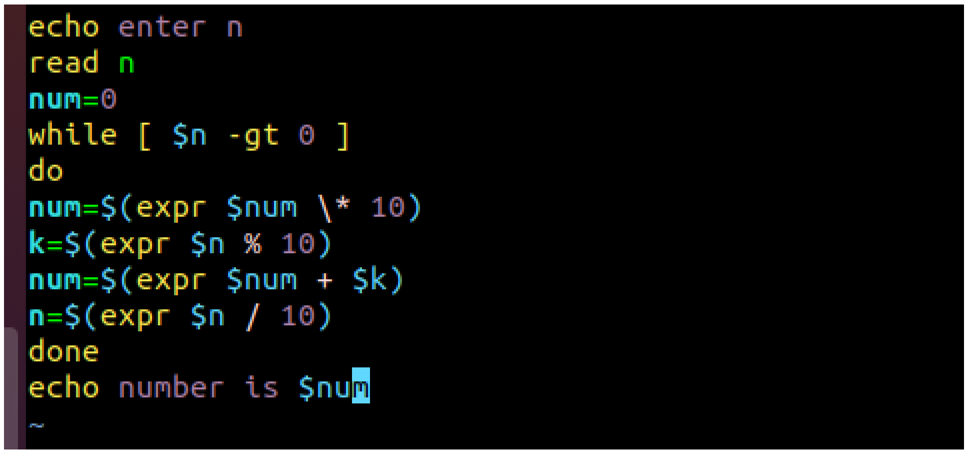
```
sjcet@Z238-UL:~/aparnamohan/S2/NS$ vim q3.sh
sjcet@Z238-UL:~/aparnamohan/S2/NS$ chmod +x q3.sh
sjcet@Z238-UL:~/aparnamohan/S2/NS$ ./q3.sh
Enter the radious of the circle
5
area of the circle is              78.50
circumference of the circle is     31.40
sjcet@Z238-UL:~/aparnamohan/S2/NS$ 
```

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

CODE

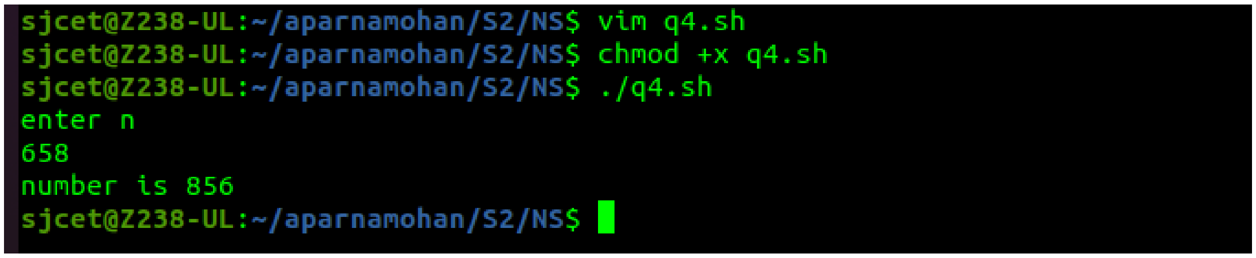
```
echo enter n
read n
num=0
while [ $n -gt 0 ]
do
num=$(expr $num \* 10)
k=$(expr $n % 10)
num=$(expr $num + $k)
n=$(expr $n / 10)
done
```

echo number is $num



```
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

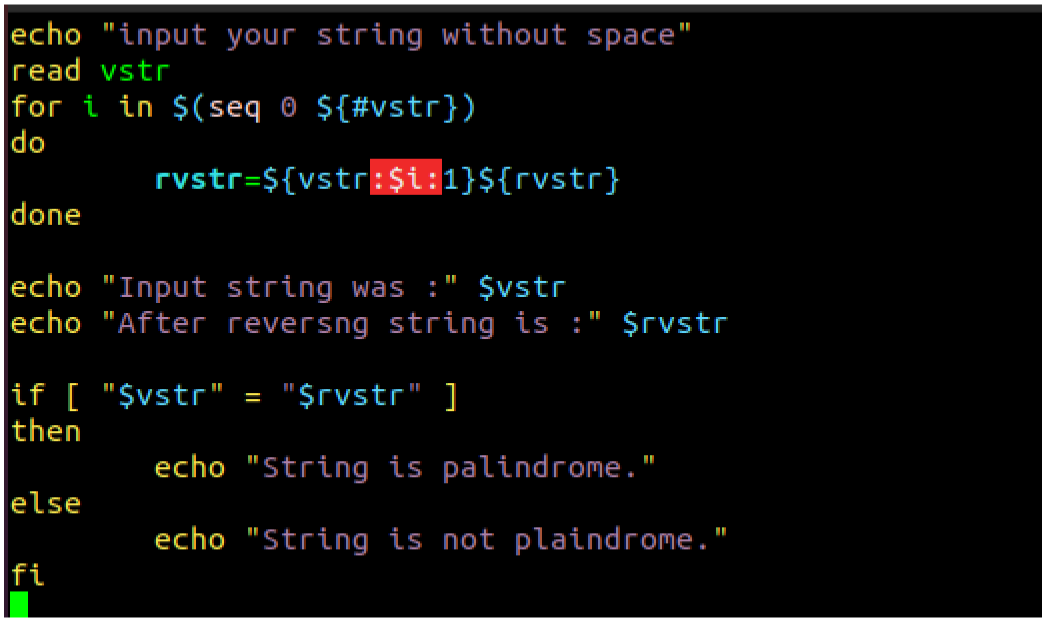


```
sjcet@Z238-UL:~/aparnamohan/S2/NS$ vim q4.sh
sjcet@Z238-UL:~/aparnamohan/S2/NS$ chmod +x q4.sh
sjcet@Z238-UL:~/aparnamohan/S2/NS$ ./q4.sh
enter n
658
number is 856
sjcet@Z238-UL:~/aparnamohan/S2/NS$
```

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

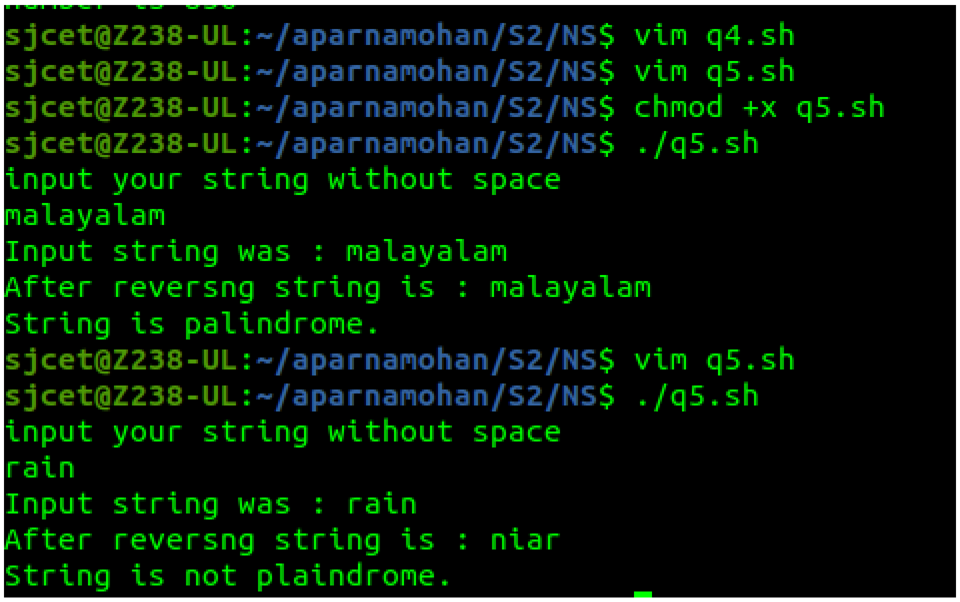
CODE

```
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
```



```
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



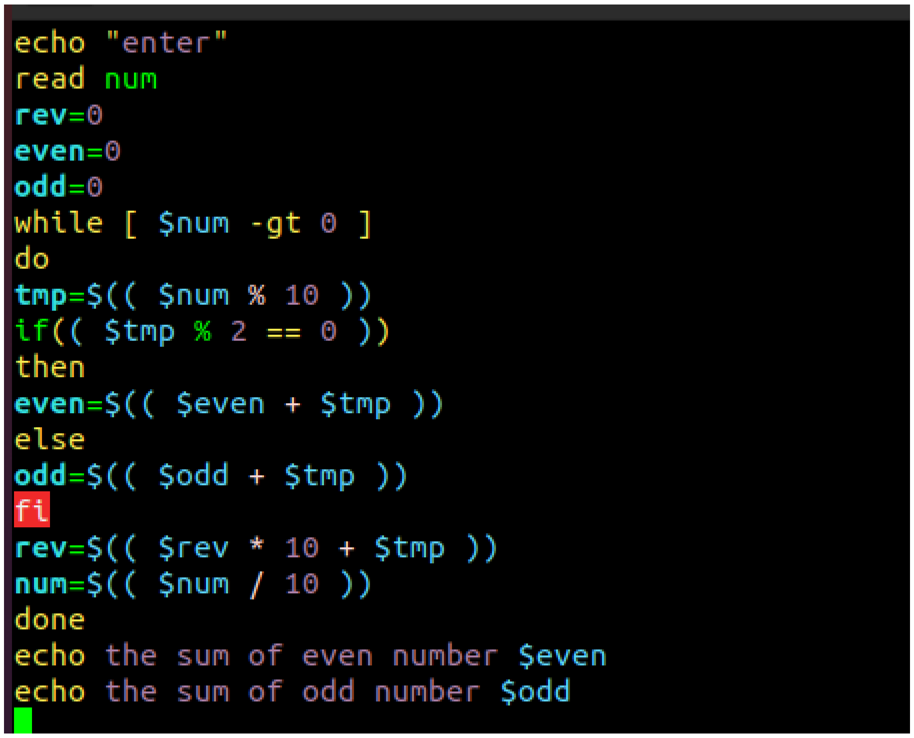
```
sjcet@Z238-UL:~/aparnamohan/S2/NS$ vim q4.sh
sjcet@Z238-UL:~/aparnamohan/S2/NS$ vim q5.sh
sjcet@Z238-UL:~/aparnamohan/S2/NS$ chmod +x q5.sh
sjcet@Z238-UL:~/aparnamohan/S2/NS$ ./q5.sh
input your string without space
malayalam
Input string was : malayalam
After reversng string is : malayalam
String is palindrome.
sjcet@Z238-UL:~/aparnamohan/S2/NS$ vim q5.sh
sjcet@Z238-UL:~/aparnamohan/S2/NS$ ./q5.sh
input your string without space
rain
Input string was : rain
After reversng string is : niar
String is not plaindrome.
```

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

CODE

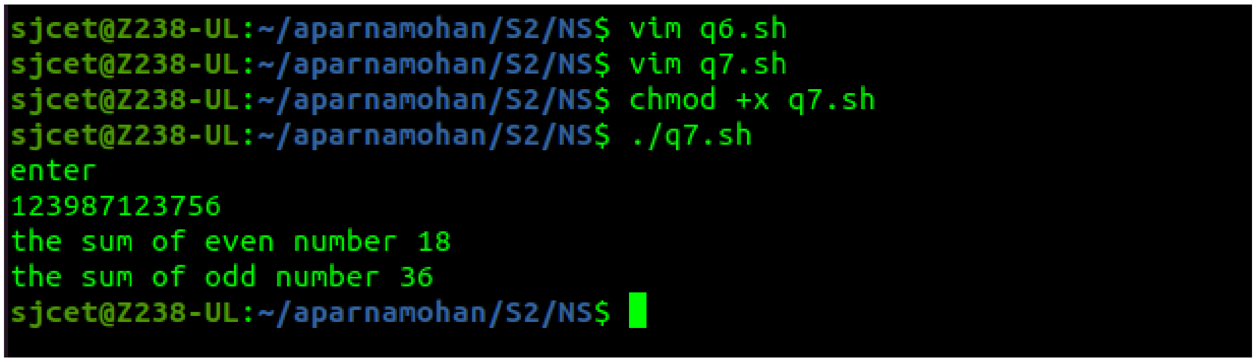
```
echo "enter"
read num
rev=0
even=0
odd=0
while [ $num -gt 0 ]
do
tmp=$(( $num % 10 ))
if(( $tmp % 2 == 0 ))
then
even=$(( $even + $tmp ))
else
odd=$(( $odd + $tmp ))
fi
rev=$(( $rev * 10 + $tmp ))
num=$(( $num / 10 ))
```

done
echo the sum of even number $even
echo the sum of odd number $odd



```
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

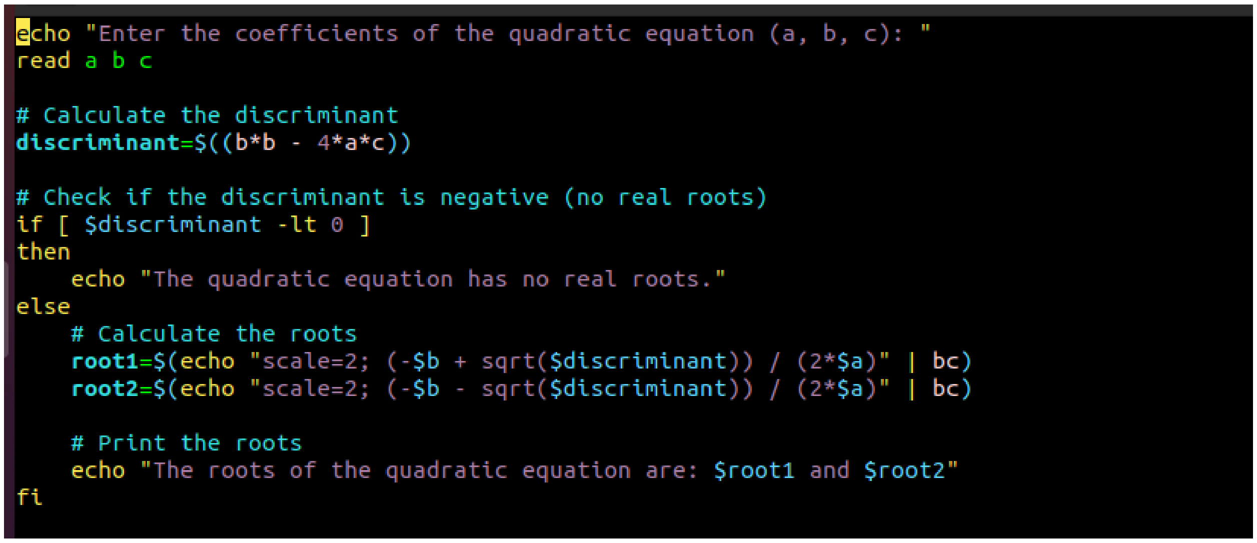


```
sjcet@Z238-UL:~/aparnamohan/S2/NS$ vim q6.sh
sjcet@Z238-UL:~/aparnamohan/S2/NS$ vim q7.sh
sjcet@Z238-UL:~/aparnamohan/S2/NS$ chmod +x q7.sh
sjcet@Z238-UL:~/aparnamohan/S2/NS$ ./q7.sh
enter
123987123756
the sum of even number 18
the sum of odd number 36
sjcet@Z238-UL:~/aparnamohan/S2/NS$
```

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

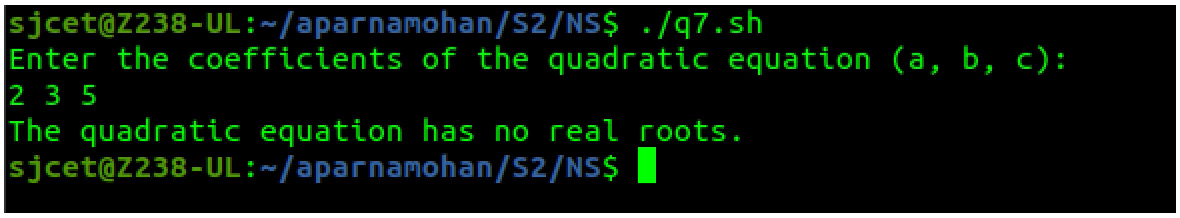
CODE

```
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
```



```
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

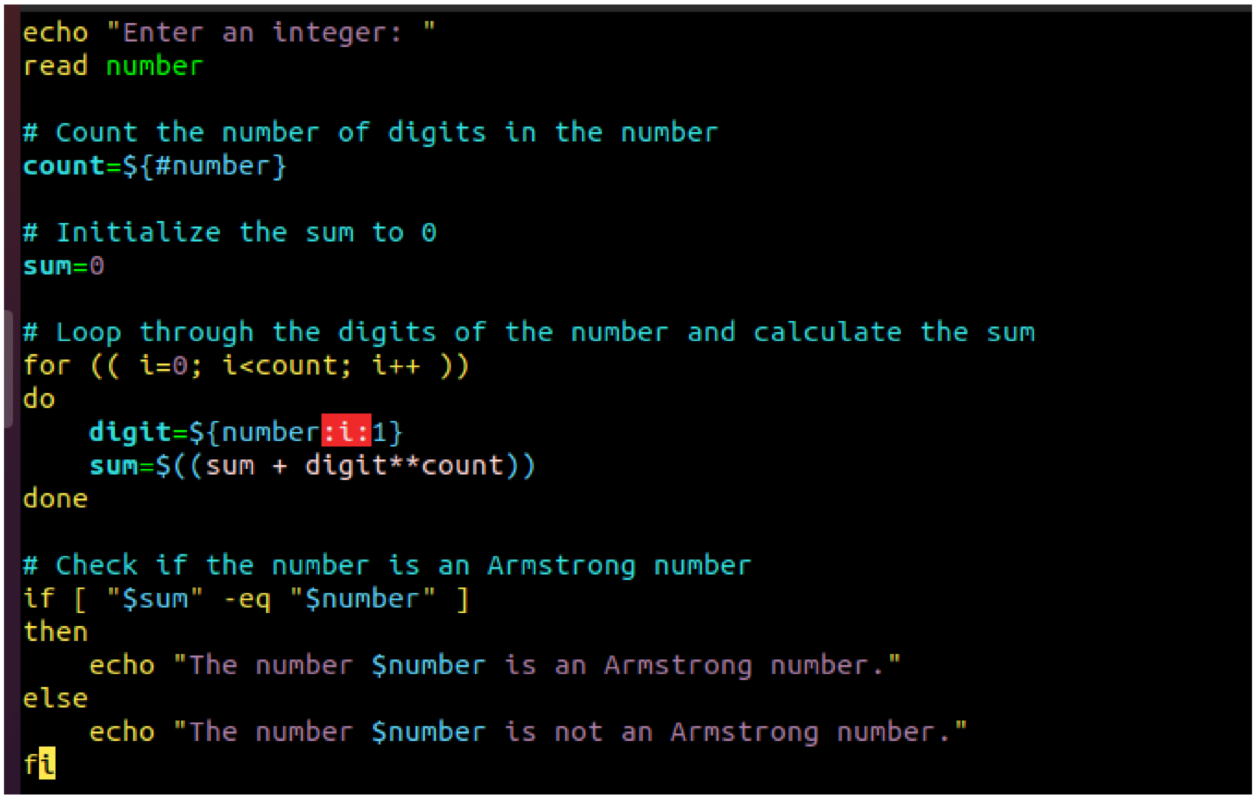


```
sjcet@Z238-UL:~/aparnamohan/S2/NS$ ./q7.sh
Enter the coefficients of the quadratic equation (a, b, c):
2 3 5
The quadratic equation has no real roots.
sjcet@Z238-UL:~/aparnamohan/S2/NS$ 
```

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

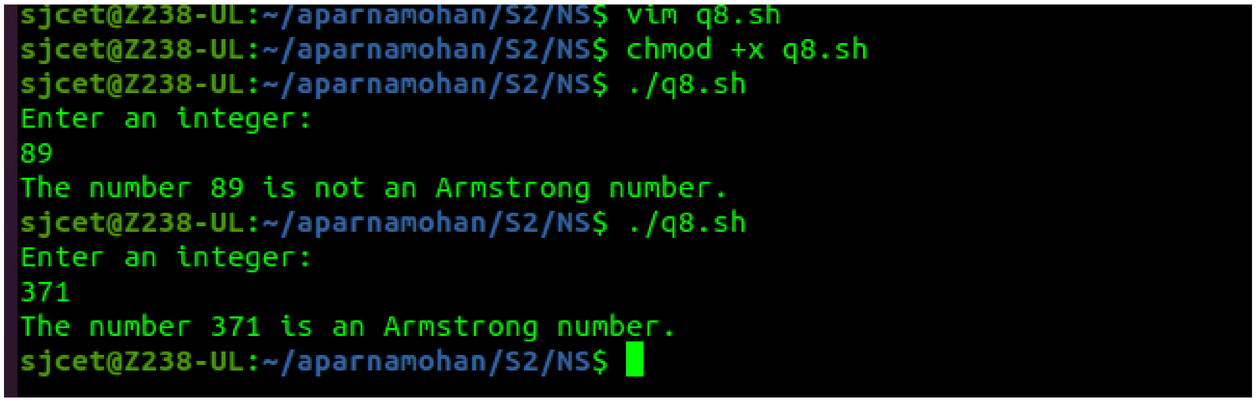
CODE

```
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
```



```
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



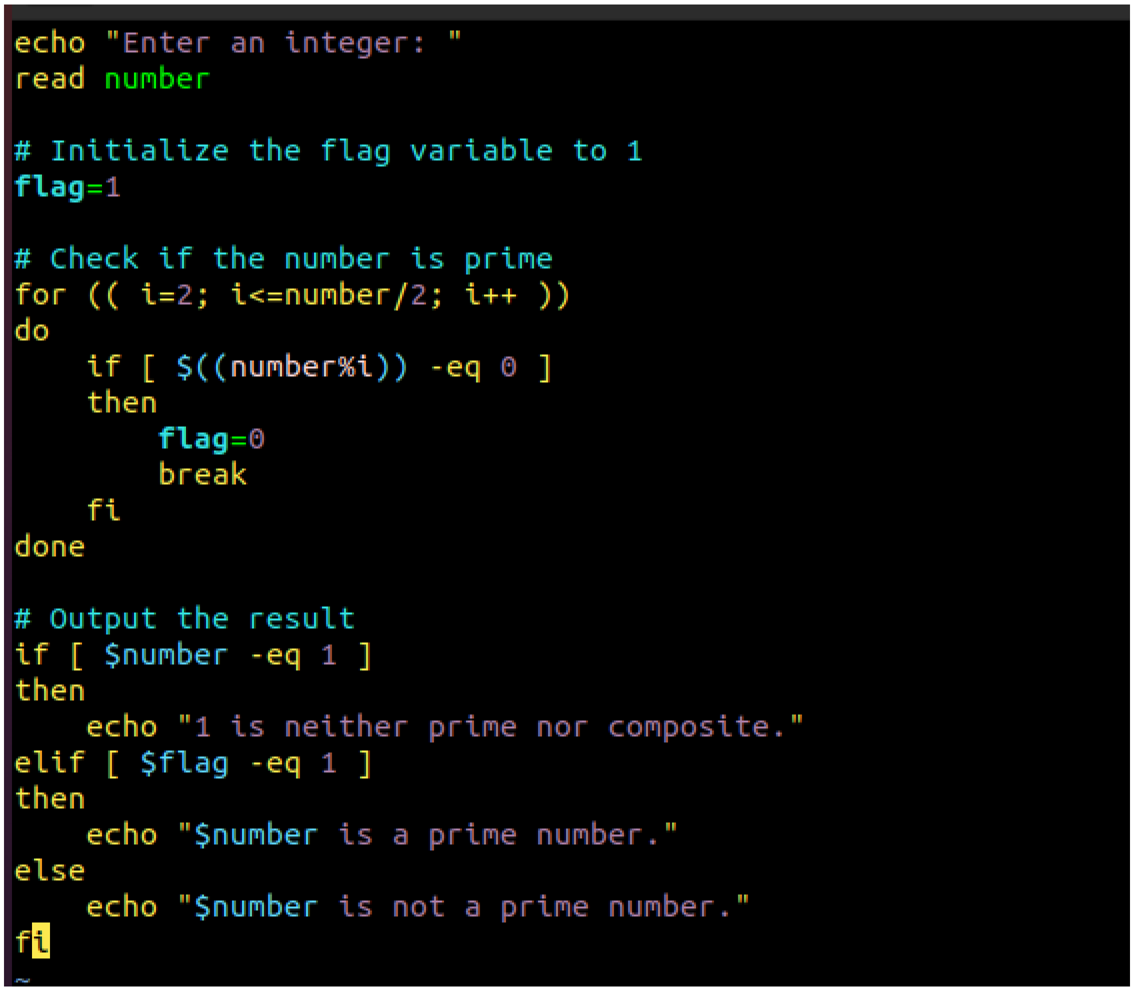
```
sjcet@Z238-UL:~/aparnamohan/S2/NS$ vim q8.sh
sjcet@Z238-UL:~/aparnamohan/S2/NS$ chmod +x q8.sh
sjcet@Z238-UL:~/aparnamohan/S2/NS$ ./q8.sh
Enter an integer:
89
The number 89 is not an Armstrong number.
sjcet@Z238-UL:~/aparnamohan/S2/NS$ ./q8.sh
Enter an integer:
371
The number 371 is an Armstrong number.
sjcet@Z238-UL:~/aparnamohan/S2/NS$
```

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

CODE

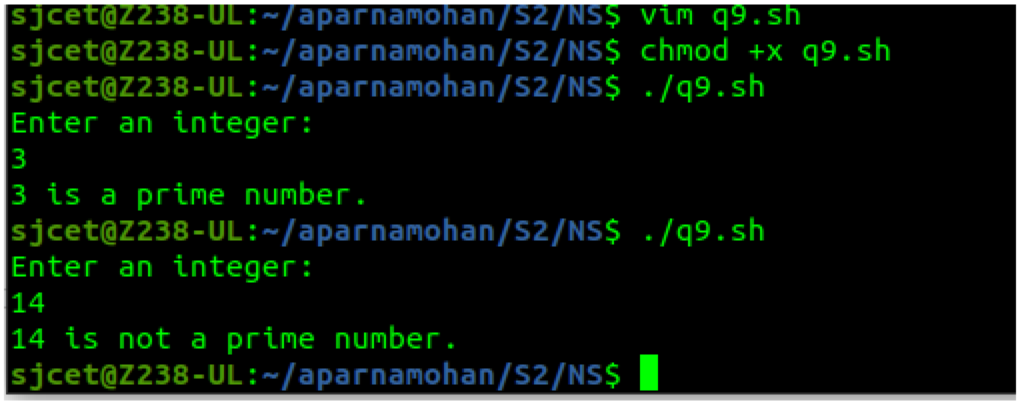
echo "Enter an integer: "
read number

```
# 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
```



```
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

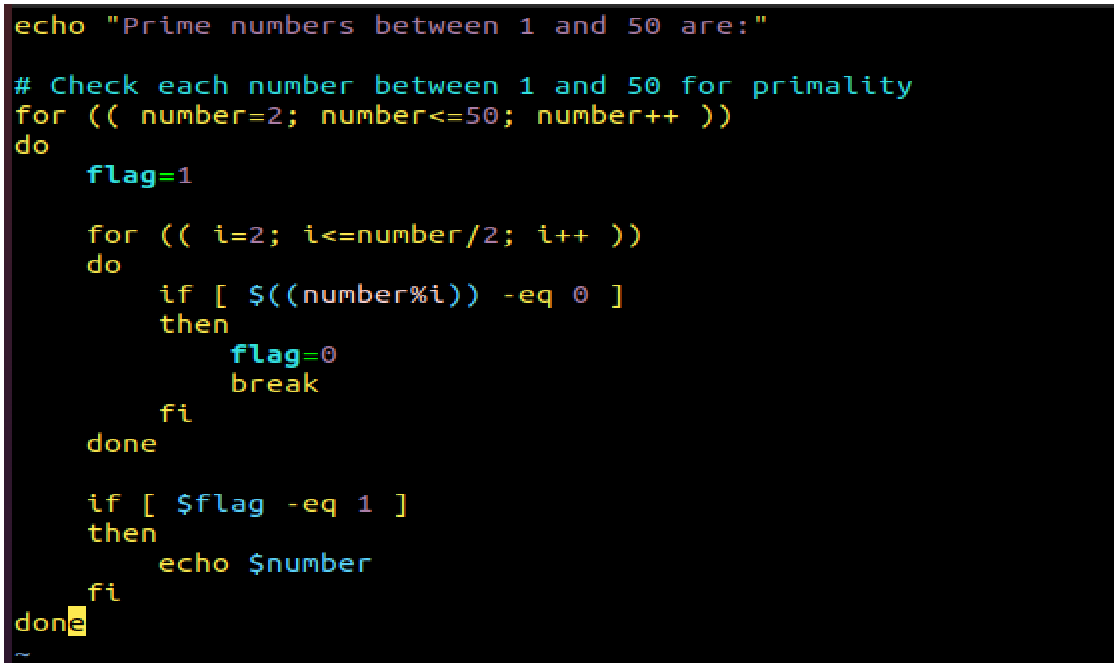


```
sjcet@Z238-UL:~/aparnamohan/S2/NS$ vim q9.sh
sjcet@Z238-UL:~/aparnamohan/S2/NS$ chmod +x q9.sh
sjcet@Z238-UL:~/aparnamohan/S2/NS$ ./q9.sh
Enter an integer:
3
3 is a prime number.
sjcet@Z238-UL:~/aparnamohan/S2/NS$ ./q9.sh
Enter an integer:
14
14 is not a prime number.
sjcet@Z238-UL:~/aparnamohan/S2/NS$ 
```

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

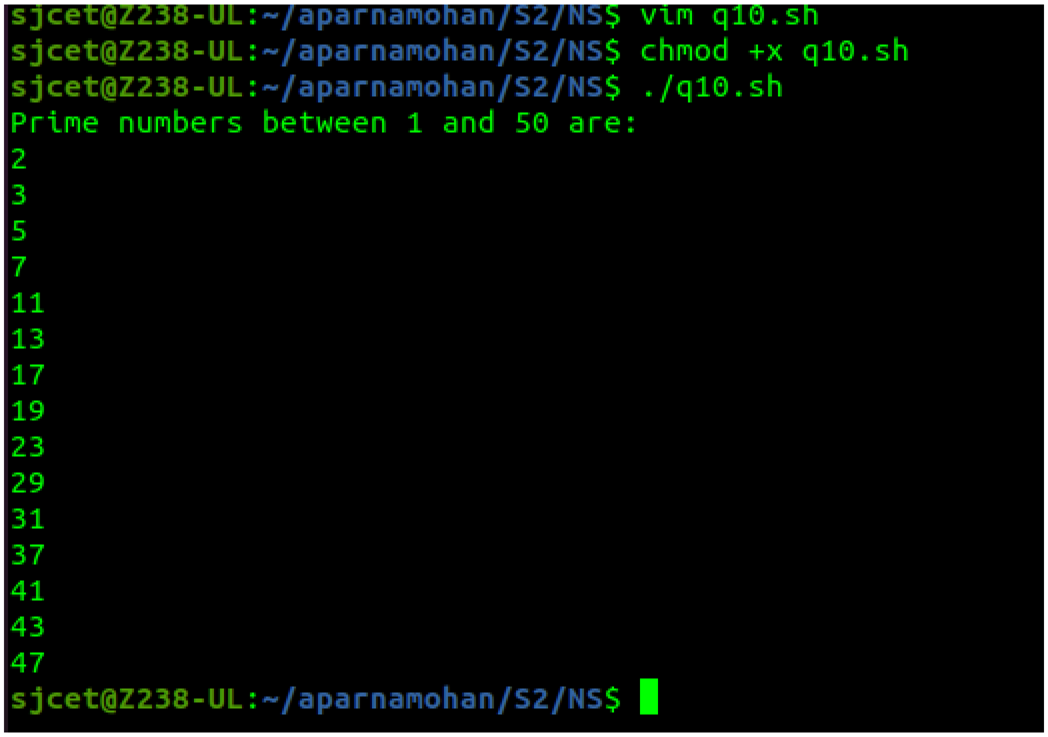
CODE

```
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
```



```
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

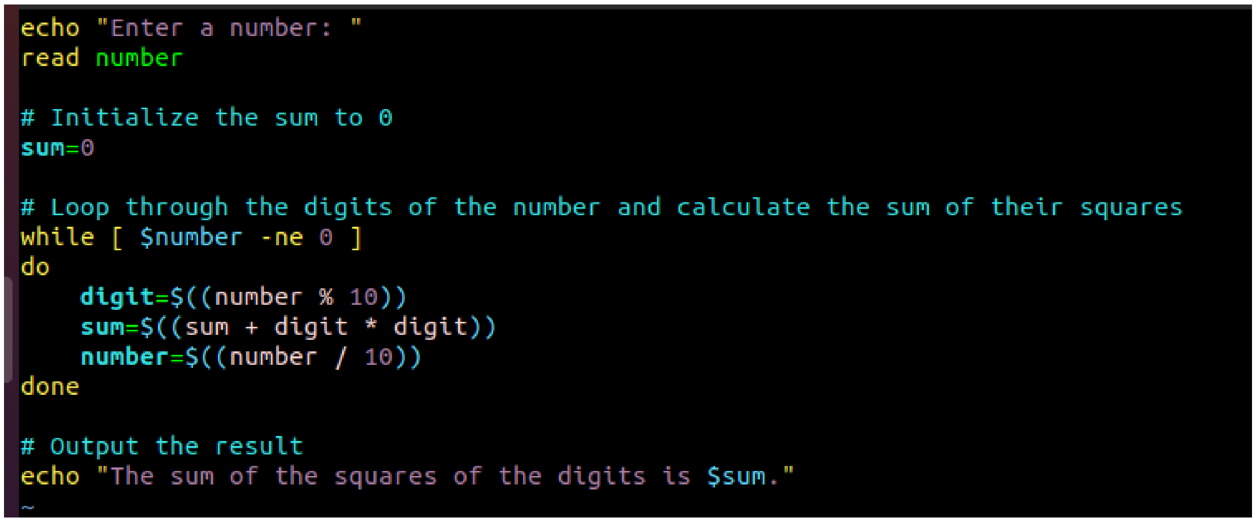


```
sjcet@Z238-UL:~/aparnamohan/S2/NS$ vim q10.sh
sjcet@Z238-UL:~/aparnamohan/S2/NS$ chmod +x q10.sh
sjcet@Z238-UL:~/aparnamohan/S2/NS$ ./q10.sh
Prime numbers between 1 and 50 are:
2
3
5
7
11
13
17
19
23
29
31
37
41
43
47
sjcet@Z238-UL:~/aparnamohan/S2/NS$ 
```

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

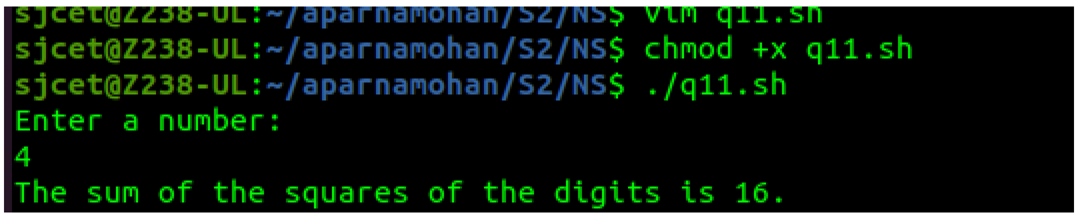
CODE

```
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."
```



```
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



```
sjcet@Z238-UL:~/aparnamohan/S2/NS$ vim q11.sh
sjcet@Z238-UL:~/aparnamohan/S2/NS$ chmod +x q11.sh
sjcet@Z238-UL:~/aparnamohan/S2/NS$ ./q11.sh
Enter a number:
4
The sum of the squares of the digits is 16.
```

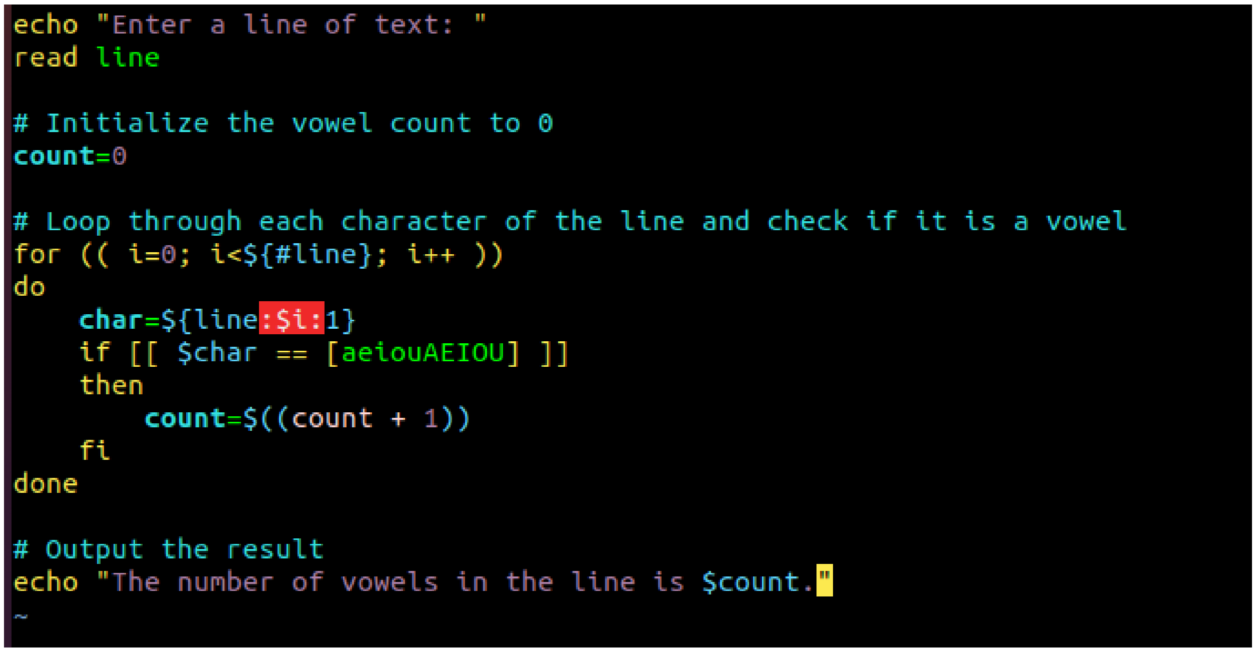
13. Write a Shell program to count the number of vowels in a line of text.

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
```

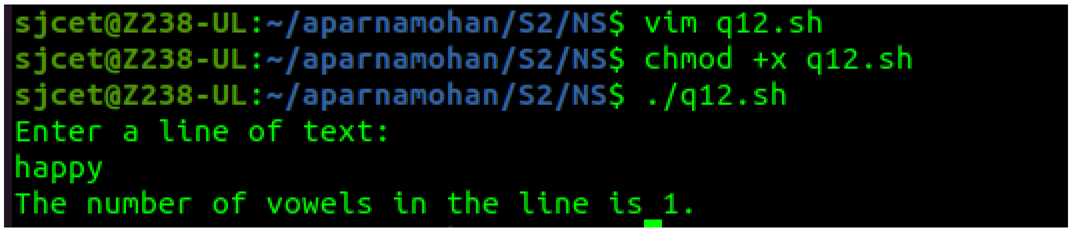
done

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



```
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

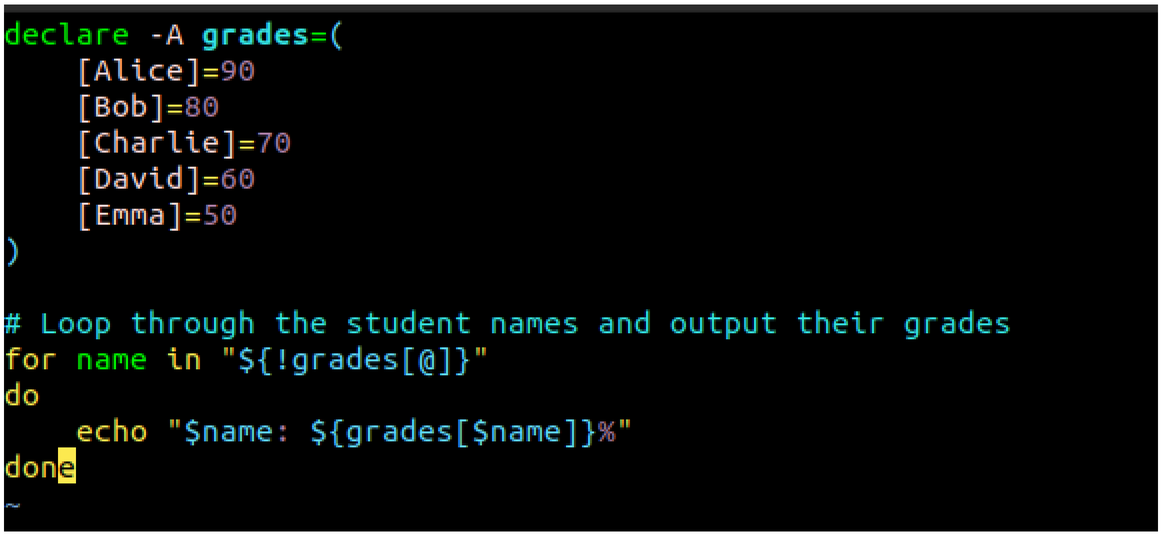


```
sjcet@Z238-UL:~/aparnamohan/S2/NS$ vim q12.sh
sjcet@Z238-UL:~/aparnamohan/S2/NS$ chmod +x q12.sh
sjcet@Z238-UL:~/aparnamohan/S2/NS$ ./q12.sh
Enter a line of text:
happy
The number of vowels in the line is 1.
```

14. Write a Shell program to display student grades.

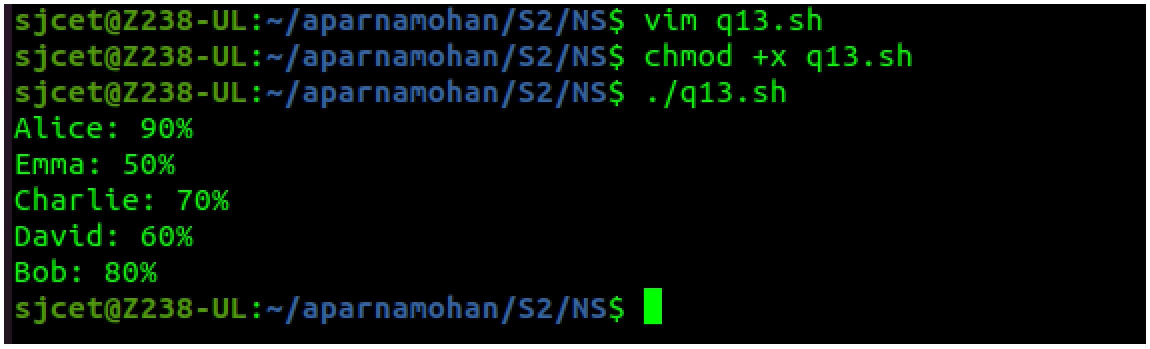
CODE

```
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
```



```
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



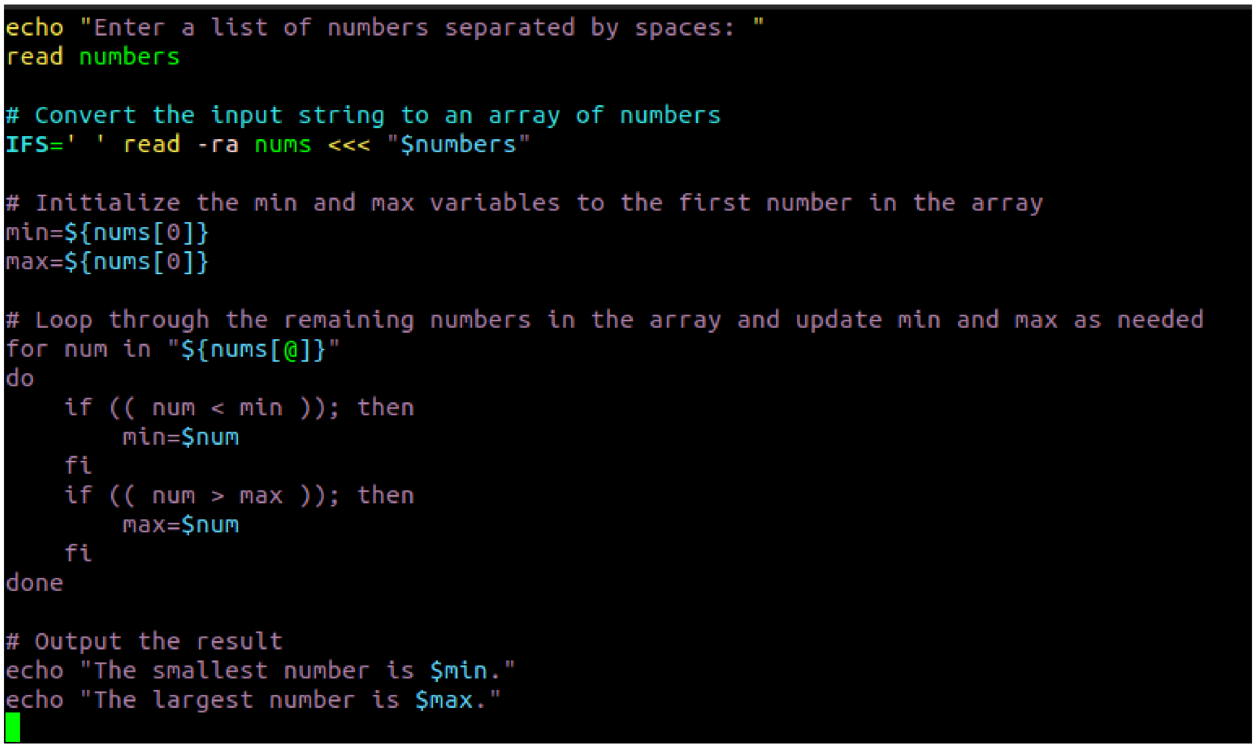
```
sjcet@Z238-UL:~/aparnamohan/S2/NS$ vim q13.sh
sjcet@Z238-UL:~/aparnamohan/S2/NS$ chmod +x q13.sh
sjcet@Z238-UL:~/aparnamohan/S2/NS$ ./q13.sh
Alice: 90%
Emma: 50%
Charlie: 70%
David: 60%
Bob: 80%
sjcet@Z238-UL:~/aparnamohan/S2/NS$
```

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

CODE

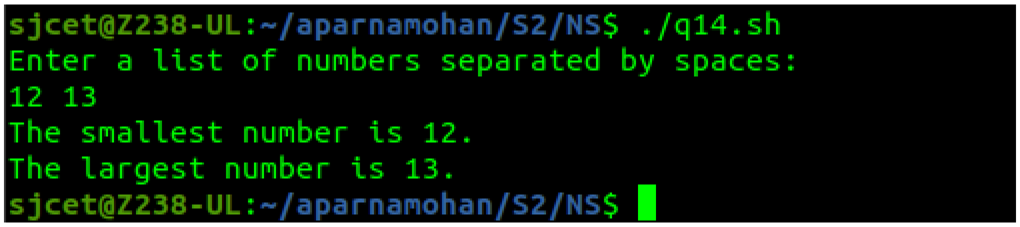
```
echo "Enter a list of numbers separated by spaces: "
read numbers

# 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."
```



```
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

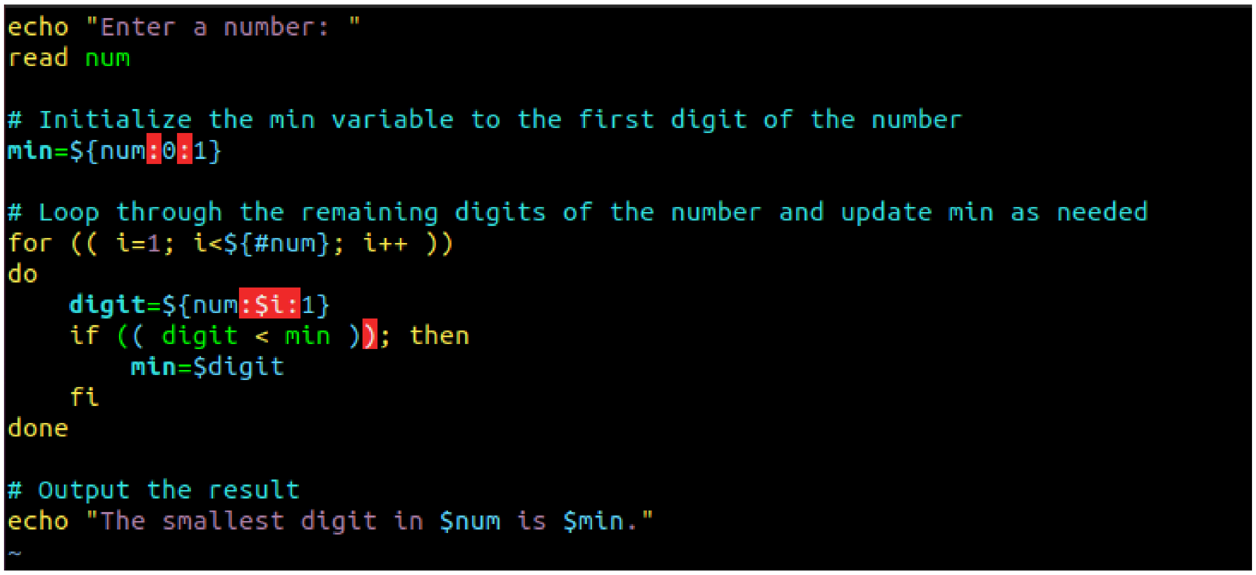


```
sjcet@Z238-UL:~/aparnamohan/S2/NS$ ./q14.sh
Enter a list of numbers separated by spaces:
12 13
The smallest number is 12.
The largest number is 13.
sjcet@Z238-UL:~/aparnamohan/S2/NS$ 
```

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

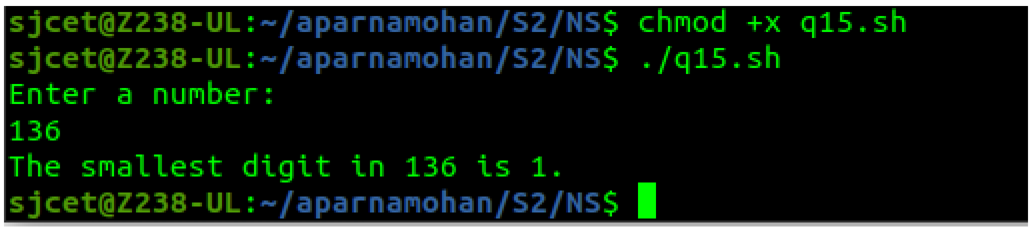
CODE

```
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."
```



```
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



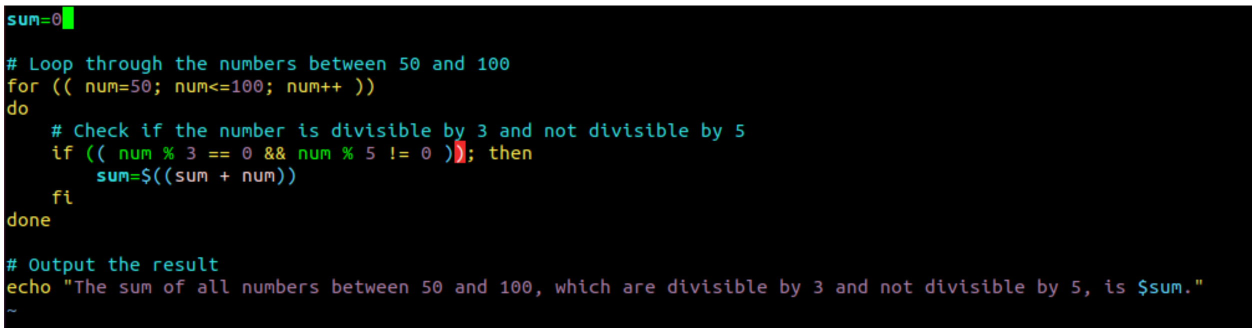
```
sjcet@Z238-UL:~/aparnamohan/S2/NS$ chmod +x q15.sh
sjcet@Z238-UL:~/aparnamohan/S2/NS$ ./q15.sh
Enter a number:
136
The smallest digit in 136 is 1.
sjcet@Z238-UL:~/aparnamohan/S2/NS$ 
```

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

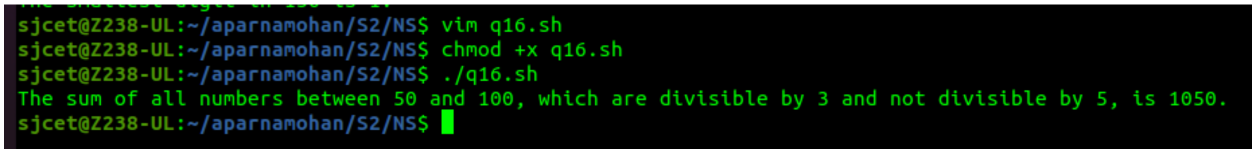
sum=0

```
# 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."
```



```
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

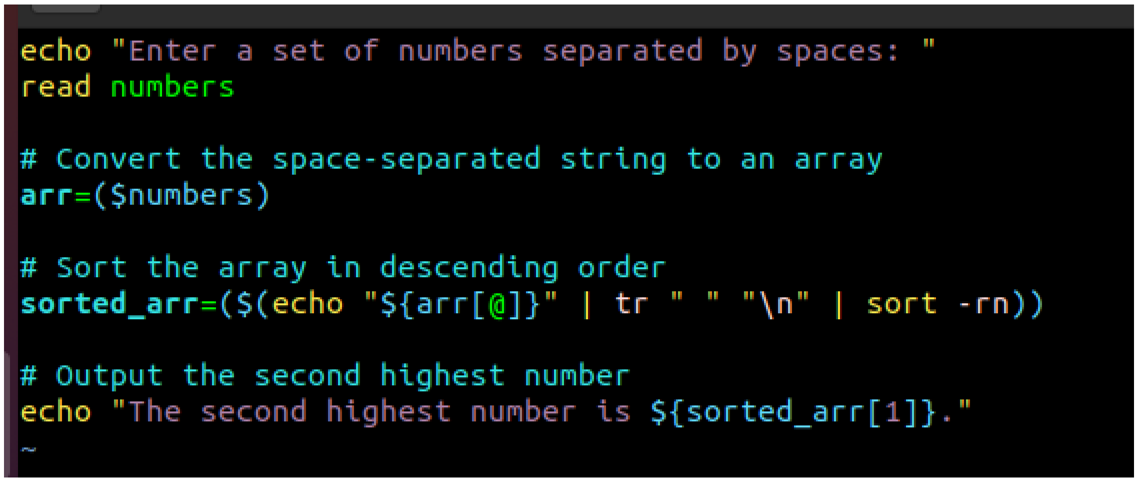


```
sjcet@Z238-UL:~/aparnamohan/S2/NS$ vim q16.sh
sjcet@Z238-UL:~/aparnamohan/S2/NS$ chmod +x q16.sh
sjcet@Z238-UL:~/aparnamohan/S2/NS$ ./q16.sh
The sum of all numbers between 50 and 100, which are divisible by 3 and not divisible by 5, is 1050.
sjcet@Z238-UL:~/aparnamohan/S2/NS$
```

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

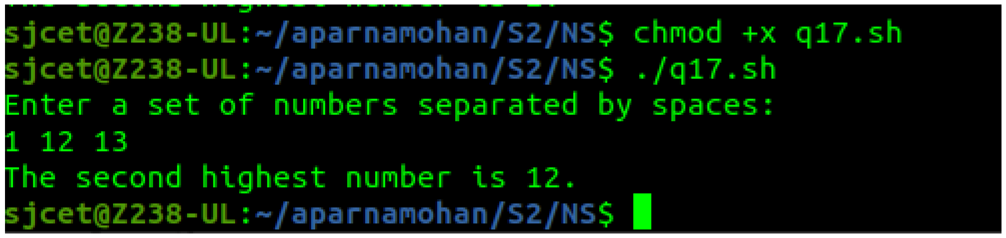
CODE

```
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]}."
```



```
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



```
sjcet@Z238-UL:~/aparnamohan/S2/NS$ chmod +x q17.sh
sjcet@Z238-UL:~/aparnamohan/S2/NS$ ./q17.sh
Enter a set of numbers separated by spaces:
1 12 13
The second highest number is 12.
sjcet@Z238-UL:~/aparnamohan/S2/NS$
```

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

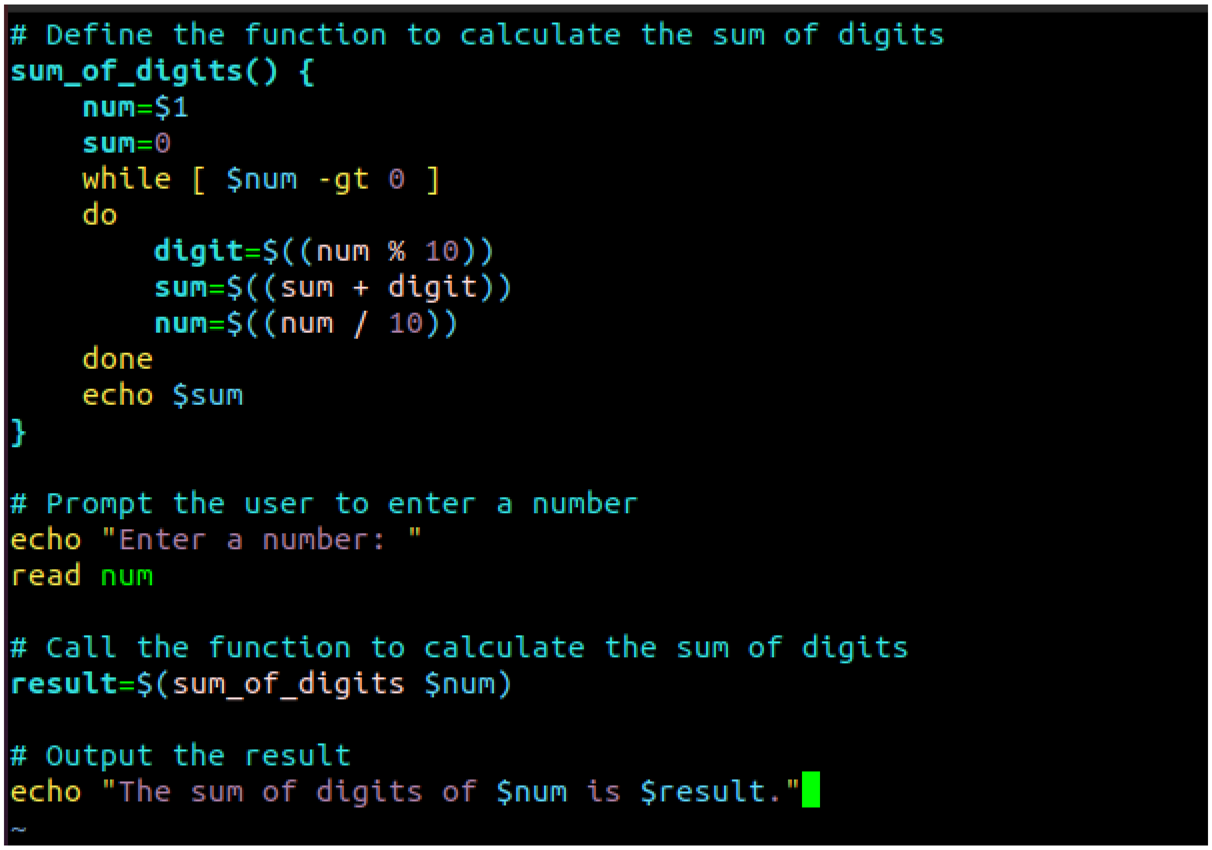
CODE

```
# 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

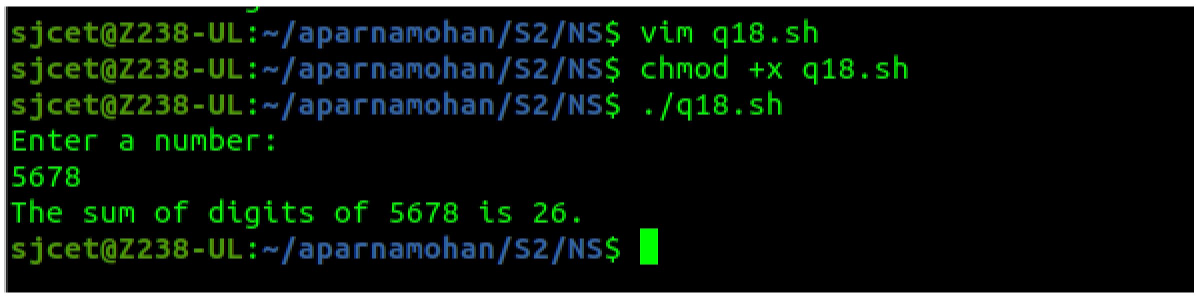
# Call the function to calculate the sum of digits
result=$(sum_of_digits $num)

# Output the result
echo "The sum of digits of $num is $result."



```
# 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

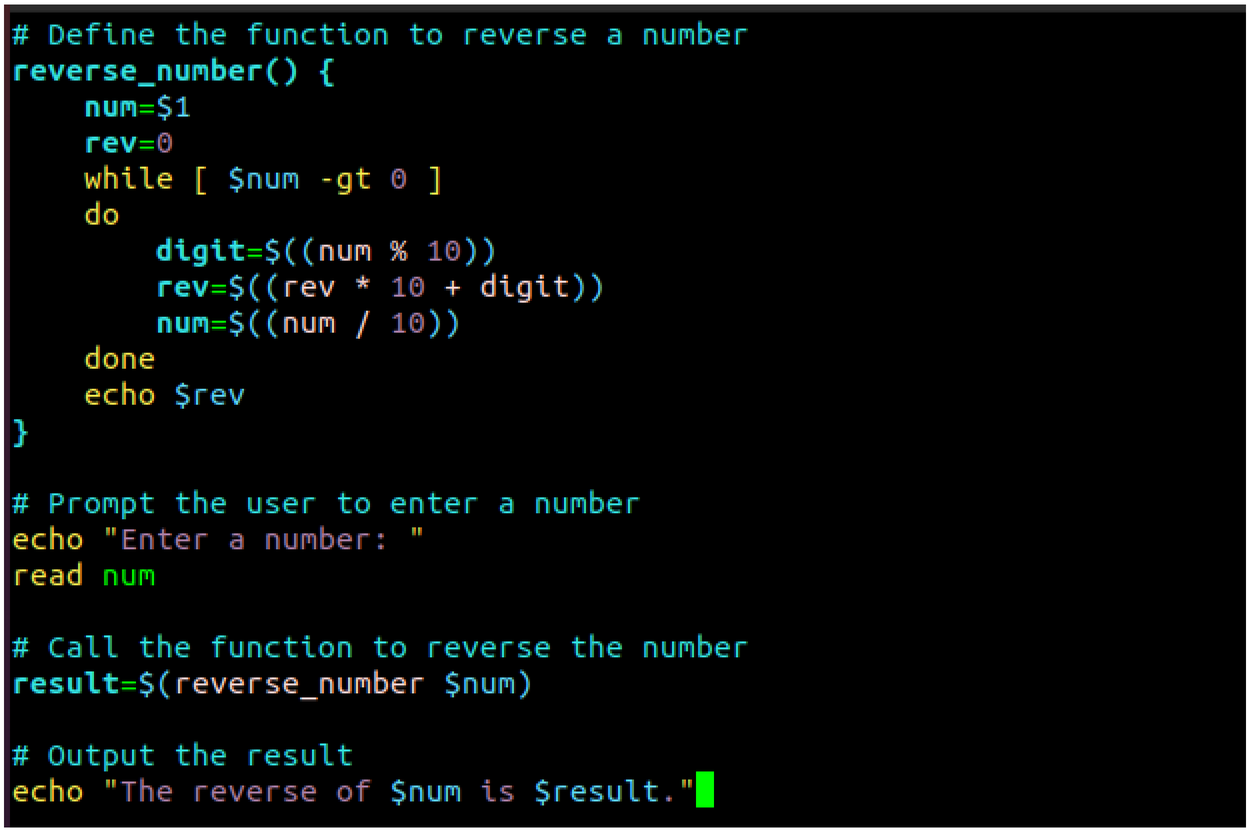


```
sjcet@Z238-UL:~/aparnamohan/S2/NS$ vim q18.sh
sjcet@Z238-UL:~/aparnamohan/S2/NS$ chmod +x q18.sh
sjcet@Z238-UL:~/aparnamohan/S2/NS$ ./q18.sh
Enter a number:
5678
The sum of digits of 5678 is 26.
sjcet@Z238-UL:~/aparnamohan/S2/NS$
```

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

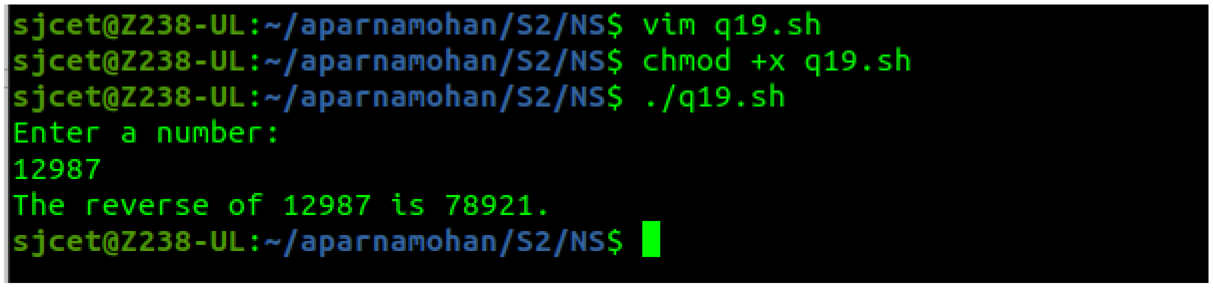
CODE

```
# 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."
```



```
# 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



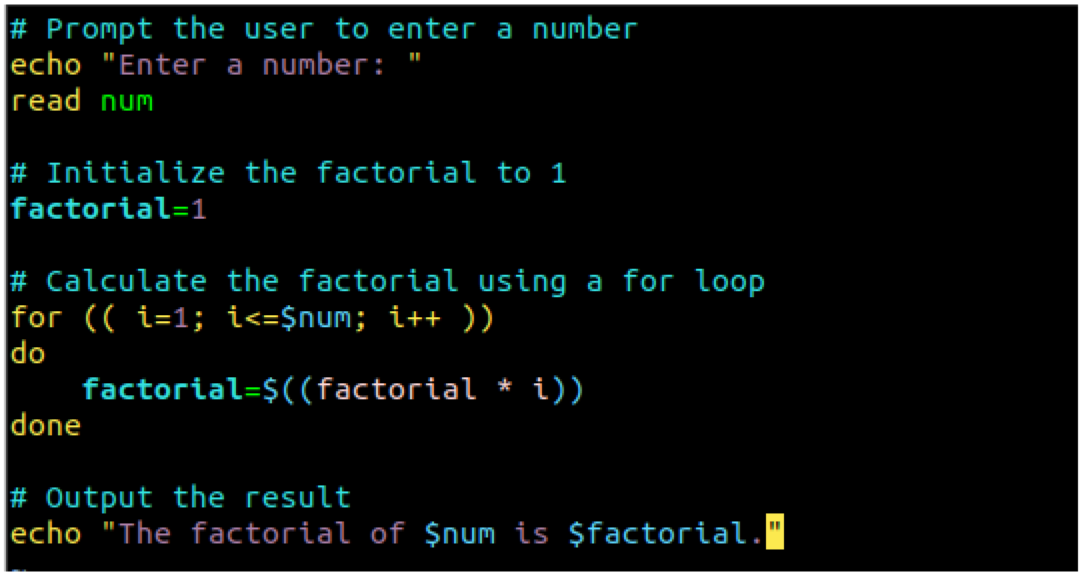
```
sjcet@Z238-UL:~/aparnamohan/S2/NS$ vim q19.sh
sjcet@Z238-UL:~/aparnamohan/S2/NS$ chmod +x q19.sh
sjcet@Z238-UL:~/aparnamohan/S2/NS$ ./q19.sh
Enter a number:
12987
The reverse of 12987 is 78921.
sjcet@Z238-UL:~/aparnamohan/S2/NS$
```

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

CODE

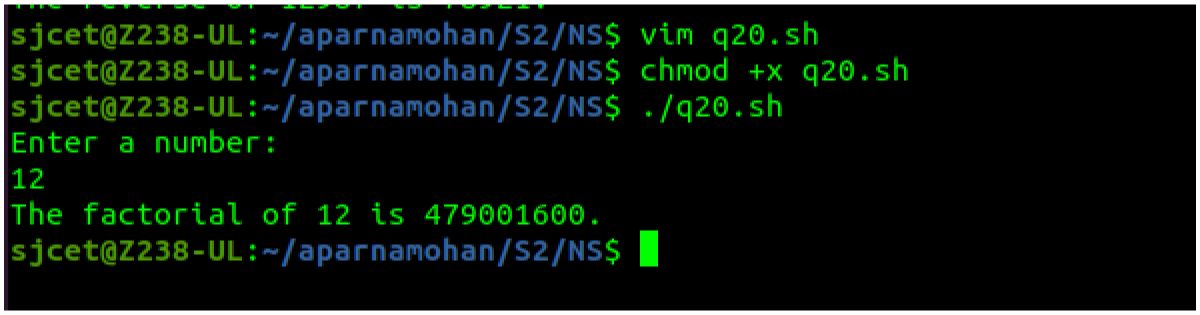
# Prompt the user to enter a number

```
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."
```



```
# 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



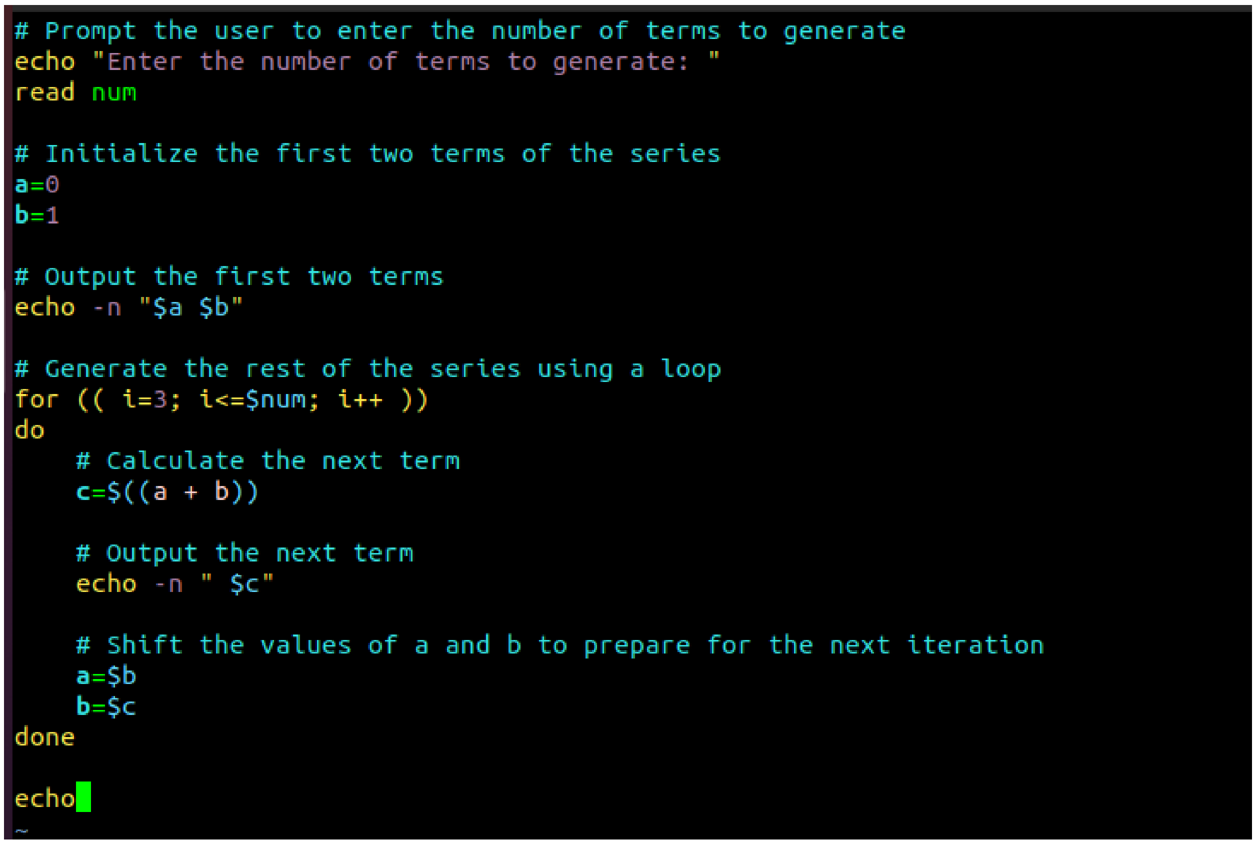
```
sjcet@Z238-UL:~/aparnamohan/S2/NS$ vim q20.sh
sjcet@Z238-UL:~/aparnamohan/S2/NS$ chmod +x q20.sh
sjcet@Z238-UL:~/aparnamohan/S2/NS$ ./q20.sh
Enter a number:
12
The factorial of 12 is 479001600.
sjcet@Z238-UL:~/aparnamohan/S2/NS$
```

22. Write a Shell program to generate Fibonacci series.

CODE

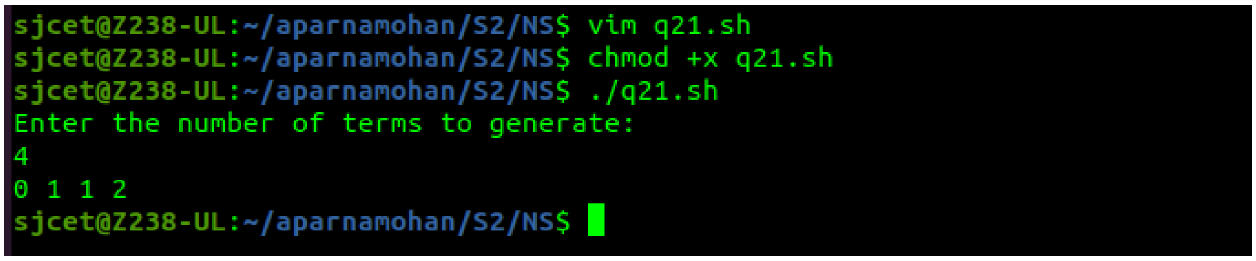
```
# Prompt the user to enter the number of terms to generate
echo "Enter the number of terms to generate: "
read num

# 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
```



```
# 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

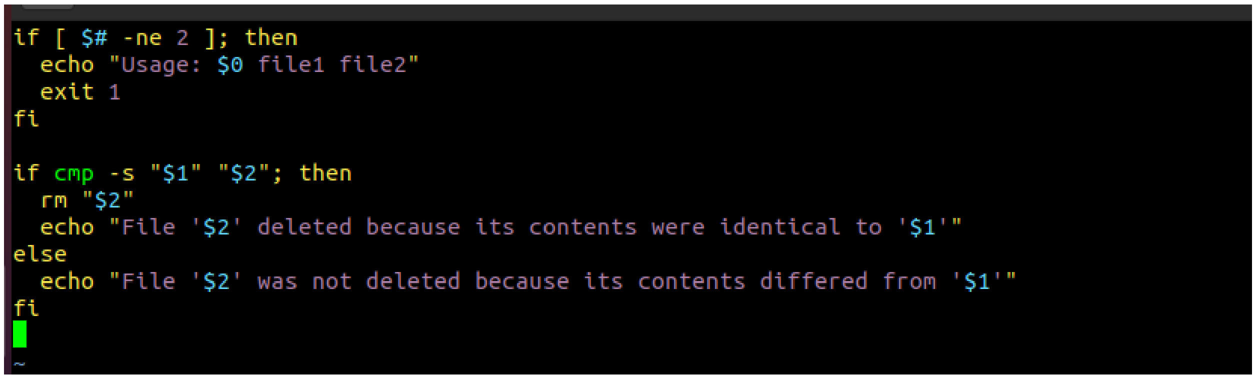


```
sjcet@Z238-UL:~/aparnamohan/S2/NS$ vim q21.sh
sjcet@Z238-UL:~/aparnamohan/S2/NS$ chmod +x q21.sh
sjcet@Z238-UL:~/aparnamohan/S2/NS$ ./q21.sh
Enter the number of terms to generate:
4
0 1 1 2
sjcet@Z238-UL:~/aparnamohan/S2/NS$
```

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.

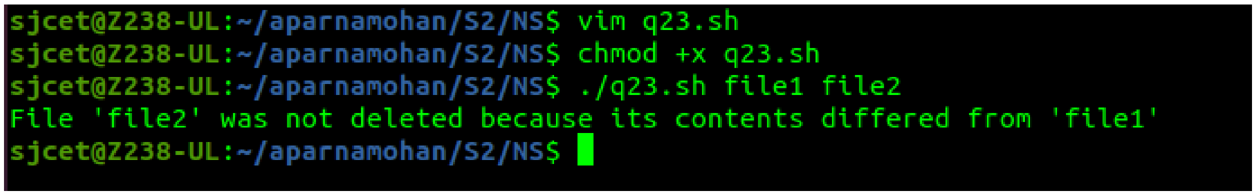
CODE

```
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
```



```
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



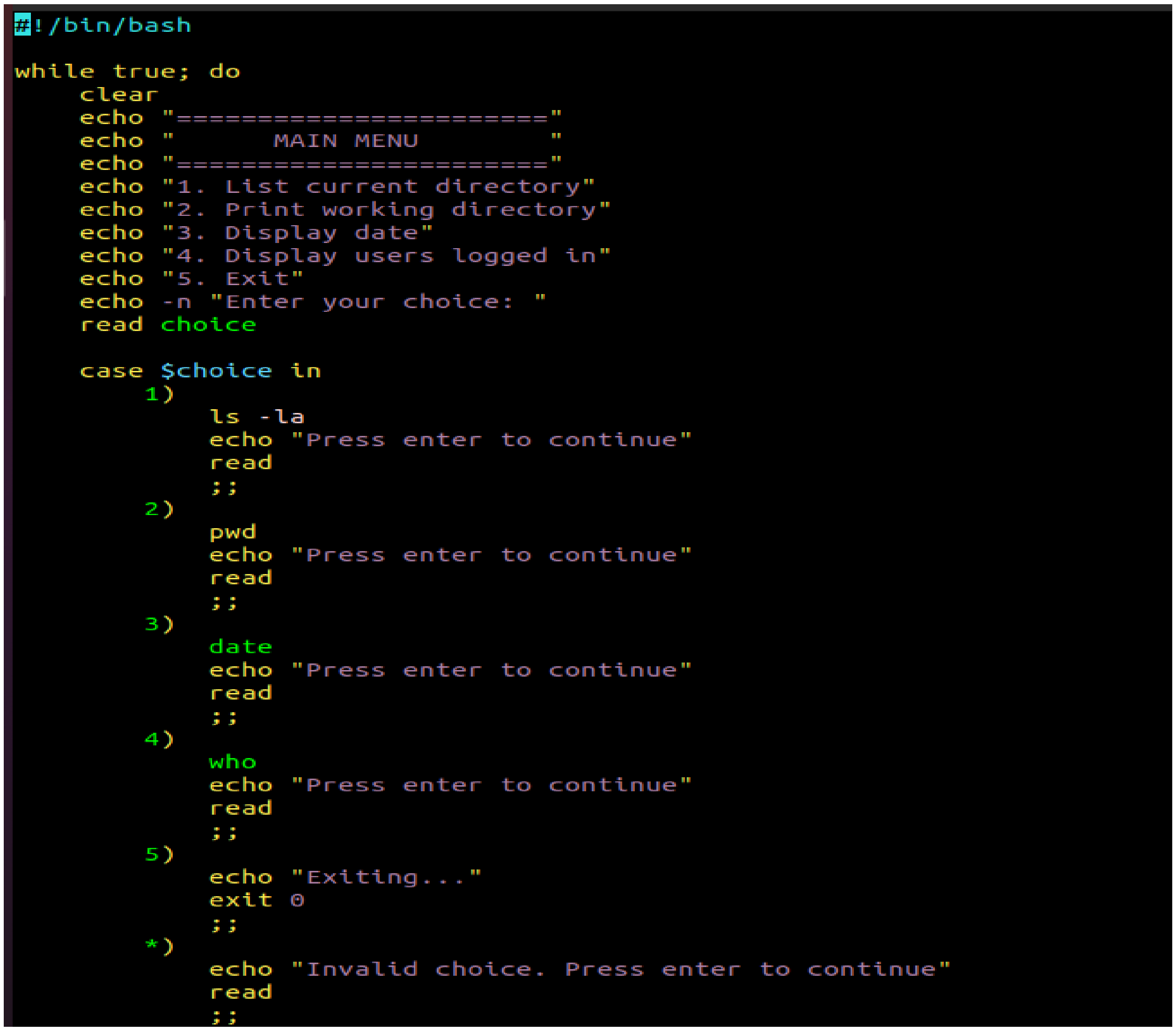
```
sjcet@Z238-UL:~/aparnamohan/S2/NS$ vim q23.sh
sjcet@Z238-UL:~/aparnamohan/S2/NS$ chmod +x q23.sh
sjcet@Z238-UL:~/aparnamohan/S2/NS$ ./q23.sh file1 file2
File 'file2' was not deleted because its contents differed from 'file1'
sjcet@Z238-UL:~/aparnamohan/S2/NS$
```

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

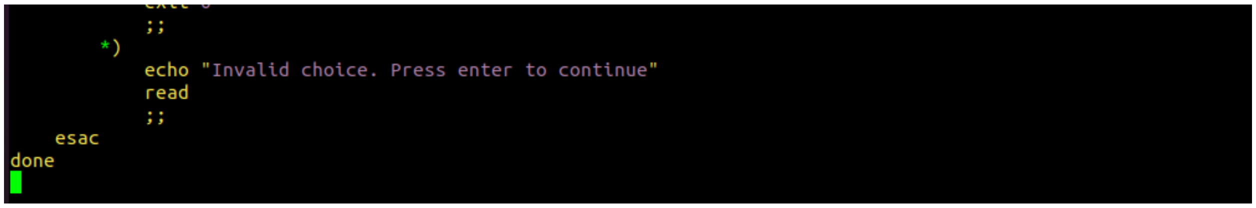
CODE

#!/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
```

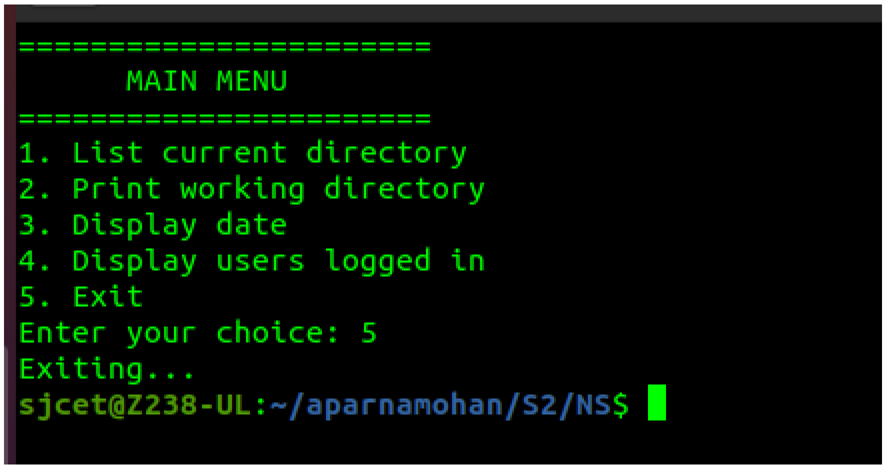


```bash
#!/bin/bash

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
            ;;
```



```bash
            ;;
        *)
            echo "Invalid choice. Press enter to continue"
            read
            ;;
    esac
done
```

OUTPUT

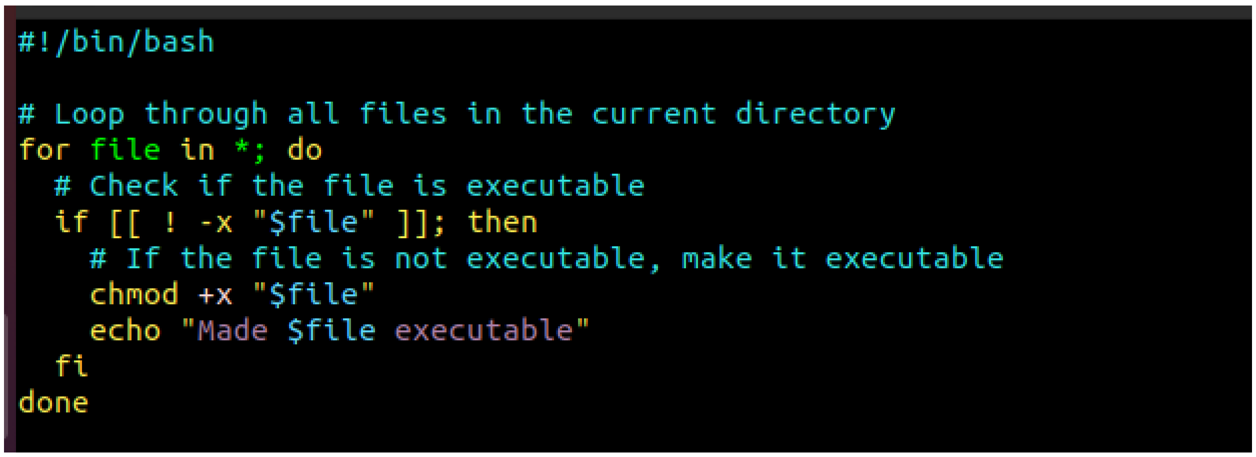


```
=======================
      MAIN MENU
=======================
1. List current directory
2. Print working directory
3. Display date
4. Display users logged in
5. Exit
Enter your choice: 5
Exiting...
sjcet@Z238-UL:~/aparnamohan/S2/NS$ 
```

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.

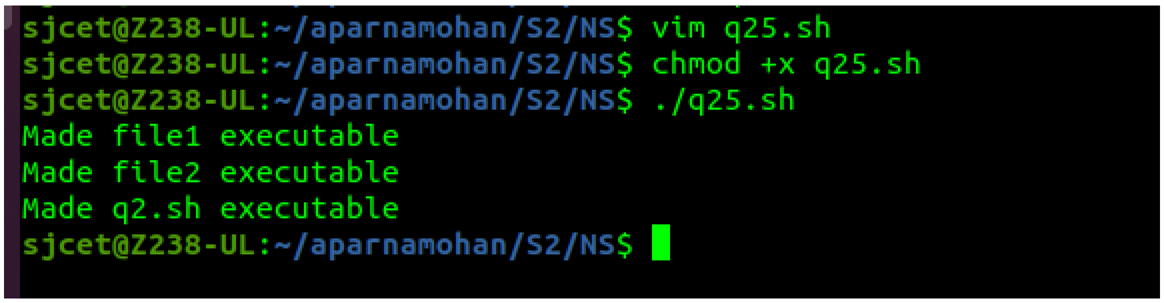
CODE

```
#!/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
```



```
#!/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

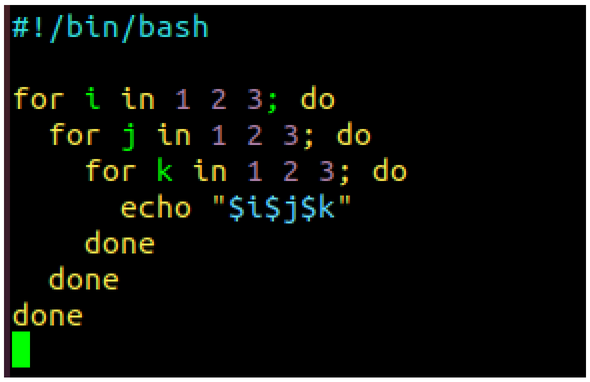


```
sjcet@Z238-UL:~/aparnamohan/S2/NS$ vim q25.sh
sjcet@Z238-UL:~/aparnamohan/S2/NS$ chmod +x q25.sh
sjcet@Z238-UL:~/aparnamohan/S2/NS$ ./q25.sh
Made file1 executable
Made file2 executable
Made q2.sh executable
sjcet@Z238-UL:~/aparnamohan/S2/NS$ 
```

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

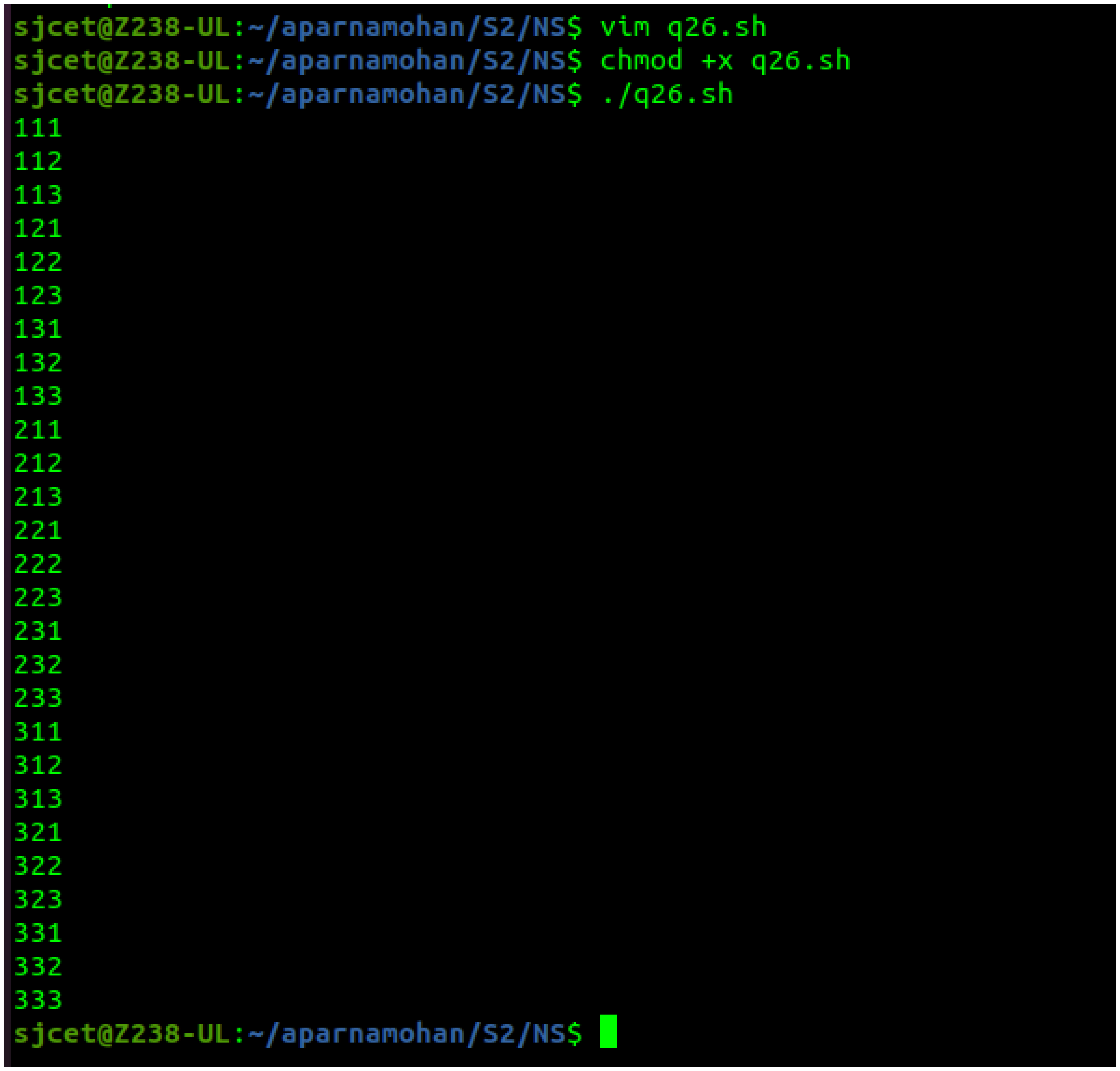
CODE

```
#!/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
```



```
#!/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



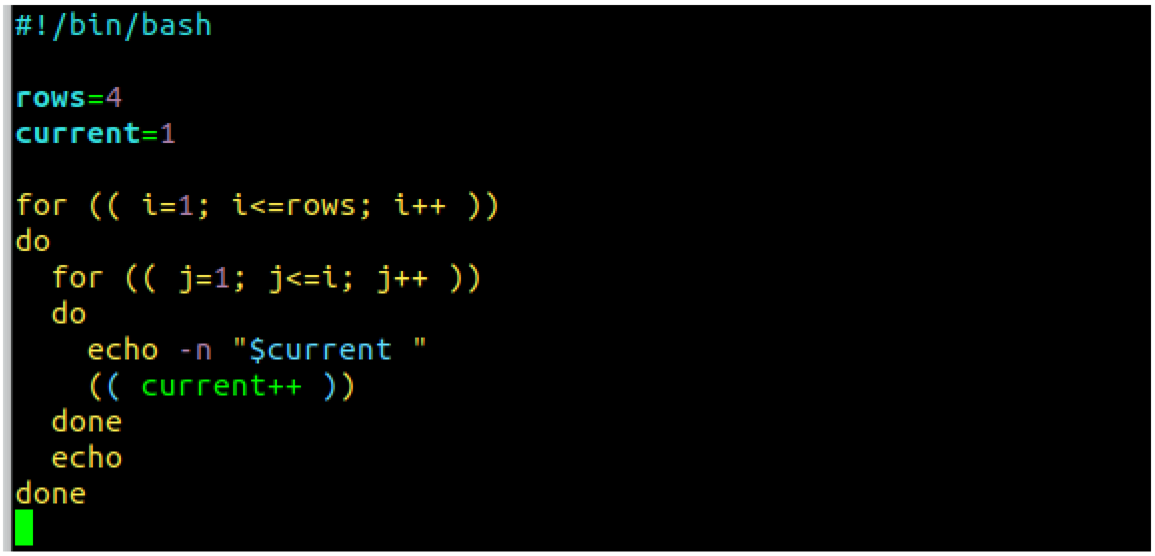
```
sjcet@Z238-UL:~/aparnamohan/S2/NS$ vim q26.sh
sjcet@Z238-UL:~/aparnamohan/S2/NS$ chmod +x q26.sh
sjcet@Z238-UL:~/aparnamohan/S2/NS$ ./q26.sh
111
112
113
121
122
123
131
132
133
211
212
213
221
222
223
231
232
233
311
312
313
321
322
323
331
332
333
sjcet@Z238-UL:~/aparnamohan/S2/NS$ 
```

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

1
2 3
4 5 6
7___ 8 9 10

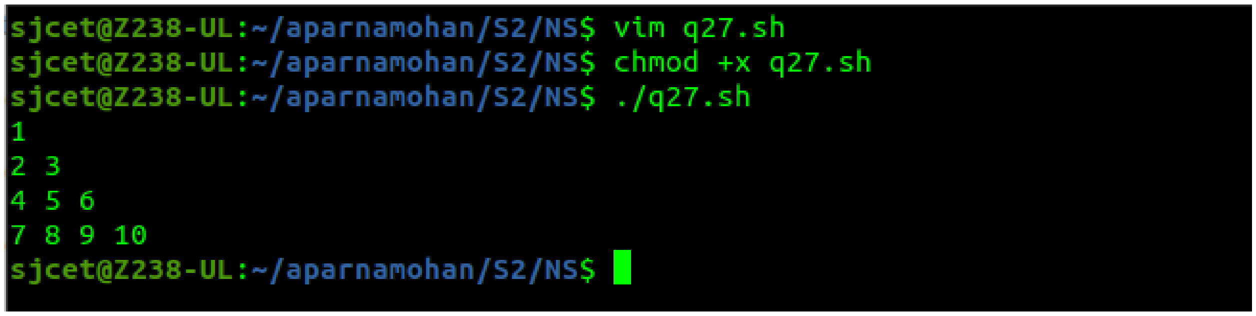
CODE

```
#!/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
```



```
#!/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

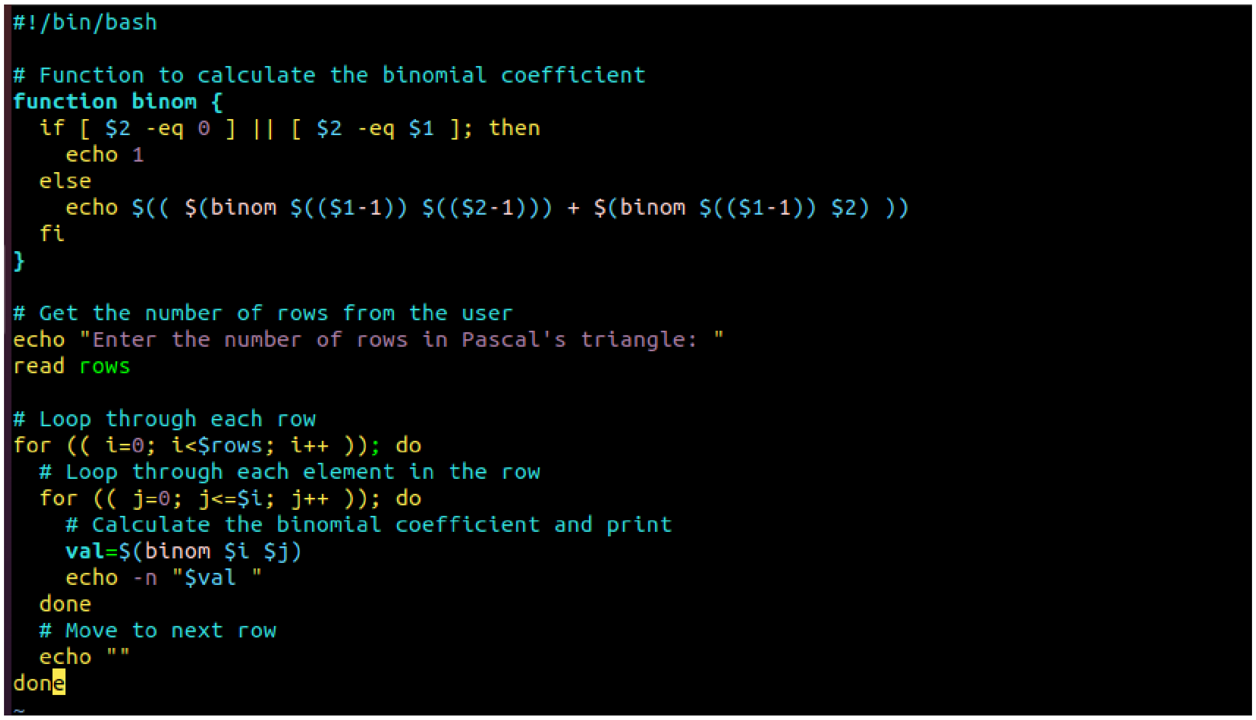

```
sjcet@Z238-UL:~/aparnamohan/S2/NS$ vim q27.sh
sjcet@Z238-UL:~/aparnamohan/S2/NS$ chmod +x q27.sh
sjcet@Z238-UL:~/aparnamohan/S2/NS$ ./q27.sh
1
2 3
4 5 6
7 8 9 10
sjcet@Z238-UL:~/aparnamohan/S2/NS$ 
```

28. Write a Shell program to create Pascal's triangle.

CODE

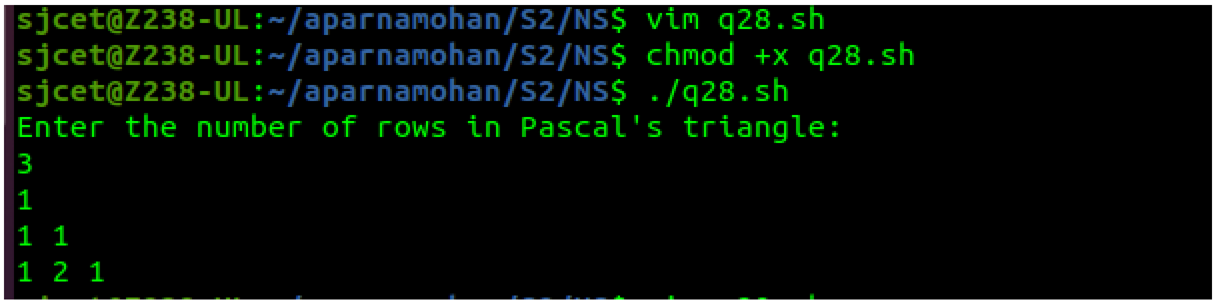
```
#!/bin/bash

# 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
```



```
#!/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

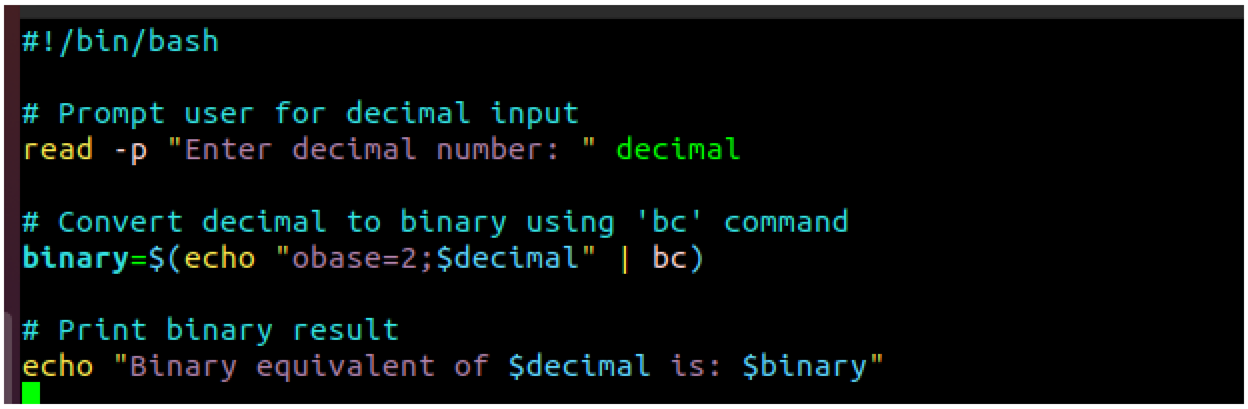


```
sjcet@Z238-UL:~/aparnamohan/S2/NS$ vim q28.sh
sjcet@Z238-UL:~/aparnamohan/S2/NS$ chmod +x q28.sh
sjcet@Z238-UL:~/aparnamohan/S2/NS$ ./q28.sh
Enter the number of rows in Pascal's triangle:
3
1
1 1
1 2 1
```

29. Write a Decimal to Binary Conversion Shell Script

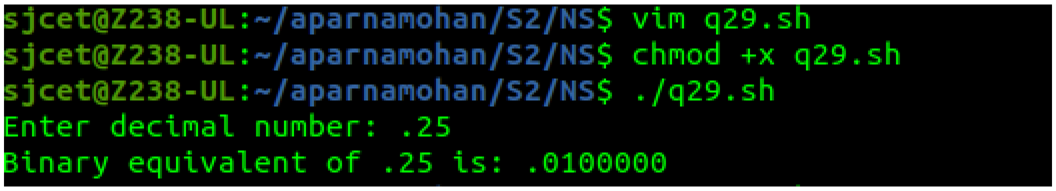
CODE

```
#!/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"
```



```
#!/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



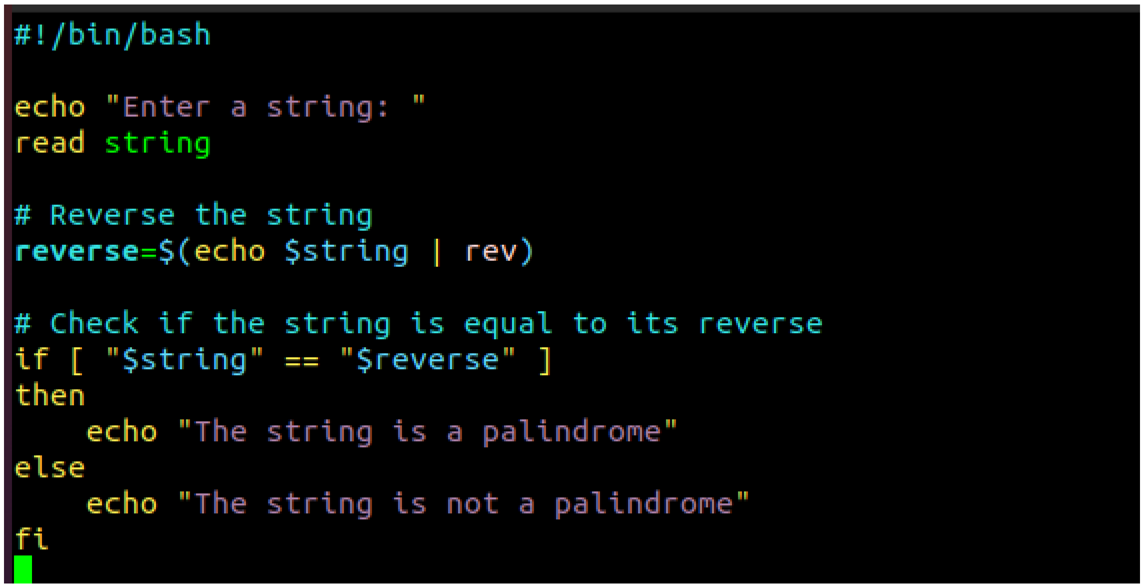
```
sjcet@Z238-UL:~/aparnamohan/S2/NS$ vim q29.sh
sjcet@Z238-UL:~/aparnamohan/S2/NS$ chmod +x q29.sh
sjcet@Z238-UL:~/aparnamohan/S2/NS$ ./q29.sh
Enter decimal number: .25
Binary equivalent of .25 is: .0100000
```

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

CODE

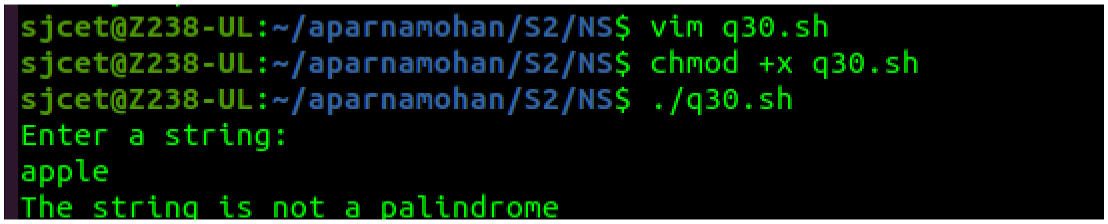
```
#!/bin/bash

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
```



```
#!/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



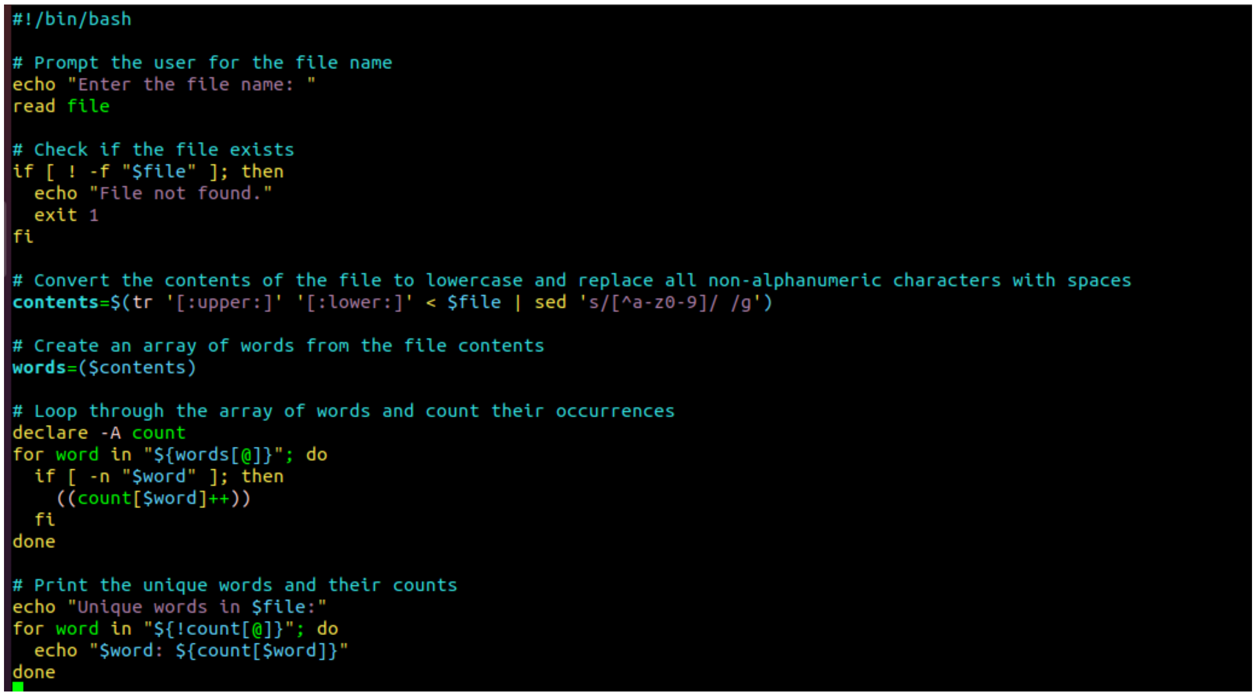
```
sjcet@Z238-UL:~/aparnamohan/S2/NS$ vim q30.sh
sjcet@Z238-UL:~/aparnamohan/S2/NS$ chmod +x q30.sh
sjcet@Z238-UL:~/aparnamohan/S2/NS$ ./q30.sh
Enter a string:
apple
The string is not a palindrome
```

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

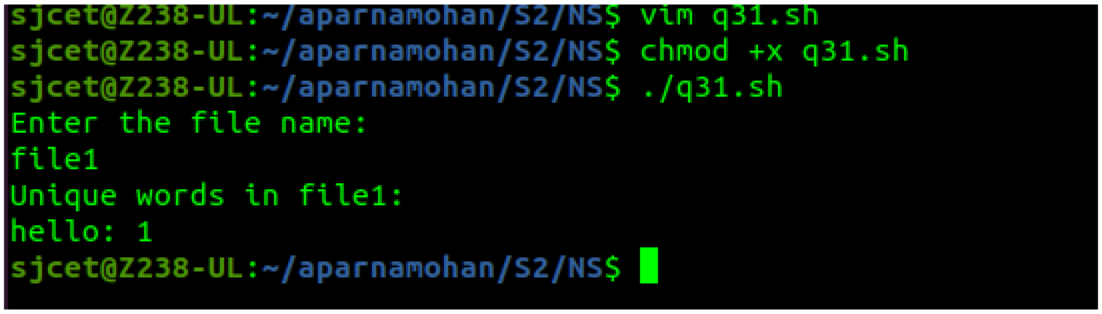
```
#!/bin/bash

# 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
```



```
#!/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

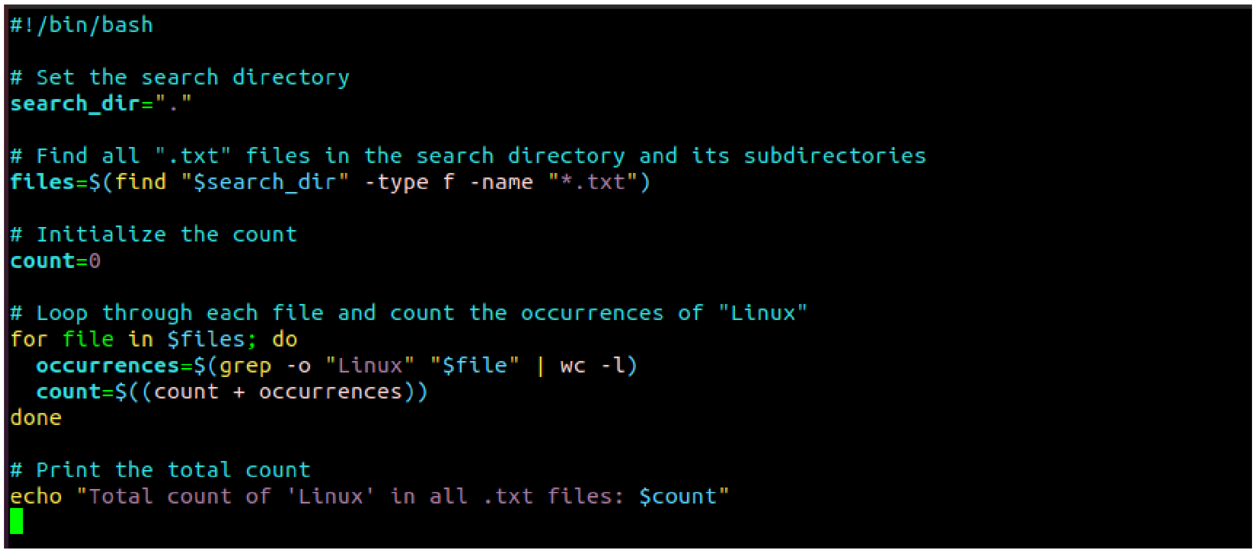


```
sjcet@Z238-UL:~/aparnamohan/S2/NS$ vim q31.sh
sjcet@Z238-UL:~/aparnamohan/S2/NS$ chmod +x q31.sh
sjcet@Z238-UL:~/aparnamohan/S2/NS$ ./q31.sh
Enter the file name:
file1
Unique words in file1:
hello: 1
sjcet@Z238-UL:~/aparnamohan/S2/NS$
```

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.

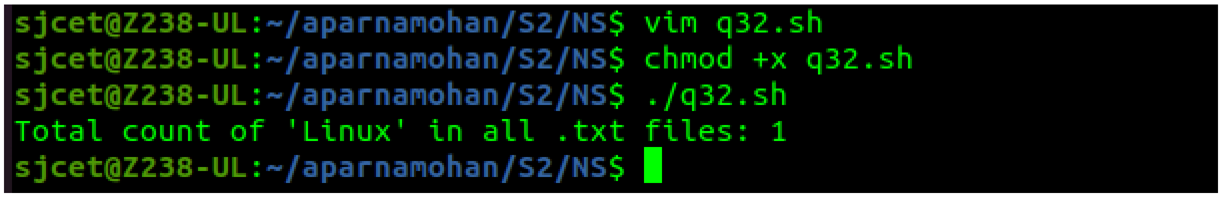
CODE

```
#!/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"
```



```
#!/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

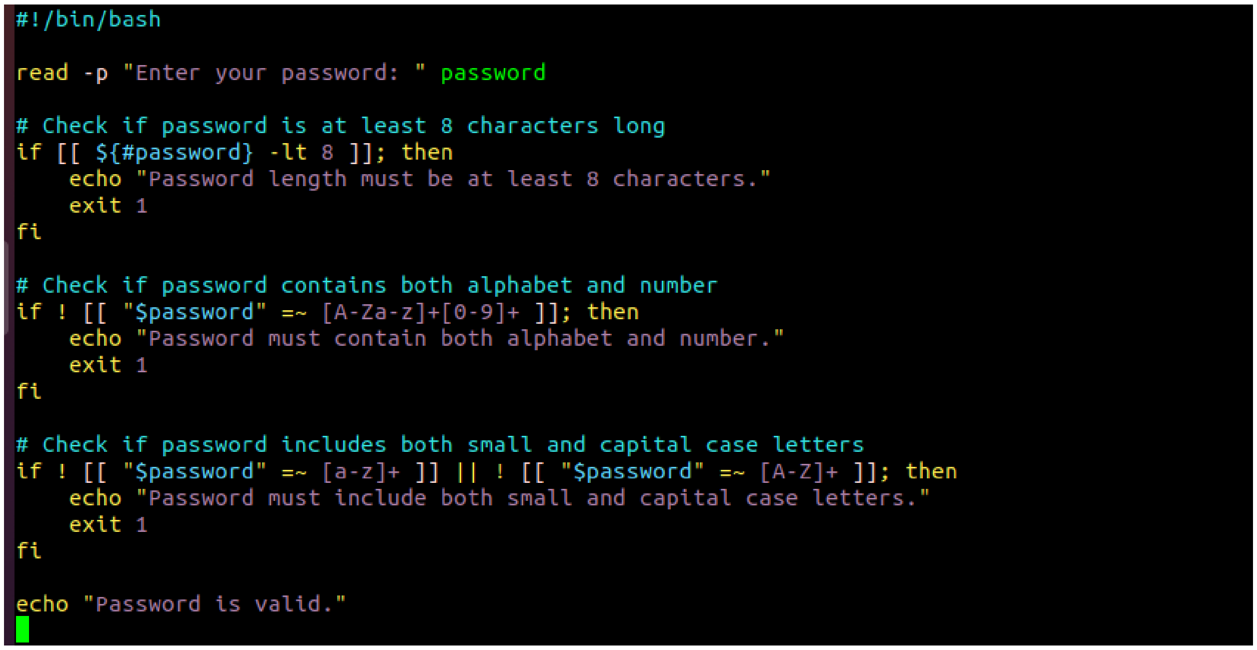


```
sjcet@Z238-UL:~/aparnamohan/S2/NS$ vim q32.sh
sjcet@Z238-UL:~/aparnamohan/S2/NS$ chmod +x q32.sh
sjcet@Z238-UL:~/aparnamohan/S2/NS$ ./q32.sh
Total count of 'Linux' in all .txt files: 1
sjcet@Z238-UL:~/aparnamohan/S2/NS$ 
```

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.

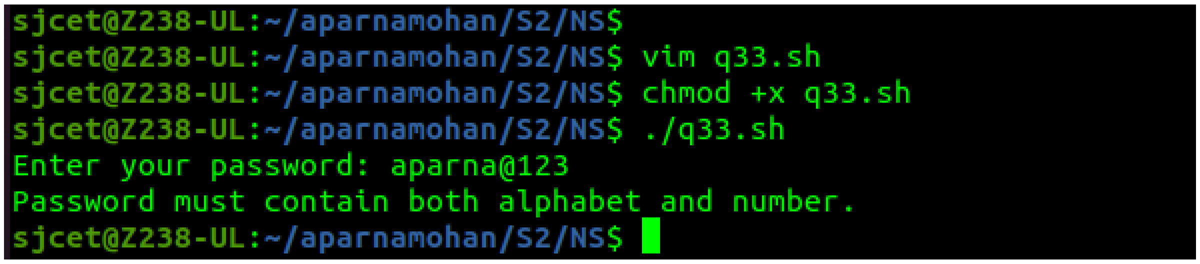
CODE

```
#!/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."
```



```
#!/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



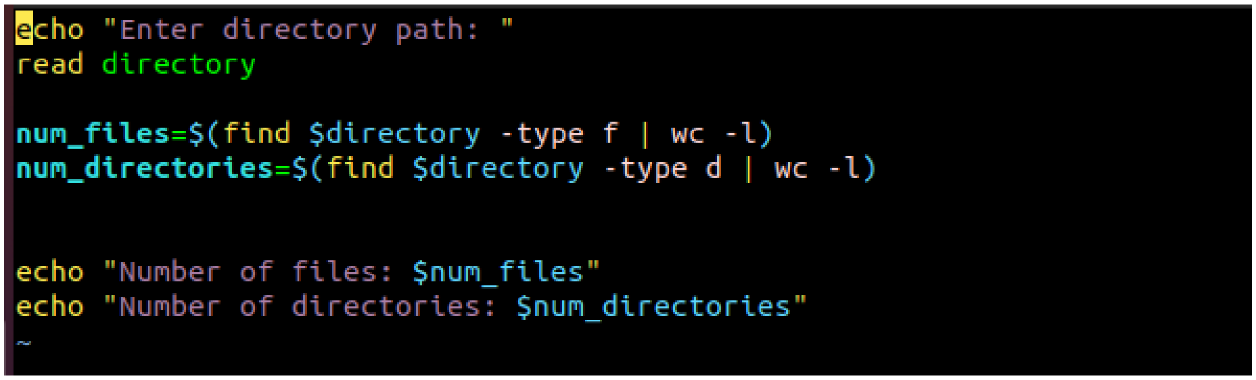
```
sjcet@Z238-UL:~/aparnamohan/S2/NS$
sjcet@Z238-UL:~/aparnamohan/S2/NS$ vim q33.sh
sjcet@Z238-UL:~/aparnamohan/S2/NS$ chmod +x q33.sh
sjcet@Z238-UL:~/aparnamohan/S2/NS$ ./q33.sh
Enter your password: aparna@123
Password must contain both alphabet and number.
sjcet@Z238-UL:~/aparnamohan/S2/NS$
```

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

CODE

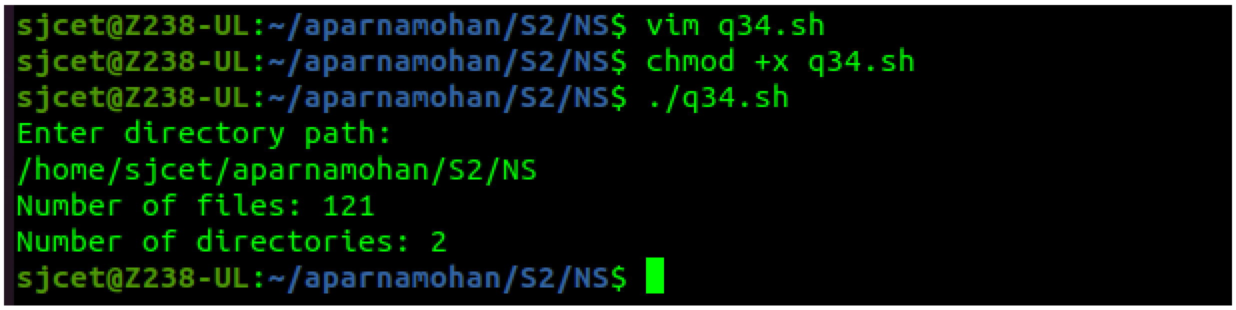
```
echo "Enter directory path: "
read directory

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"



```
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

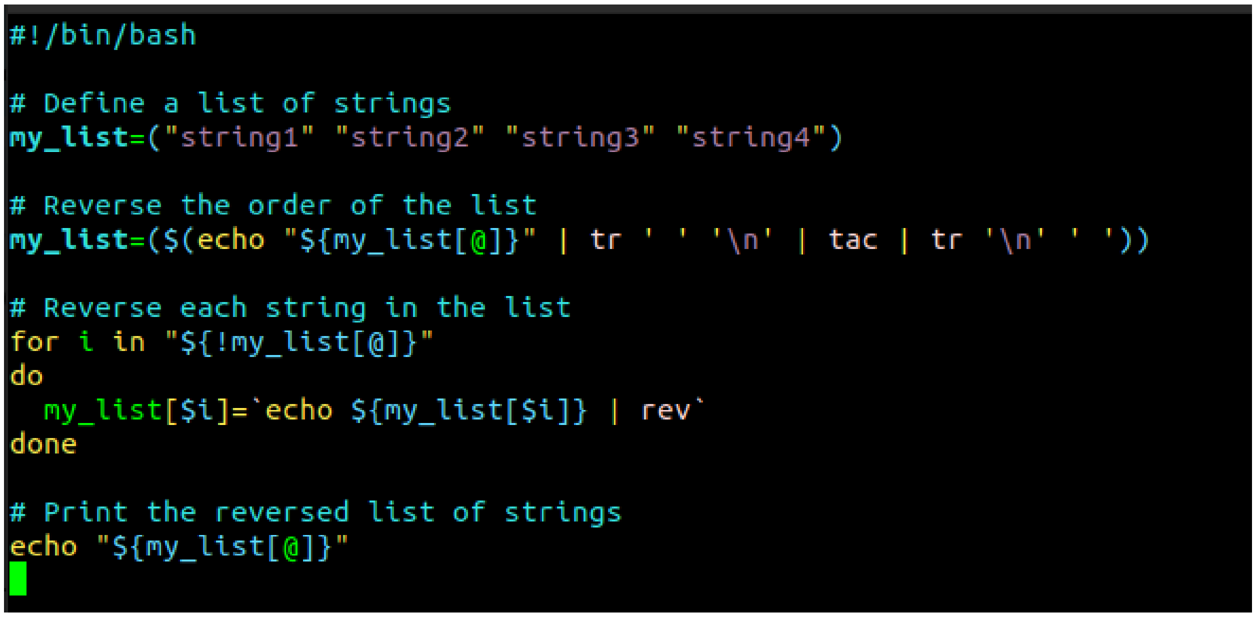


```
sjcet@Z238-UL:~/aparnamohan/S2/NS$ vim q34.sh
sjcet@Z238-UL:~/aparnamohan/S2/NS$ chmod +x q34.sh
sjcet@Z238-UL:~/aparnamohan/S2/NS$ ./q34.sh
Enter directory path:
/home/sjcet/aparnamohan/S2/NS
Number of files: 121
Number of directories: 2
sjcet@Z238-UL:~/aparnamohan/S2/NS$
```

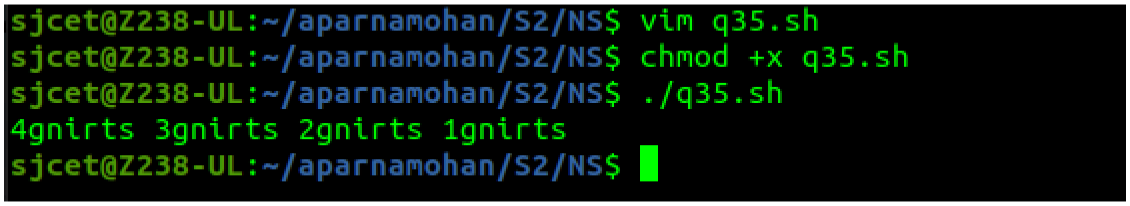
35.write a shell script to reverse list of strings and reverse each string further in the list.

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[@]}"
```


```
#!/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


```
sjcet@Z238-UL:~/aparnamohan/S2/NS$ vim q35.sh
sjcet@Z238-UL:~/aparnamohan/S2/NS$ chmod +x q35.sh
sjcet@Z238-UL:~/aparnamohan/S2/NS$ ./q35.sh
4gnirts 3gnirts 2gnirts 1gnirts
sjcet@Z238-UL:~/aparnamohan/S2/NS$ 
```