

SHELL SCRIPTING PROGRAMS

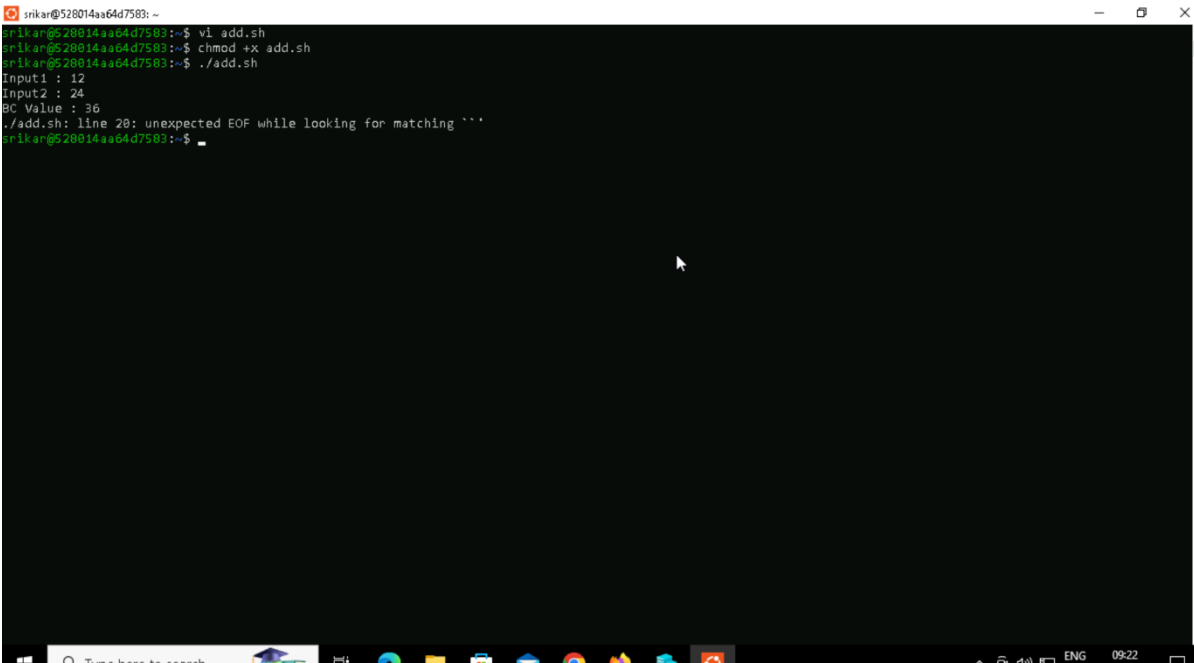
1. Sum of two numbers

CODE

```
srikan@528014aa64d7583: ~  
$ vi /usr/bin/bash  
read -p "Input1 : " inp1  
  
if [[ -z $inp1 ]]  
then  
    echo "Input1 cannot be empty, please enter an integer."  
    exit  
fi  
  
read -p "Input2 : " inp2  
if [[ -z $inp2 ]]  
then  
    echo "Input2 cannot be empty, please enter an integer."  
    exit  
fi  
  
bc_val=$(echo "$inp1+$inp2" | bc)  
echo "BC Value : $bc_val"  
  
expr_val=$((inp1 + inp2))  
echo "EXPR Value : $expr_val"
```

"add.sh" 21L, 425B

OUTPUT



```
snikar@528014aa64d7583: ~  
snikar@528014aa64d7583:~$ v1 add.sh  
snikar@528014aa64d7583:~$ chmod +x add.sh  
snikar@528014aa64d7583:~$ ./add.sh  
Input1 : 12  
Input2 : 24  
BC Value : 36  
./add.sh: line 20: unexpected EOF while looking for matching `'  
snikar@528014aa64d7583:~$
```

2. Sum of array

CODE

Sum of array code taking the input directly

```
Select snikar@528014aa64d7583: ~  
$1/bin/bash  
arr=(39 10 26 1 31)  
sum=0  
  
for (( i = 0; i < ${#arr[*]}; i++ )); do  
    if (( arr[i] > 0 )); then  
        sum=$((sum + arr[i]))  
    fi  
done  
echo $sum  
  
sumarray.sh 11L, 166B
```

Sum of array code taking the user input

```
snikar@528014aa64d7583: ~  
$1/bin/bash  
read -p "Enter the size of the array: " size  
echo "Enter $size elements:"  
read -a arr  
sum=0  
for (( i = 0; i < ${#arr[*]}; i++ )); do  
    if (( arr[i] > 0 )); then  
        sum=$((sum + arr[i]))  
    fi  
done  
echo "Sum of positive numbers: $sum"  
  
sumarray1.sh 12L, 249B
```

OUTPUT

```
snikar@528014aa64d7583: ~  
snikar@528014aa64d7583:~$ vi sumarray.sh  
snikar@528014aa64d7583:~$ chmod +x sumarray.sh  
snikar@528014aa64d7583:~$ ./sumarray.sh  
81  
snikar@528014aa64d7583:~$ vi sumarray1.sh  
snikar@528014aa64d7583:~$ vi sumarray.sh  
snikar@528014aa64d7583:~$ vi sumarray1.sh  
snikar@528014aa64d7583:~$ chmod +x sumarray1.sh  
snikar@528014aa64d7583:~$ ./sumarray1.sh  
Enter the size of the array: 6  
Enter 6 elements:  
10 20 30 40 50 60  
Sum of positive numbers: 210  
snikar@528014aa64d7583:~$
```

3. Reverse a number

CODE

```
snikar@528014aa64d7583: ~  
#!/bin/bash  
read -p "Enter a number: " n  
if ! [[ $n =~ ^-[0-9]+$ ]]; then  
    echo "Error: Please enter a valid integer!"  
    exit 1  
fi  
  
rev=0  
sign=1  
if (( n < 0 )); then  
    sign=-1  
    n=$(( -n ))  
fi  
  
while (( n > 0 )); do  
    rev=$(( rev * 10 + n % 10 ))  
    n=$(( n / 10 ))  
done  
rev=$(( rev * sign ))  
echo "Reverse Number is $rev"  
  
"Reverse.sh" 21L, 355B
```

OUTPUT

```
srikiran@528014aa64d7583: ~  
srikiran@528014aa64d7583:~$ vi reverse.sh  
srikiran@528014aa64d7583:~$ chmod +x reverse.sh  
srikiran@528014aa64d7583:~$ ./reverse.sh  
Enter a number: 1234  
Reverse Number is 4321  
srikiran@528014aa64d7583:~$ ./reverse.sh  
Enter a number: -2345  
Reverse Number is -5432  
srikiran@528014aa64d7583:~$ ./reverse.sh  
Enter a number:  
Error: Please enter a valid integer!  
srikiran@528014aa64d7583:~$
```

4. Palindrome number

CODE

```
#!/bin/bash  
read -p "Enter a number: " n  
if ! [[ $n =~ ^-[0-9]+$ ]]; then  
    echo "Error: Please enter a valid integer!"  
    exit 1  
fi  
  
original_n=$n  
rev=0  
sign=1  
if (( n < 0 )); then  
    sign=-1  
n=$(( -n ))  
fi  
  
while (( n > 0 )); do  
    rev=$(( rev * 10 + n % 10 ))  
    n=$(( n / 10 ))  
done  
rev=$(( rev * sign ))  
echo "Reverse Number is $rev"  
  
if [ "$rev" -eq "$original_n" ]; then  
    echo "The number $original_n is a palindrome."  
else  
    echo "The number $original_n is not a palindrome."  
fi
```

OUTPUT

```
snikar@528014aa64d7583: ~  
snikar@528014aa64d7583:~$ vi palindrome.sh  
snikar@528014aa64d7583:~$ chmod +x palindrome.sh  
snikar@528014aa64d7583:~$ ./palindrome.sh  
Enter a number: 1234  
Reverse Number is 4321  
The number 1234 is not a palindrome.  
snikar@528014aa64d7583:~$ ./palindrome.sh  
Enter a number: 12321  
Reverse Number is 12321  
The number 12321 is a palindrome.  
snikar@528014aa64d7583:~$ ./palindrome.sh  
Enter a number: -345  
Reverse Number is -543  
The number -345 is not a palindrome.  
snikar@528014aa64d7583:~$
```

5. Bubble sort

CODE

```
snikar@528014aa64d7583: ~  
#!/bin/bash  
echo "Enter array elements separated by space:"  
read -a arr  
  
n=${#arr[@]}  
for ((i = 0; i < n-1; i++)); do  
    for ((j = 0; j < n-i-1; j++)); do  
        if (( arr[j] > arr[j+1] )); then  
            temp=${arr[j]}  
            arr[j]=${arr[j+1]}  
            arr[j+1]=$temp  
        fi  
    done  
done  
echo "Sorted array:"  
echo "${arr[@]}"  
  
bubble.sh" 17L, 316B
```

OUTPUT

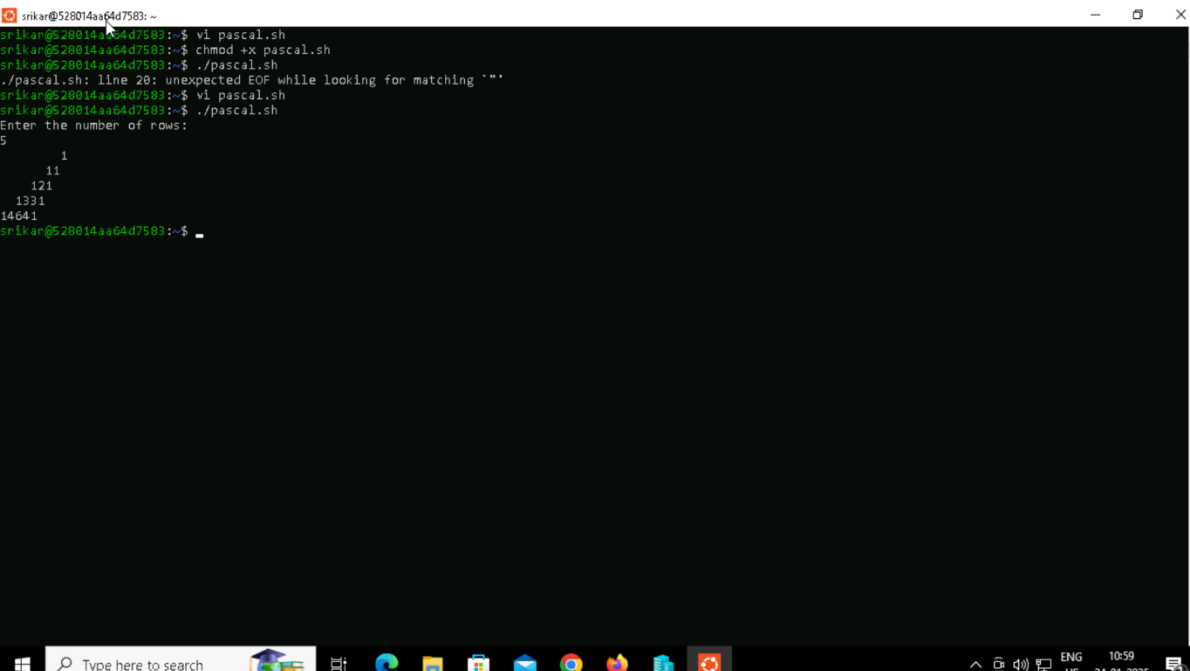
```
srikan@528014aa64d7583: ~  
srikan@528014aa64d7583:~$ vi bubble.sh  
srikan@528014aa64d7583:~$ chmod +x bubble.sh  
srikan@528014aa64d7583:~$ ./bubble.sh  
Enter array elements separated by space:  
20 40 10 50 30  
Sorted array:  
10 20 30 40 50  
srikan@528014aa64d7583:~$ ./bubble.sh  
Enter array elements separated by space:  
4 9 1 10 3 8 2 6 5 7  
Sorted array:  
1 2 3 4 5 6 7 8 9 10  
srikan@528014aa64d7583:~$
```

6. Pascal Triangle

CODE

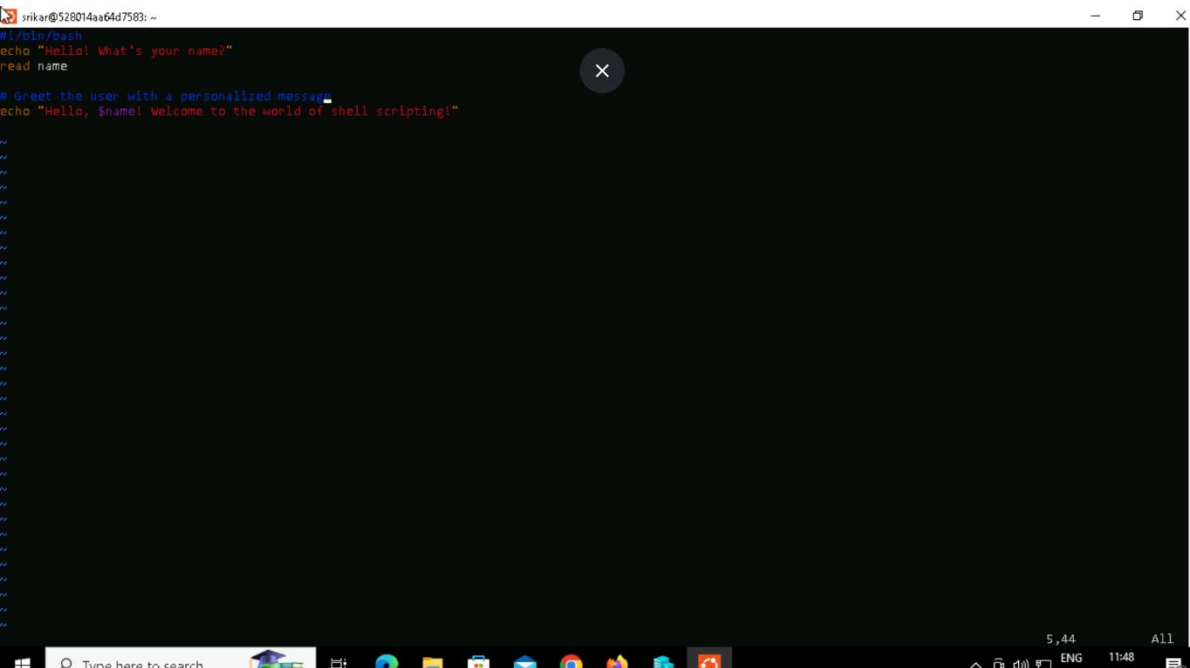
```
#!/bin/bash  
pastr1() {  
  r=$1  
  c=1  
  for ((l = 0; l < r; l++)); do  
    for ((s = 1; s < r - l; s++)); do  
      echo -n " "  
    done  
    for ((j = 0; j <= l; j++)); do  
      if [ $j -eq 0 -o $l -eq 0 ]; then  
        c=1  
      else  
        c=$((c * (l - j + 1) / j))  
      fi  
      echo -n "$c "  
    done  
    echo  
  done  
  echo "Enter the number of rows:"  
  read r  
  pastr1 $r  
}
```

OUTPUT



```
snikar@528014aa64d7583: ~  
snikar@528014aa64d7583:~$ vi pascal.sh  
snikar@528014aa64d7583:~$ chmod +x pascal.sh  
snikar@528014aa64d7583:~$ ./pascal.sh  
./pascal.sh: line 28: unexpected EOF while looking for matching `''  
snikar@528014aa64d7583:~$ vi pascal.sh  
snikar@528014aa64d7583:~$ ./pascal.sh  
Enter the number of rows:  
5  
      1  
     11  
    121  
   1331  
  14641  
snikar@528014aa64d7583:~$
```

7. Personalized name Program

CODE

The screenshot displays a Windows 10 desktop environment. A terminal window is open, showing a shell script being executed. The script prompts the user for their name and displays a personalized welcome message. The taskbar at the bottom shows various application icons and the system clock.

Terminal Window Content:

```
snikar@528014aa64d7583: ~
#!/bin/bash
echo "Hello! What's your name?"
read name

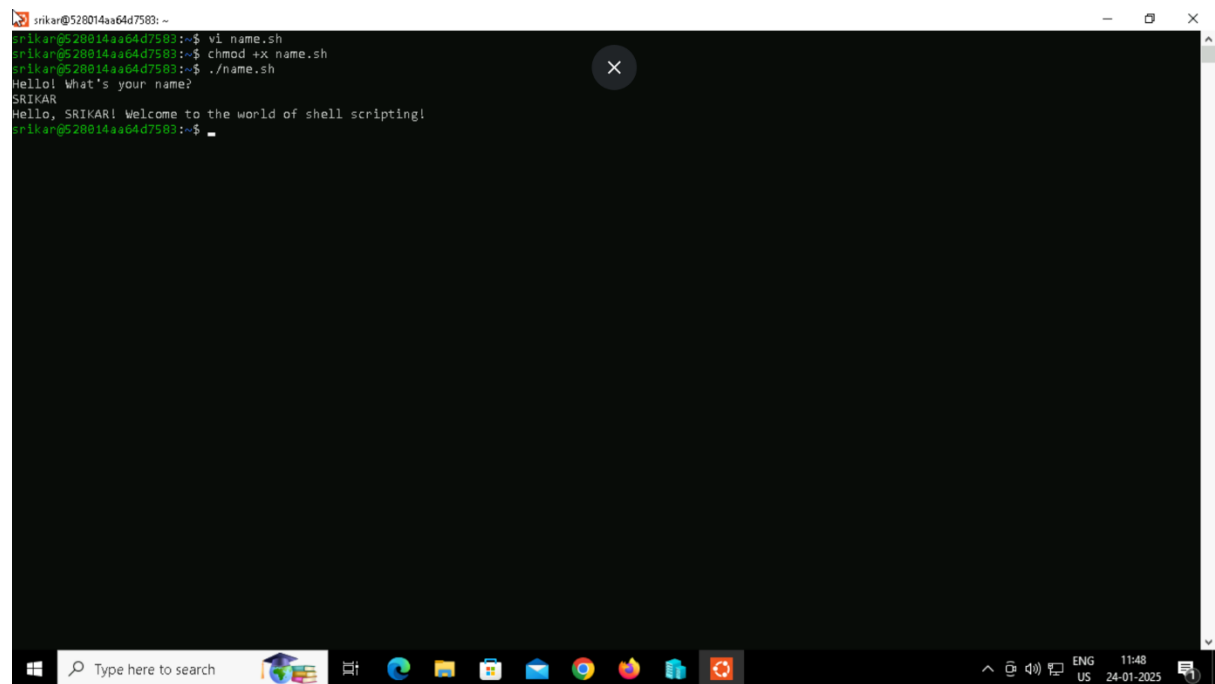
# Greet the user with a personalised message
echo "Hello, $name! Welcome to the world of shell scripting!"
```

Taskbar and System Tray:

- Search bar: "Type here to search"
- Taskbar icons: File Explorer, Microsoft Edge, Google Chrome, Firefox, and others.
- System tray: Network status, volume, and system clock showing 11:48 on 24-01-2025.

OUTPUT

```
srikar@528014aa64d7583: ~  
srikar@528014aa64d7583:~$ vi name.sh  
srikar@528014aa64d7583:~$ chmod +x name.sh  
srikar@528014aa64d7583:~$ ./name.sh  
Hello! What's your name?  
SRIKAR  
Hello, SRIKAR! Welcome to the world of shell scripting!  
srikar@528014aa64d7583:~$
```

A screenshot of a terminal window running on a Linux system. The terminal shows the execution of a shell script named 'name.sh'. The user 'srikar' is at the prompt. The script prompts for a name, and the user enters 'SRIKAR'. The script then prints a welcome message. The terminal window is titled 'srikar@528014aa64d7583: ~' and has standard window controls. The background is black, and the text is green. A Windows taskbar is visible at the bottom of the image.