

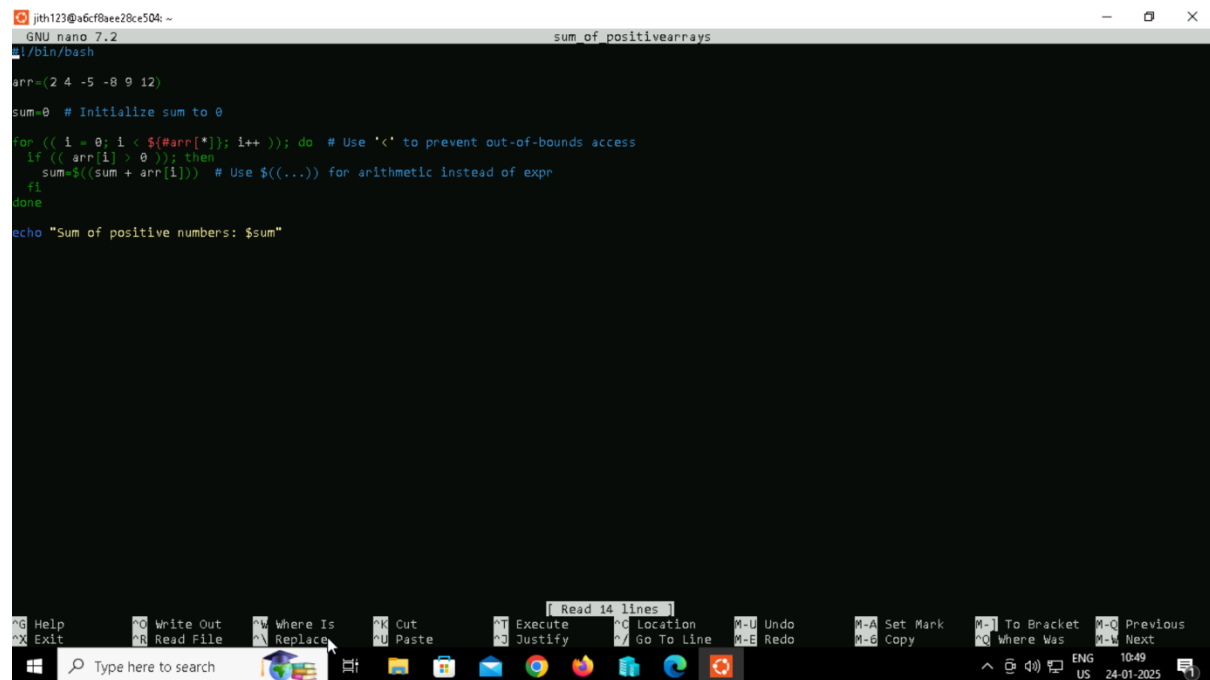
# ALL FILES EXECUTED IN SHELL:

```
jith123@a6cf8aee28ce504: ~  
Bubble_sort arithmetic_operations palindrome pascal reverse_number sum_of_positivearrays  
jith123@a6cf8aee28ce504:~$ ./Bubblesort  
-bash: ./Bubblesort: No such file or directory  
jith123@a6cf8aee28ce504:~$ ./Bubble_sort  
-bash: ./Bubble_sort: No such file or directory  
jith123@a6cf8aee28ce504:~$ ./Bubble_sort  
Entered array:  
10 8 20 100 12  
Sorted array:  
8 10 12 20 100  
jith123@a6cf8aee28ce504:~$ ./arithmetic_operations  
-bash: ./arithmetic_operations: No such file or directory  
jith123@a6cf8aee28ce504:~$ ./arithmetic_operations  
-bash: ./arithmetic_operations: No such file or directory  
jith123@a6cf8aee28ce504:~$ ./arithmetic_operations  
Input1 : 34  
Input2 : 32  
BC Value: 66  
EXPR Value: 66  
jith123@a6cf8aee28ce504:~$ ./palindrome  
Enter the number:  
234  
432  
The number is not a palindrome number!  
jith123@a6cf8aee28ce504:~$ ./pascal  
enter the number of rows:  
4  
1  
1 2 1  
1 3 3 1  
jith123@a6cf8aee28ce504:~$ ./reverse_number  
./reverse_number: line 8: [: -gt: unary operator expected  
Reverse Number is 0  
jith123@a6cf8aee28ce504:~$ ./reverse_number123  
-bash: ./reverse_number123: No such file or directory  
jith123@a6cf8aee28ce504:~$ ./reverse_number 123  
Reverse Number is 321  
jith123@a6cf8aee28ce504:~$ ./sum_of_positivearrays  
Sum of positive numbers: 27  
jith123@a6cf8aee28ce504:~$
```

## BUBBLE SORTING :

```
jith123@a6cf8aee28ce504: ~  
GNU nano 7.2 Bubble sort  
Bubble sort in shell with static array  
  
declare -a arr  
arr=(10 8 20 100 12)  
  
echo "Entered array:"  
echo ${arr[@]}  
  
for ((i = 0; i < 5; i++))  
do  
    for ((j = 0; j < 5-i-1; j++))  
    do  
        if [ ${arr[j]} -gt ${arr[j+1]} ];  
        then  
            temp=${arr[j]}  
            arr[j]=${arr[j+1]}  
            arr[j+1]=$temp  
        fi  
    done  
done  
echo "Sorted array:"  
echo ${arr[@]}
```

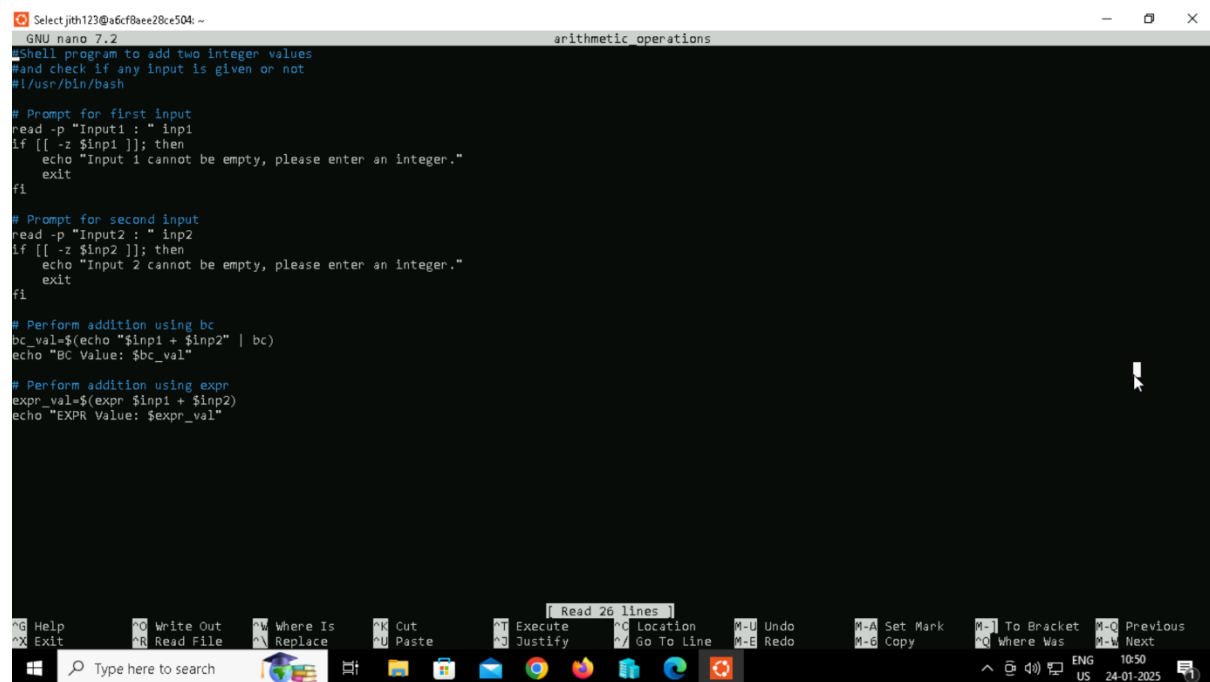
## SUM OF POSITIVE INTEGERS:



The screenshot shows a terminal window titled "sum\_of\_positivearrays" running GNU nano 7.2. The script defines an array `arr=(2 4 -5 -8 9 12)`, initializes `sum=0`, and uses a `for` loop to iterate over the array. Inside the loop, it checks if the current element is greater than 0 and adds it to the sum. The script ends with an `echo` statement to display the sum. The terminal output shows the sum of positive numbers as 14. The Windows taskbar is visible at the bottom.

```
jith123@a6cf8aee28ce504: ~  
GNU nano 7.2  
#!/bin/bash  
  
arr=(2 4 -5 -8 9 12)  
  
sum=0 # Initialize sum to 0  
  
for (( i = 0; i < ${#arr[*]}; i++ )); do # Use 'c' to prevent out-of-bounds access  
    if (( arr[i] > 0 )); then  
        sum=$((sum + arr[i])) # Use $((...)) for arithmetic instead of expr  
    fi  
done  
  
echo "Sum of positive numbers: $sum"
```

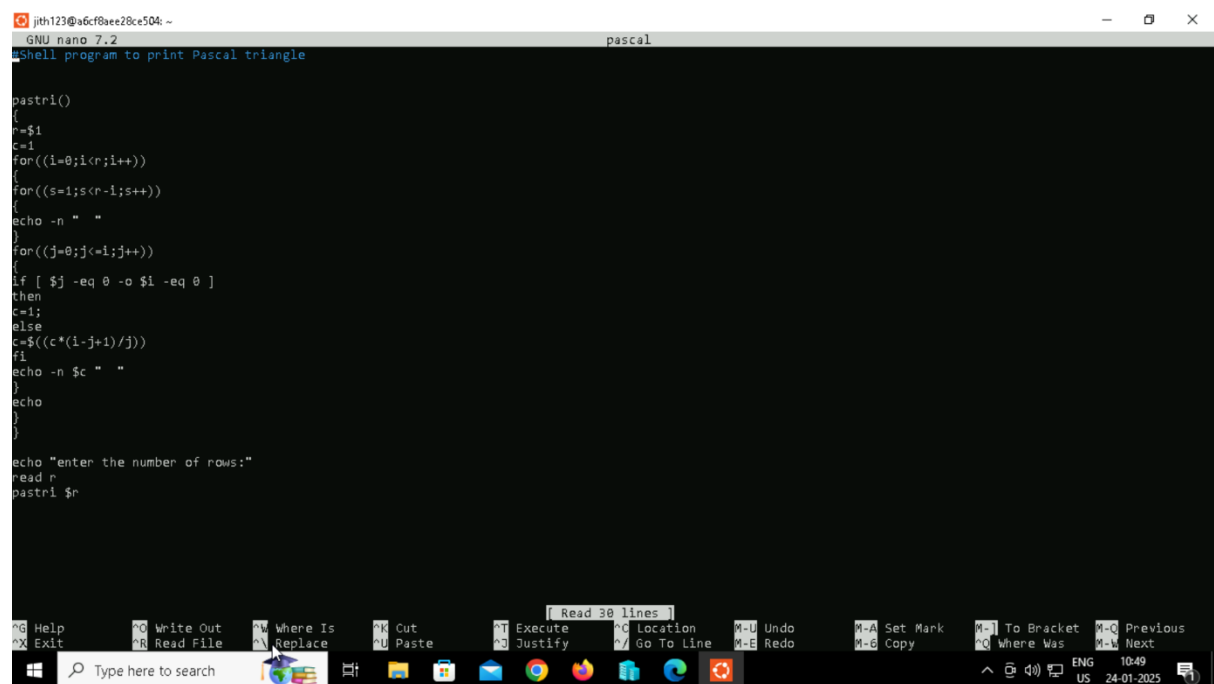
## SUM OF TWO INTEGERS:



The screenshot shows a terminal window titled "arithmetic\_operations" running GNU nano 7.2. The script prompts the user for two integers, `inp1` and `inp2`. It uses `bc` and `expr` to calculate the sum of the two integers and displays the result. The terminal output shows the sum of 12 and 4 as 16. The Windows taskbar is visible at the bottom.

```
Select jith123@a6cf8aee28ce504: ~  
GNU nano 7.2  
#Shell program to add two integer values  
#and check if any input is given or not  
#!/usr/bin/bash  
  
# Prompt for first input  
read -p "Input1 : " inp1  
if [[ -z $inp1 ]]; then  
    echo "Input 1 cannot be empty, please enter an integer."  
    exit  
fi  
  
# Prompt for second input  
read -p "Input2 : " inp2  
if [[ -z $inp2 ]]; then  
    echo "Input 2 cannot be empty, please enter an integer."  
    exit  
fi  
  
# Perform addition using bc  
bc_val=$(echo "$inp1 + $inp2" | bc)  
echo "BC Value: $bc_val"  
  
# Perform addition using expr  
expr_val=$(expr $inp1 + $inp2)  
echo "EXPR Value: $expr_val"
```

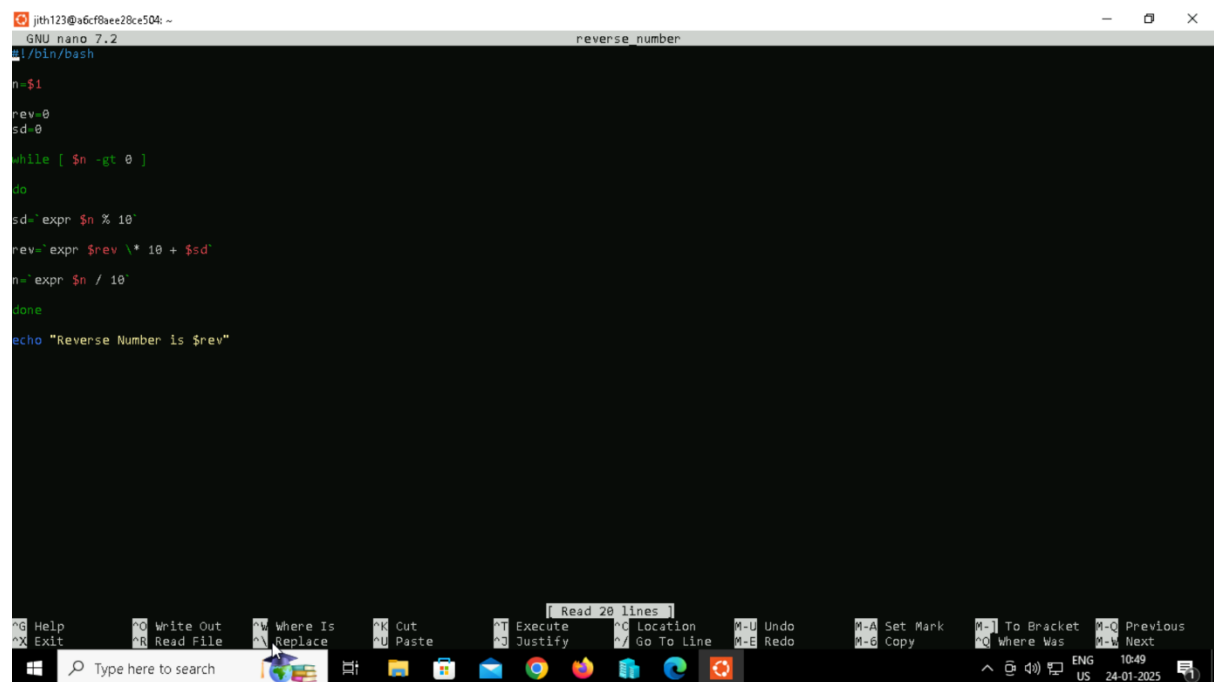
## PASCAL:



The screenshot shows a terminal window with the nano 7.2 editor open. The file is named 'pascal'. The code is a shell script to print Pascal's triangle. It takes a row number 'r' as input and prints the triangle for that row. The code uses nested loops to calculate the values of the triangle and prints them with spaces for formatting. The terminal window has a Windows taskbar at the bottom with various icons and a search bar.

```
jith123@a6cf8aee28ce504: ~  
GNU nano 7.2  
#Shell program to print Pascal triangle  
  
pastr1()  
{  
r=$1  
c=1  
for((i=0;i<r;i++))  
{  
for((s=1;s<r-1;s++))  
{  
echo -n " "  
}  
for((j=0;j<=i;j++))  
{  
if [ $j -eq 0 -o $i -eq 0 ]  
then  
c=1;  
else  
c=$((c*(i-j+1)/j))  
fi  
echo -n $c " "  
}  
echo  
}  
}  
  
echo "enter the number of rows:"  
read r  
pastr1 $r
```

## REVERSE NUMBER:

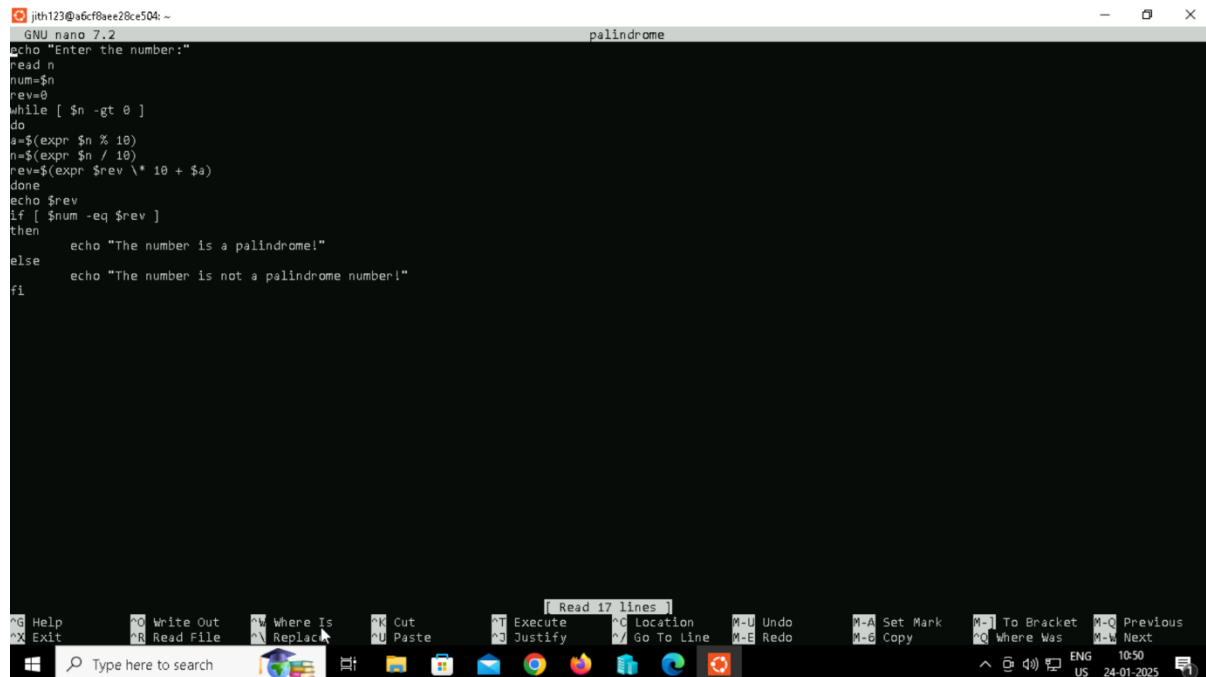


The screenshot shows a terminal window with the nano 7.2 editor open. The file is named 'reverse number'. The code is a shell script to reverse a number. It takes a number 'n' as input and uses a while loop to extract the last digit, calculate the reversed number, and then divide the original number by 10. The final reversed number is printed. The terminal window has a Windows taskbar at the bottom with various icons and a search bar.

```
jith123@a6cf8aee28ce504: ~  
GNU nano 7.2  
#!/bin/bash  
  
n=$1  
rev=0  
sd=0  
  
while [ $n -gt 0 ]  
do  
sd=$((expr $n % 10))  
rev=$((expr $rev * 10 + $sd))  
n=$((expr $n / 10))  
done  
  
echo "Reverse Number is $rev"
```

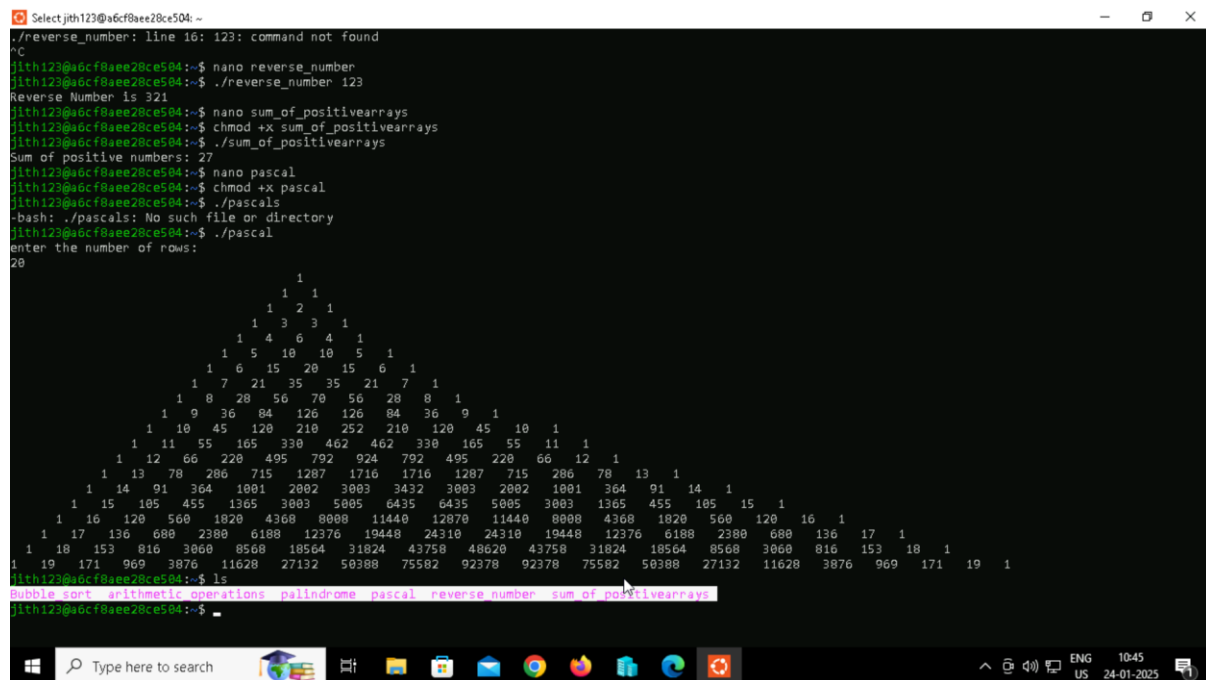
## PALINDROME:

```
jith123@a6cf8aee28ce504: ~
GNU nano 7.2                                palindrome
echo "Enter the number:"
read n
num=$n
rev=0
while [ $n -gt 0 ]
do
a=$(expr $n % 10)
n=$(expr $n / 10)
rev=$(expr $rev \* 10 + $a)
done
echo $rev
if [ $num -eq $rev ]
then
    echo "The number is a palindrome!"
else
    echo "The number is not a palindrome number!"
fi
```



## REPRESENTING ALL FILES USING ls COMMAND:

```
Select jith123@a6cf8aee28ce504: ~
./reverse_number: line 16: 123: command not found
^C
jith123@a6cf8aee28ce504:~$ nano reverse_number
jith123@a6cf8aee28ce504:~$ ./reverse_number 123
Reverse Number is 321
jith123@a6cf8aee28ce504:~$ nano sum_of_positivearrays
jith123@a6cf8aee28ce504:~$ chmod +x sum_of_positivearrays
jith123@a6cf8aee28ce504:~$ ./sum_of_positivearrays
Sum of positive numbers: 27
jith123@a6cf8aee28ce504:~$ nano pascal
jith123@a6cf8aee28ce504:~$ chmod +x pascal
jith123@a6cf8aee28ce504:~$ ./pascals
-bash: ./pascals: No such file or directory
jith123@a6cf8aee28ce504:~$ ./pascal
enter the number of rows:
20
      1
     1 1
    1 2 1
   1 3 3 1
  1 4 6 4 1
 1 5 10 10 5 1
1 6 15 20 15 6 1
1 7 21 35 35 21 7 1
1 8 28 56 70 56 28 8 1
1 9 36 84 126 126 84 36 9 1
1 10 45 120 210 252 210 120 45 10 1
1 11 55 165 330 462 462 330 165 55 11 1
1 12 66 220 495 792 924 792 495 220 66 12 1
1 13 78 286 715 1287 1716 1716 1287 715 286 13 1
1 14 91 364 1001 2002 3003 3432 3003 2002 1001 364 91 14 1
1 15 105 455 1365 3003 5005 6435 6435 5005 3003 1365 455 105 15 1
1 16 120 560 1820 4368 8008 11440 12870 11440 8008 4368 1820 560 120 16 1
1 17 136 680 2380 6188 12376 19448 24310 19448 12376 6188 2380 680 136 17 1
1 18 153 816 3060 8568 18564 31824 43758 48620 43758 31824 18564 8568 3060 816 153 18 1
1 19 171 969 3876 11628 27132 50388 75582 92378 92378 75582 50388 27132 11628 3876 969 171 19 1
jith123@a6cf8aee28ce504:~$ ls
bubble sort  arithmetic operations  palindrome  pascal  reverse number  sum of positive arrays
```



Greeting user:

```
GNU nano 7.2                                greet
#!/bin/bash

# Prompt the user for their name
echo "Hello! What's your name?"
read name

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

Output:

```
jithu123@bcf0aee28ce504:~$ nano greet
jithu123@bcf0aee28ce504:~$ nano greet
jithu123@bcf0aee28ce504:~$ chmod +x greet
jithu123@bcf0aee28ce504:~$ ./greet
Hello! What's your name?
jithu
Hello, jithu! Welcome to the world of shell scripting!
jithu123@bcf0aee28ce504:~$
```