Write a script that takes a filename as an argument and prints the number of lines, words, and characters in the file.

```
[root@mohamedsalah ~]# cat file.sh
#!/bin/bash

read -p "enter your file name -" filename
# Count the number of lines, words,  in the file
lines=$(cat "$filename" | wc -l)
words=$(cat "$filename" | wc -w)

# Print the results
echo "File: $filename"
echo "Number of lines: $lines"
echo "Number of words: $words"

[root@mohamedsalah ~]# cat test-file.txt
mohamed salah
sebaei
[root@mohamedsalah ~]# ./file.sh
enter your file name -test-file.txt
File: test-file.txt
Number of lines: 2
Number of words: 3
[root@mohamedsalah ~]#
```

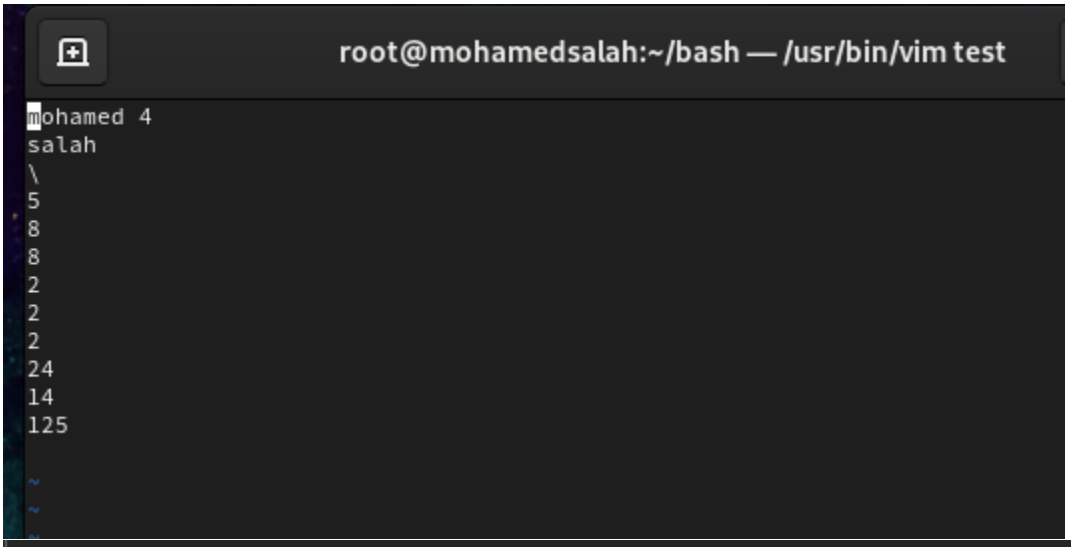
write a script that takes a string as an argument and prints the string in reverse order.

```
[root@mohamedsalah bash]# cat reverse.sh
#!/bin/bash

# Check if the user provided a string argument
if [ $# -ne 1 ]; then
    echo "Usage: $0 <string>"
    exit 1
fi

input_string="$1"

# Use the 'rev'
reversed_string=$(echo "$input_string" | rev)

echo "Original : $input_string"
echo "Reversed : $reversed_string"

[root@mohamedsalah bash]# ./reverse.sh "mohamed, salah!"
Original : mohamed, salah!
Reversed : !halas ,demahom
[root@mohamedsalah bash]#
```

Write a script that takes a directory as an argument and prints the number of files and directories in the directory.

```
[root@mohamedsalah bash]# ./count-file-dir.sh /root
Number of files in /root: 10
Number of directories in /root: 9
[root@mohamedsalah bash]# cat count-file-dir.sh
#!/bin/bash

if [ $# -ne 1 ]; then
  echo "Usage: $0 <directory>"
  exit 1
fi

target_directory="$1"

if [ ! -d "$target_directory" ]; then
  echo "$target_directory is not a valid directory."
  exit 1
fi

num_files=0
num_directories=0

# Loop through the contents of the directory
for entry in "$target_directory"/*; do
  if [ -f "$entry" ]; then
    ((num_files++))
  elif [ -d "$entry" ]; then
    ((num_directories++))
  fi
done

echo "Number of files in $target_directory: $num_files"
echo "Number of directories in $target_directory: $num_directories"

[root@mohamedsalah bash]#
```

Write a script that takes a filename as an argument and prints the contents of the file in reverse order.




```
[root@mohamedsalah bash]# ./reverse-file.sh test

125
14
24
2
2
2
8
8
5
\
salah
mohamed 4
[root@mohamedsalah bash]# cat reverse-file.sh
#!/bin/bash

if [ $# -ne 1 ]; then
  echo "Usage: $0 <filename>"
  exit 1
fi

filename="$1"

# Check if the file exists
if [ ! -e "$filename" ]; then
  echo "$filename does not exist."
  exit 1
fi

# Read the file line by line into an array
mapfile -t lines < "$filename"

# Print the lines in reverse order
for ((i=${#lines[@]}-1; i>=0; i--)); do
  echo "${lines[i]}"
done
```

Write a script that takes a number as an argument and checks if it is even or odd.

```
[root@mohamedsalah bash]# vi even-odd.sh
[root@mohamedsalah bash]# chmod +x even-odd.sh
[root@mohamedsalah bash]# ./even-odd.sh 5
5 is odd.
[root@mohamedsalah bash]# ./even-odd.sh 4
4 is even.
[root@mohamedsalah bash]# cat even-odd.sh
#!/bin/bash

# Check if an argument is provided
if [ $# -ne 1 ]; then
  echo "Usage: $0 <number>"
  exit 1
fi

number="$1"

# Check if the number is an integer
if ! [[ "$number" =~ ^[0-9]+$ ]]; then
  echo "$number is not a valid integer."
  exit 1
fi

# Check if the number is even or odd
if ((number % 2 == 0)); then
  echo "$number is even."
else
  echo "$number is odd."
fi

[root@mohamedsalah bash]#
```

Write a script that takes a list of numbers as arguments and finds the largest and smallest numbers in the list.

```
[root@mohamedsalah bash]# vi largest-smallest.sh
[root@mohamedsalah bash]# chmod +x largest-smallest.sh
[root@mohamedsalah bash]# ./largest-smallest.sh
Usage: ./largest-smallest.sh <number1> <number2> ...
[root@mohamedsalah bash]# ./largest-smallest.sh  5 6 8 7 1 0
Largest number: 8
Smallest number: 0
[root@mohamedsalah bash]# cat largest-smallest.sh
#!/bin/bash

# Check if at least one argument is provided
if [ $# -lt 1 ]; then
  echo "Usage: $0 <number1> <number2> ..."
  exit 1
fi

# Use the 'sort' command to sort the numbers
sorted_numbers=($(printf "%s\n" "$@" | sort -n))

smallest="${sorted_numbers[0]}"

largest="${sorted_numbers[${#sorted_numbers[@]}-1]}"

echo "Largest number: $largest"
echo "Smallest number: $smallest"

[root@mohamedsalah bash]#
```