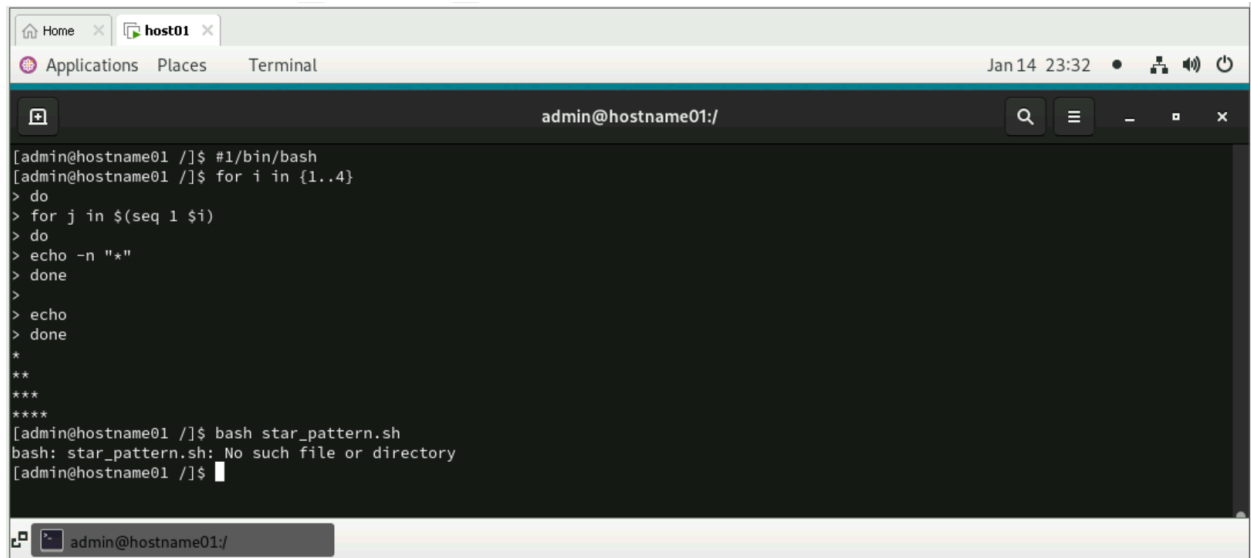


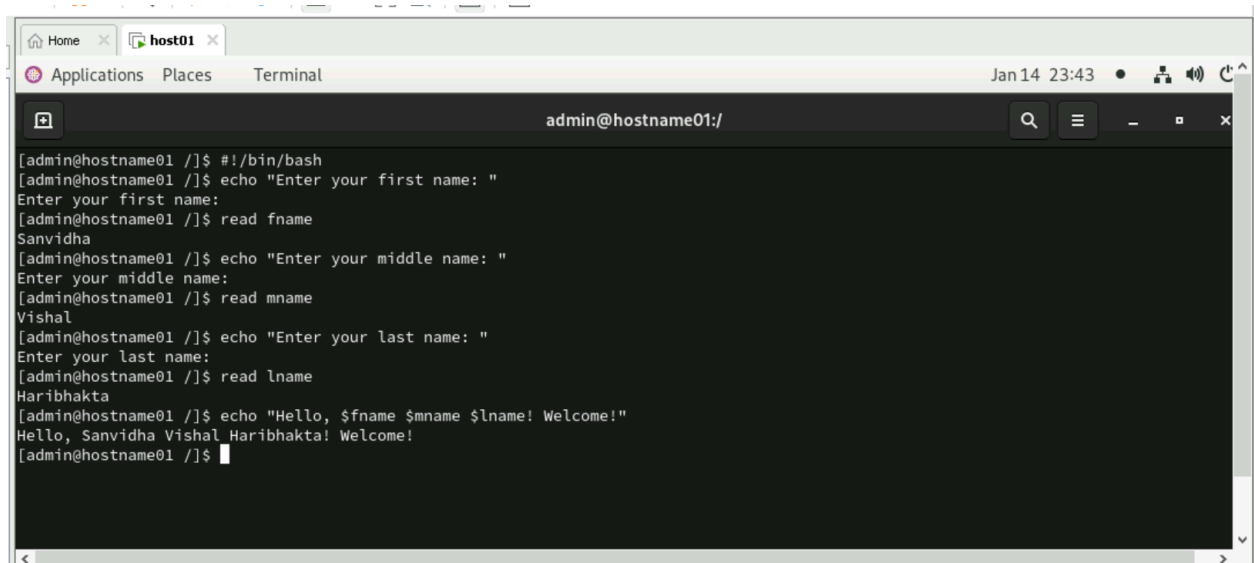
- 1 Write a shell script which will generate the O/P as follows

```
*
**
***
****
```



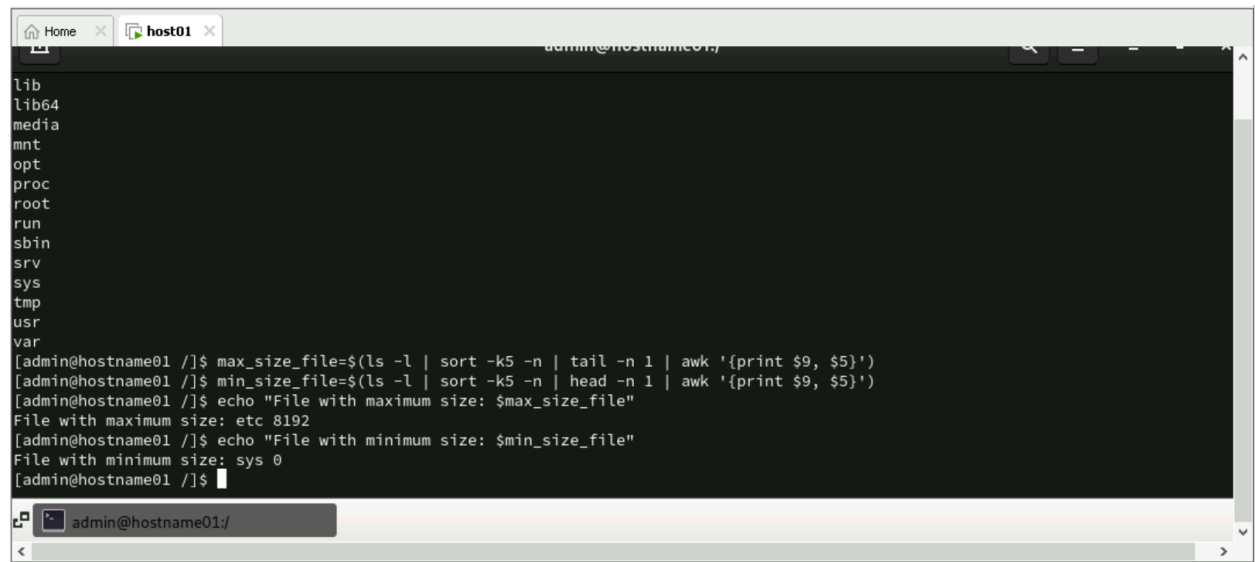
```
admin@hostname01 /]$ #!/bin/bash
admin@hostname01 /]$ for i in {1..4}
> do
> for j in $(seq 1 $i)
> do
> echo -n "*"
> done
> echo
> done
*
**
***
****
admin@hostname01 /]$ bash star_pattern.sh
bash: star_pattern.sh: No such file or directory
admin@hostname01 /]$
```

- 2 Accept the first name, middle name, and last name of a person in variables fname, mname and lname respectively. Greet the person (take his full name) using appropriate message.



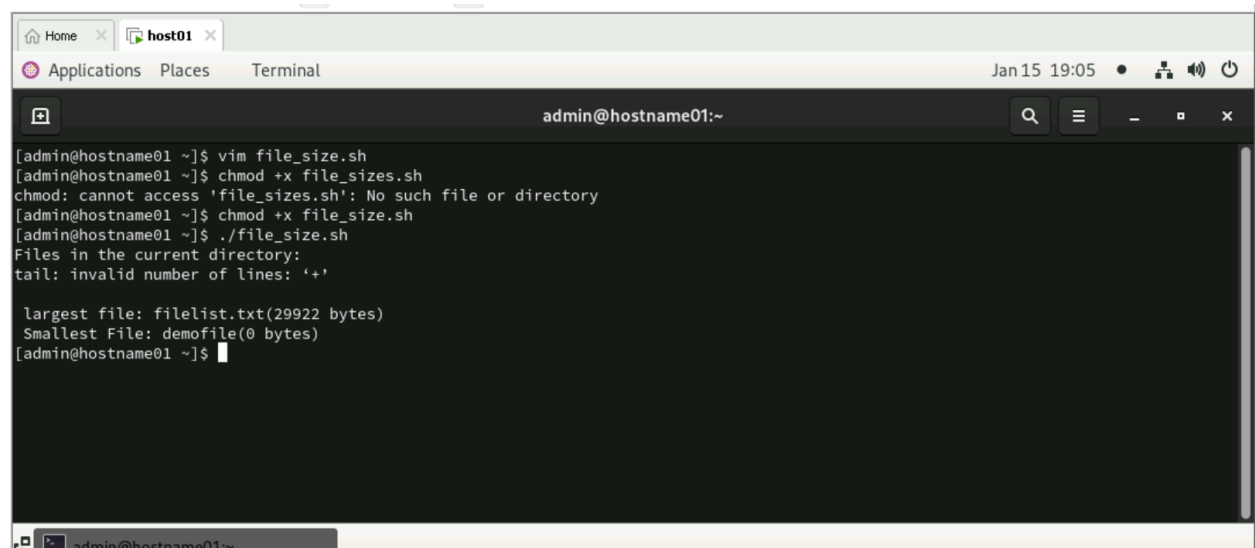
```
admin@hostname01 /]$ #!/bin/bash
admin@hostname01 /]$ echo "Enter your first name: "
Enter your first name:
admin@hostname01 /]$ read fname
Sanvidha
admin@hostname01 /]$ echo "Enter your middle name: "
Enter your middle name:
admin@hostname01 /]$ read mname
Vishal
admin@hostname01 /]$ echo "Enter your last name: "
Enter your last name:
admin@hostname01 /]$ read lname
Haribhakta
admin@hostname01 /]$ echo "Hello, $fname $mname $lname! Welcome!"
Hello, Sanvidha Vishal Haribhakta! Welcome!
admin@hostname01 /]$
```

- 3 Display the name of files in the current directory along with the names of files with maximum & minimum size. The file size is considered in bytes.



A terminal window titled 'host01' showing a directory listing of system folders: lib, lib64, media, mnt, opt, proc, root, run, sbin, srv, sys, tmp, usr, var. Below the listing, several shell commands are executed to find the maximum and minimum file sizes in the current directory. The commands use 'ls -l', 'sort -k5 -n', 'tail -n 1', 'head -n 1', and 'awk' to process the output. The results show the maximum file size is 8192 bytes (for 'etc') and the minimum is 0 bytes (for 'sys').

```
lib
lib64
media
mnt
opt
proc
root
run
sbin
srv
sys
tmp
usr
var
[admin@hostname01 ~]$ max_size_file=$(ls -l | sort -k5 -n | tail -n 1 | awk '{print $9, $5}')
[admin@hostname01 ~]$ min_size_file=$(ls -l | sort -k5 -n | head -n 1 | awk '{print $9, $5}')
[admin@hostname01 ~]$ echo "File with maximum size: $max_size_file"
File with maximum size: etc 8192
[admin@hostname01 ~]$ echo "File with minimum size: $min_size_file"
File with minimum size: sys 0
[admin@hostname01 ~]$
```

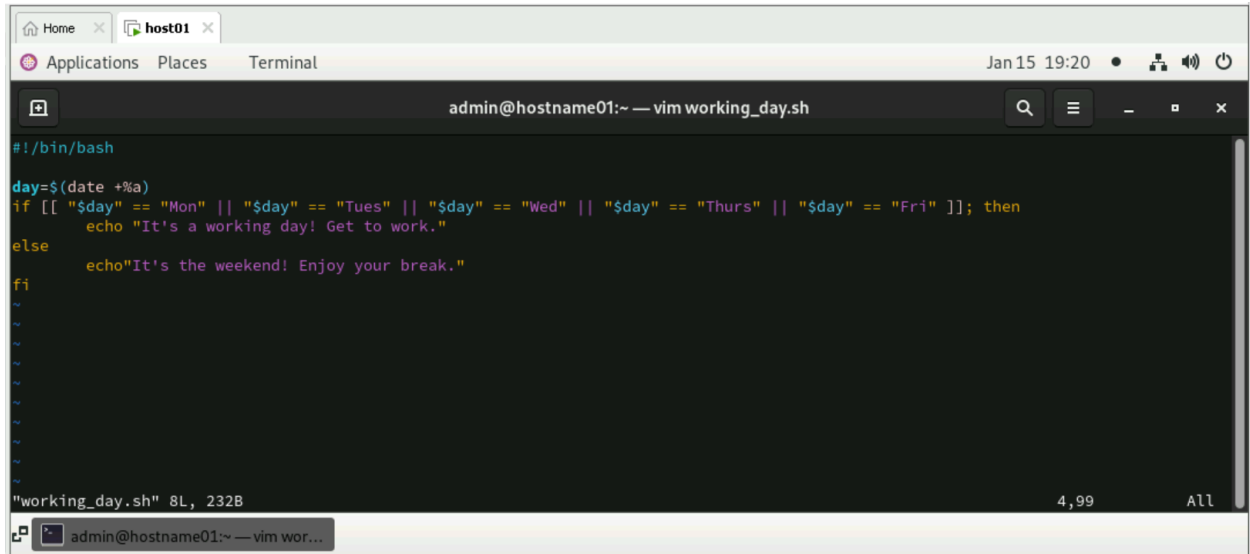


A terminal window titled 'host01' showing the execution of a script named 'file_size.sh'. The user attempts to run the script using 'vim', 'chmod +x', and './file_size.sh'. The script execution results in several error messages: 'chmod: cannot access 'file_sizes.sh': No such file or directory', 'Files in the current directory:', and 'tail: invalid number of lines: '+''. The script then outputs the largest file as 'filelist.txt(29922 bytes)' and the smallest file as 'demofile(0 bytes)'. The terminal window also shows the system date and time as 'Jan 15 19:05'.

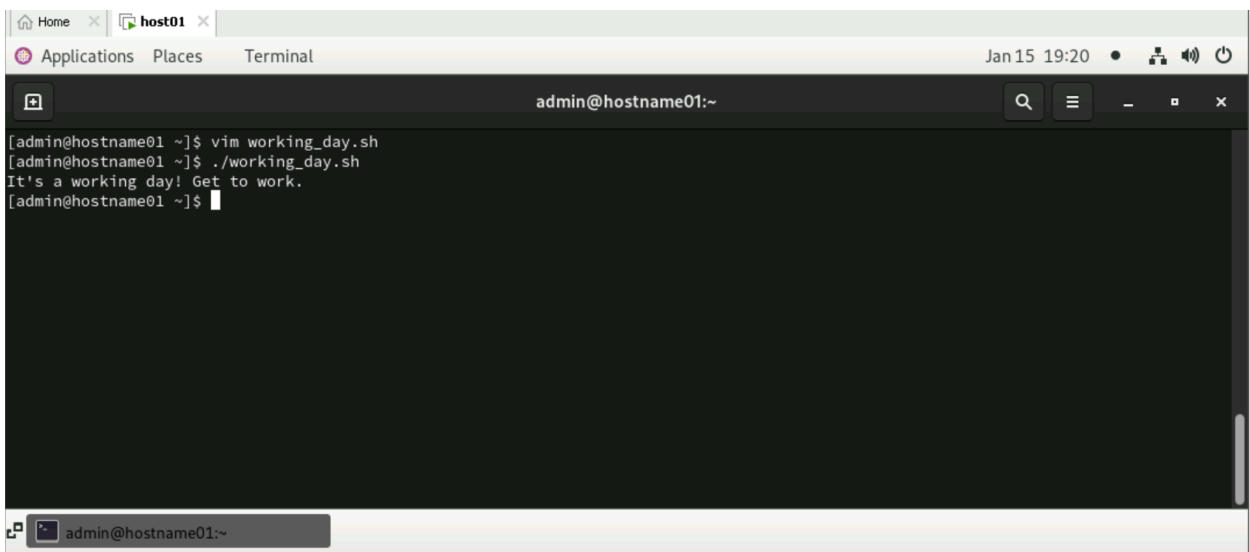
```
[admin@hostname01 ~]$ vim file_size.sh
[admin@hostname01 ~]$ chmod +x file_sizes.sh
chmod: cannot access 'file_sizes.sh': No such file or directory
[admin@hostname01 ~]$ chmod +x file_size.sh
[admin@hostname01 ~]$ ./file_size.sh
Files in the current directory:
tail: invalid number of lines: '+'

largest file: filelist.txt(29922 bytes)
Smallest File: demofile(0 bytes)
[admin@hostname01 ~]$
```

- 4 Write a script which when executed checks out whether it is a working day or not?
(Note: Working day Mon-Fri)



```
admin@hostname01:~ — vim working_day.sh
#!/bin/bash
day=$(date +%a)
if [[ "$day" == "Mon" || "$day" == "Tues" || "$day" == "Wed" || "$day" == "Thurs" || "$day" == "Fri" ]]; then
    echo "It's a working day! Get to work."
else
    echo "It's the weekend! Enjoy your break."
fi
"working_day.sh" 8L, 232B
```



```
admin@hostname01:~$ vim working_day.sh
admin@hostname01:~$ ./working_day.sh
It's a working day! Get to work.
admin@hostname01:~$
```

- 5 Write a script that accepts a member into HP health club, if the weight of the person is within the range of 30-250 Kgs.

```
#!/bin/bash
```

```
# Accept weight from the user
```

```
echo -n "Enter your weight (in Kgs): "
```

```
read weight
```

```
# Validate if input is a number
```

```
if [[ ! $weight =~ ^[0-9]+$ ]]; then
```

```
    echo "Invalid input. Please enter a numeric value for weight."
```

```

    exit 1
fi

# Check if the weight is within the valid range
if [[ $weight -ge 30 && $weight -le 250 ]]; then
    echo "You are eligible to join the HP Health Club."
else
    echo " Your weight is not within the eligible range (30-250 Kgs)."
fi

```

- 6 Write a shell script that greets the user with an appropriate message depending on the system time.

```
#!/bin/bash
```

```
hour=$(date +%H)
```

```

if [[ $hour -ge 0 && $hour -lt 12 ]]; then
    echo "Good Morning!"
elif [[ $hour -ge 12 && $hour -lt 17 ]]; then
    echo "Good Afternoon!"
else
    echo "Good Evening!"
fi

```

- 7 A data file file has some student records including rollno, names and subject marks. The fields are separated by a “.”. Write a shell script that accepts roll number from the user, searches it in the file and if the roll number is present - allows the user to modify name and marks in 3 subjects.
If the roll number is not present, display a message “Roll No Not Found”. Allow the user to modify one record at a time.

```
#!/bin/bash
```

```
hour=$(date +%H)
```

```
if [[ $hour -ge 0 && $hour -lt 12 ]]; then  
    echo "Good Morning!"  
elif [[ $hour -ge 12 && $hour -lt 17 ]]; then  
    echo "Good Afternoon!"  
else  
    echo "Good Evening!"  
fi
```

- 8 Modify program 7 to accept the RollNo from the command line.

```
#!/bin/bash
```

```
rollno=$1
```

```
grep -q "$rollno" file  
if [[ $? -eq 0 ]]; then  
    echo "Roll number found. You can now modify the record."  
    echo "Enter name: "  
    read name  
    echo "Enter marks for subject 1: "  
    read marks1  
    echo "Enter marks for subject 2: "  
    read marks2  
    echo "Enter marks for subject 3: "  
    read marks3
```

```
sed -i "s/^\$rollno:.*\/\$rollno:$name:$marks1:$marks2:$marks3/" file
```

```

    echo "Record updated successfully."
else
    echo "Roll No Not Found"
fi

```

- 9 Modify the program 7 to accept the RollNo and display the record and ask for delete confirmation. Once confirmed delete the record and update the data file.

```
#!/bin/bash
```

```

echo "Enter roll number to delete: "
read rollno

```

```

grep -q "$rollno" file
if [[ $? -eq 0 ]]; then
    echo "Roll number found. Do you want to delete the record? (yes/no)"
    read confirmation
    if [[ "$confirmation" == "yes" ]]; then

```

```

        sed -i "/^$rollno:/d" file
        echo "Record deleted successfully."
    else
        echo "Record not deleted."
    fi
else
    echo "Roll No Not Found"
fi

```

- 10 Write a script that takes a command line argument and reports on its file type (regular file, directory file, etc.). For more than one argument generate error message.

```
#!/bin/bash
```

```

if [ $# -eq 1 ]; then
    file=$1
    if [ -f "$file" ]; then
        echo "$file is a regular file."
    elif [ -d "$file" ]; then
        echo "$file is a directory."
    else
        echo "$file is of an unknown type."
    fi
else
    echo "Error: Please provide exactly one file name."
fi

```

- 11 Add some student records in the “student” file manually. The fields to be considered are “RollNo”, “Name”, “Marks_Hindi”, “Marks_Maths”, “Marks_Physics”.

Write a script which does the following

- a If the roll number already exists, then store the record and the following message “roll number exists” in a log file “log1”.
- b If the marks in the subjects is not in the range of 1 – 99 then store such a record followed by a message “marks out of range” in “log1”
- c If the data is valid, the calculate total, percentage, grade and display on the terminal

```
#!/bin/bash
```

```

# Accept student data
echo "Enter RollNo: "
read rollno
echo "Enter Name: "
read name
echo "Enter Marks for Hindi: "
read marks_hindi
echo "Enter Marks for Maths: "
read marks_maths
echo "Enter Marks for Physics: "
read marks_physics

```

```

# Validate the data
if [[ $(grep -q "$rollno" student) ]]; then
    echo "Roll number exists" >> log1

```

```

else
  if [[ $marks_hindi -lt 1 || $marks_hindi -gt 99 || $marks_maths -lt 1 || $marks_maths -gt 99 ||
$marks_physics -lt 1 || $marks_physics -gt 99 ]]; then
    echo "Marks out of range" >> log1
  else
    total=$((marks_hindi + marks_maths + marks_physics))
    percentage=$((total / 3))
    grade=""

    if [[ $percentage -ge 90 ]]; then
      grade="A"
    elif [[ $percentage -ge 75 ]]; then
      grade="B"
    elif [[ $percentage -ge 60 ]]; then
      grade="C"
    else
      grade="D"
    fi

    echo
    "$rollno:$name:$marks_hindi:$marks_maths:$marks_physics:$total:$percentage:$grade" >>
student
    echo "Student record added successfully."
  fi
fi

```