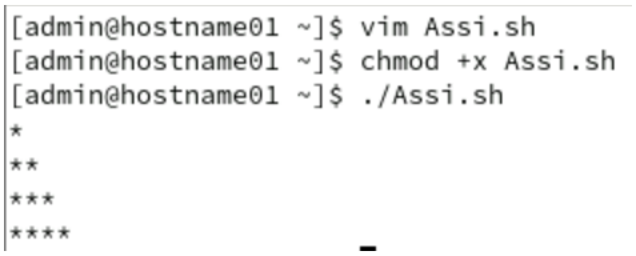
1 Write a shell script which will generate the O/P as follows

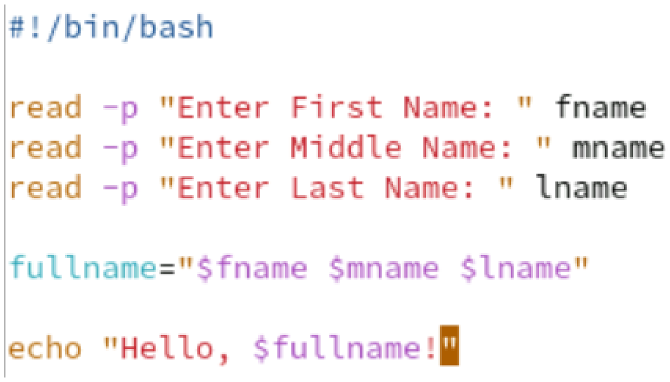
```
*
**
***
****
```

```
#!/bin/bash

for ((i=1; i<=4; i++)); do
  for ((j=1; j<=i; j++)); do
    echo -n "*"
  done
  echo ""
done
```

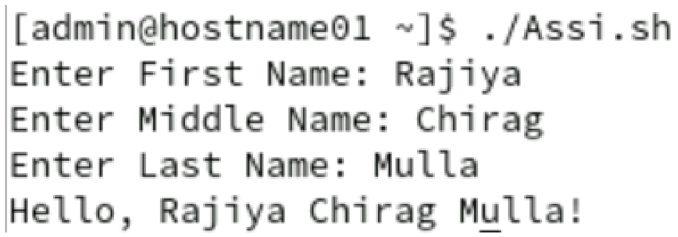


```
[admin@hostname01 ~]$ vim Assi.sh
[admin@hostname01 ~]$ chmod +x Assi.sh
[admin@hostname01 ~]$ ./Assi.sh
*
**
***
****
```

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.

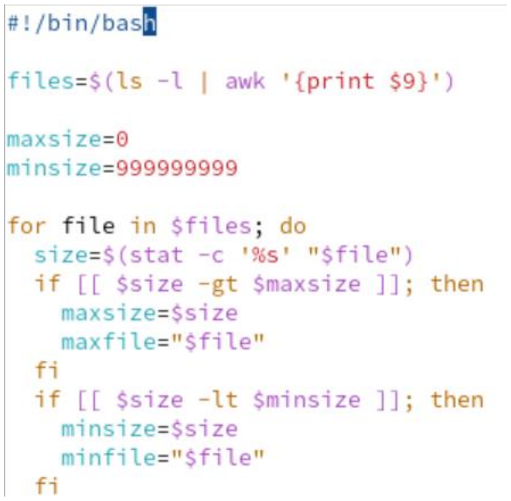


```
#!/bin/bash

read -p "Enter First Name: " fname
read -p "Enter Middle Name: " mname
read -p "Enter Last Name: " lname

fullname="$fname $mname $lname"

echo "Hello, $fullname!"
```

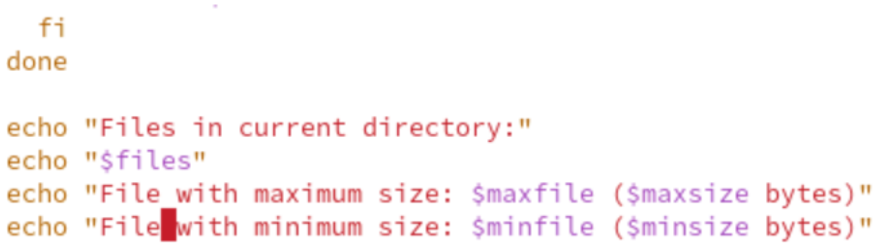


```
[admin@hostname01 ~]$ ./Assi.sh
Enter First Name: Rajiya
Enter Middle Name: Chirag
Enter Last Name: Mulla
Hello, Rajiya Chirag Mulla!
```

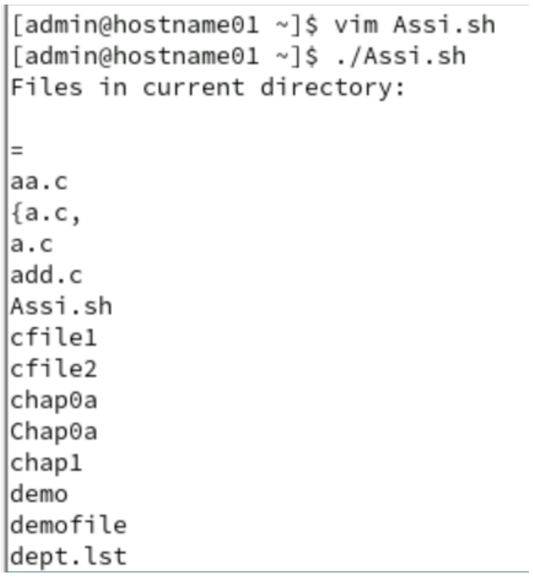
3 Display the name of files in the current directory along with the names of files with maximum & minimum size. The file size is considered in bytes.



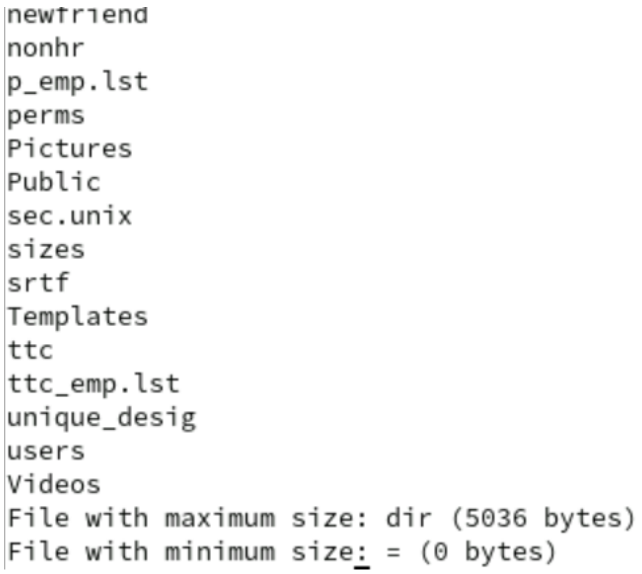
```
#!/bin/bash

files=$(ls -l | awk '{print $9}')

maxsize=0
minsize=999999999

for file in $files; do
  size=$(stat -c '%s' "$file")
  if [[ $size -gt $maxsize ]]; then
    maxsize=$size
    maxfile="$file"
  fi
  if [[ $size -lt $minsize ]]; then
    minsize=$size
    minfile="$file"
  fi
```



```
  fi
done

echo "Files in current directory:"
echo "$files"
echo "File with maximum size: $maxfile ($maxsize bytes)"
echo "File with minimum size: $minfile ($minsize bytes)"
```



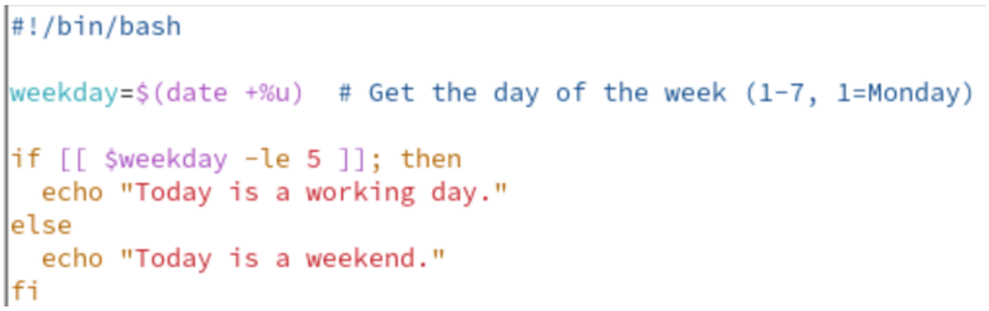
```
[admin@hostname01 ~]$ vim Assi.sh
[admin@hostname01 ~]$ ./Assi.sh
Files in current directory:

=
aa.c
{a.c,
a.c
add.c
Assi.sh
cfile1
cfile2
chap0a
Chap0a
chap1
demo
demofile
dept.lst
```



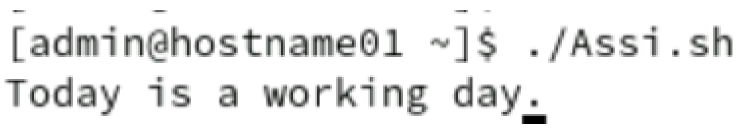
```
newfriend
nonhr
p_emp.lst
perms
Pictures
Public
sec.unix
sizes
srtf
Templates
ttc
ttc_emp.lst
unique_desig
users
Videos
File with maximum size: dir (5036 bytes)
File with minimum size: = (0 bytes)
```

4 Write a script which when executed checks out whether it is a working day or not?

(Note: Working day Mon-Fri)

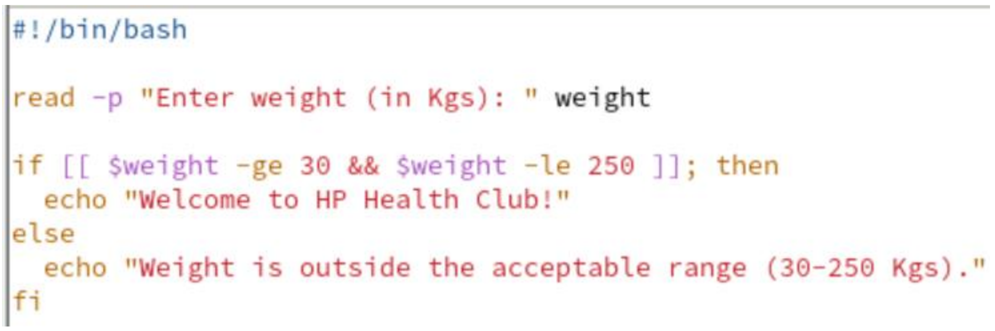


```
#!/bin/bash

weekday=$(date +%u)  # Get the day of the week (1-7, 1=Monday)

if [[ $weekday -le 5 ]]; then
  echo "Today is a working day."
else
  echo "Today is a weekend."
fi
```

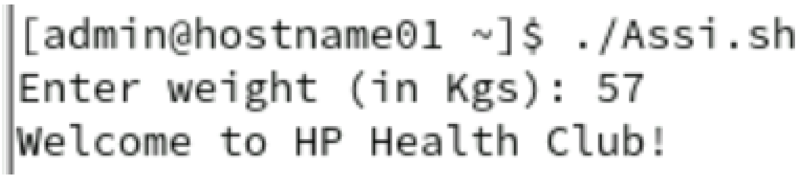


```
[admin@hostname01 ~]$ ./Assi.sh
Today is a working day.
```

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

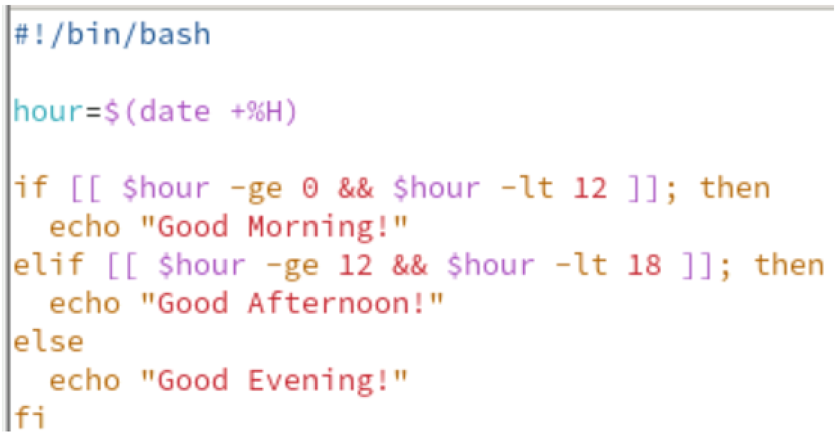


```
#!/bin/bash

read -p "Enter weight (in Kgs): " weight

if [[ $weight -ge 30 && $weight -le 250 ]]; then
  echo "Welcome to HP Health Club!"
else
  echo "Weight is outside the acceptable range (30-250 Kgs)."
fi
```

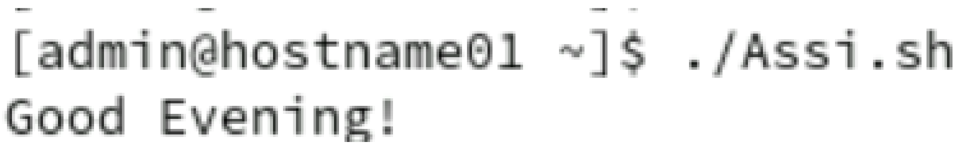


```
[admin@hostname01 ~]$ ./Assi.sh
Enter weight (in Kgs): 57
Welcome to HP Health Club!
```

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



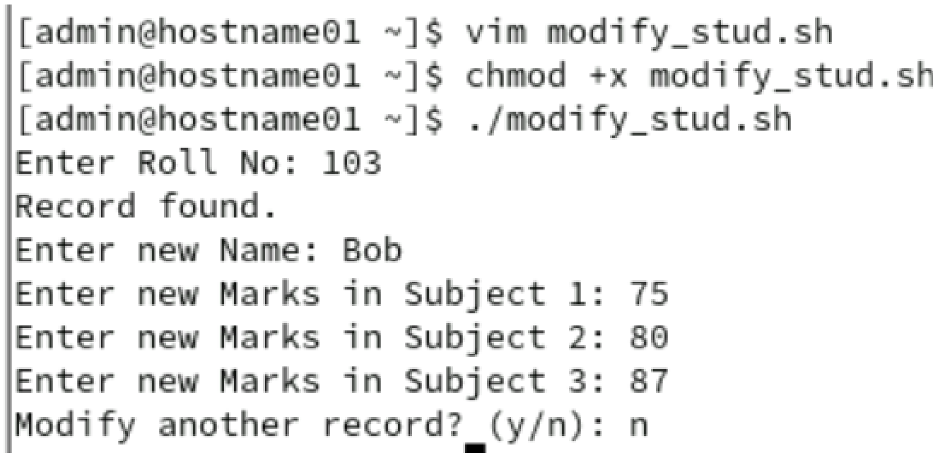
```
#!/bin/bash

hour=$(date +%H)

if [[ $hour -ge 0 && $hour -lt 12 ]]; then
  echo "Good Morning!"
elif [[ $hour -ge 12 && $hour -lt 18 ]]; then
  echo "Good Afternoon!"
else
  echo "Good Evening!"
fi
```



```
[admin@hostname01 ~]$ ./Assi.sh
Good Evening!
```

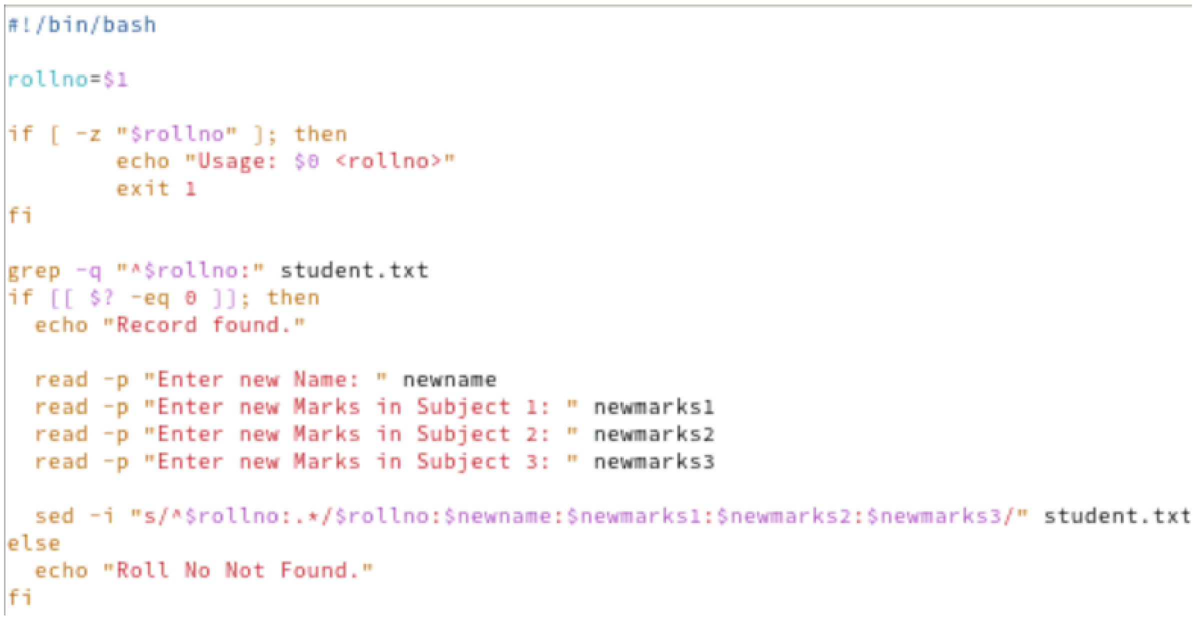
7 A data file file has some student records including rollno, names and subject marks. The fields are separated by a “:”. Write a shell script that accepts roll number from the user, searches it in the file and if the roll number is present - allows the user to modify name and marks in 3 subjects.
If 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

    while true; do
      read -p "Enter Roll No: " rollno

      grep -q "^$rollno:" student.txt
      if [[ $? -eq 0 ]]; then
        echo "Record found."

        read -p "Enter new Name: " newname
        read -p "Enter new Marks in Subject 1: " newmarks1
        read -p "Enter new Marks in Subject 2: " newmarks2
        read -p "Enter new Marks in Subject 3: " newmarks3

        sed -i "s/^$rollno:.*/$rollno:$newname:$newmarks1:$newmarks2:$newmarks3/"
    student.txt
      else
        echo "Roll No Not Found."
      fi

      read -p "Modify another record? (y/n): " choice
      if [[ $choice != "y" ]]; then
        break
      fi
    done
```

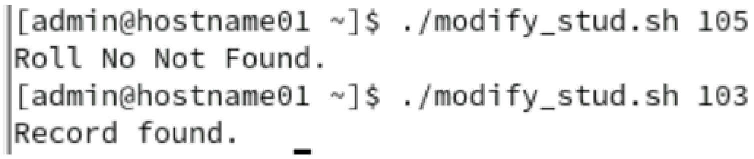


```
[admin@hostname01 ~]$ vim modify_stud.sh
[admin@hostname01 ~]$ chmod +x modify_stud.sh
[admin@hostname01 ~]$ ./modify_stud.sh
Enter Roll No: 103
Record found.
Enter new Name: Bob
Enter new Marks in Subject 1: 75
Enter new Marks in Subject 2: 80
Enter new Marks in Subject 3: 87
Modify another record? (y/n): n
```

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

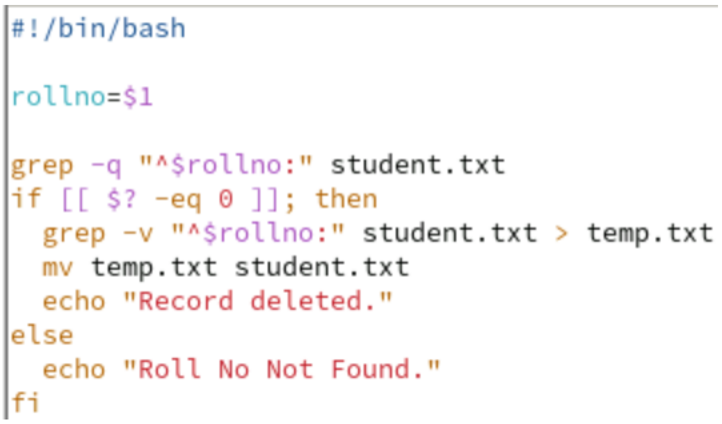


```
#!/bin/bash

rollno=$1

if [ -z "$rollno" ]; then
        echo "Usage: $0 <rollno>"
        exit 1
fi

grep -q "^$rollno:" student.txt
if [[ $? -eq 0 ]]; then
  echo "Record found."

  read -p "Enter new Name: " newname
  read -p "Enter new Marks in Subject 1: " newmarks1
  read -p "Enter new Marks in Subject 2: " newmarks2
  read -p "Enter new Marks in Subject 3: " newmarks3

  sed -i "s/^$rollno:.*/$rollno:$newname:$newmarks1:$newmarks2:$newmarks3/" student.txt
else
  echo "Roll No Not Found."
fi
```

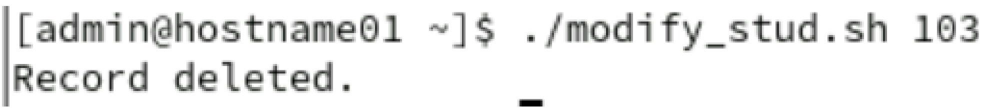


```
[admin@hostname01 ~]$ ./modify_stud.sh 105
Roll No Not Found.
[admin@hostname01 ~]$ ./modify_stud.sh 103
Record found.
```

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.

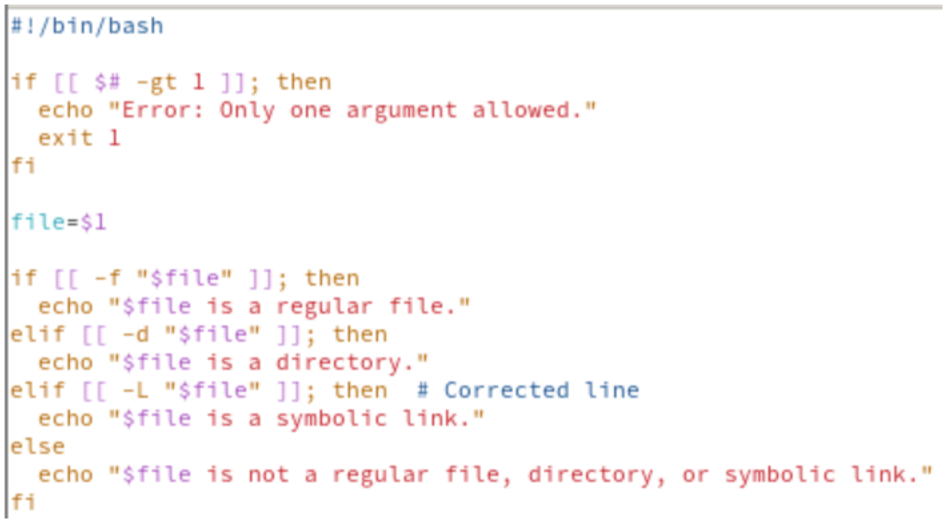


```
#!/bin/bash

rollno=$1

grep -q "^$rollno:" student.txt
if [[ $? -eq 0 ]]; then
  grep -v "^$rollno:" student.txt > temp.txt
  mv temp.txt student.txt
  echo "Record deleted."
else
  echo "Roll No Not Found."
fi
```

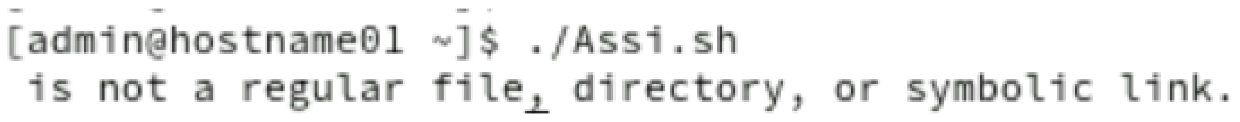


```
[admin@hostname01 ~]$ ./modify_stud.sh 103
Record deleted.
```

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.



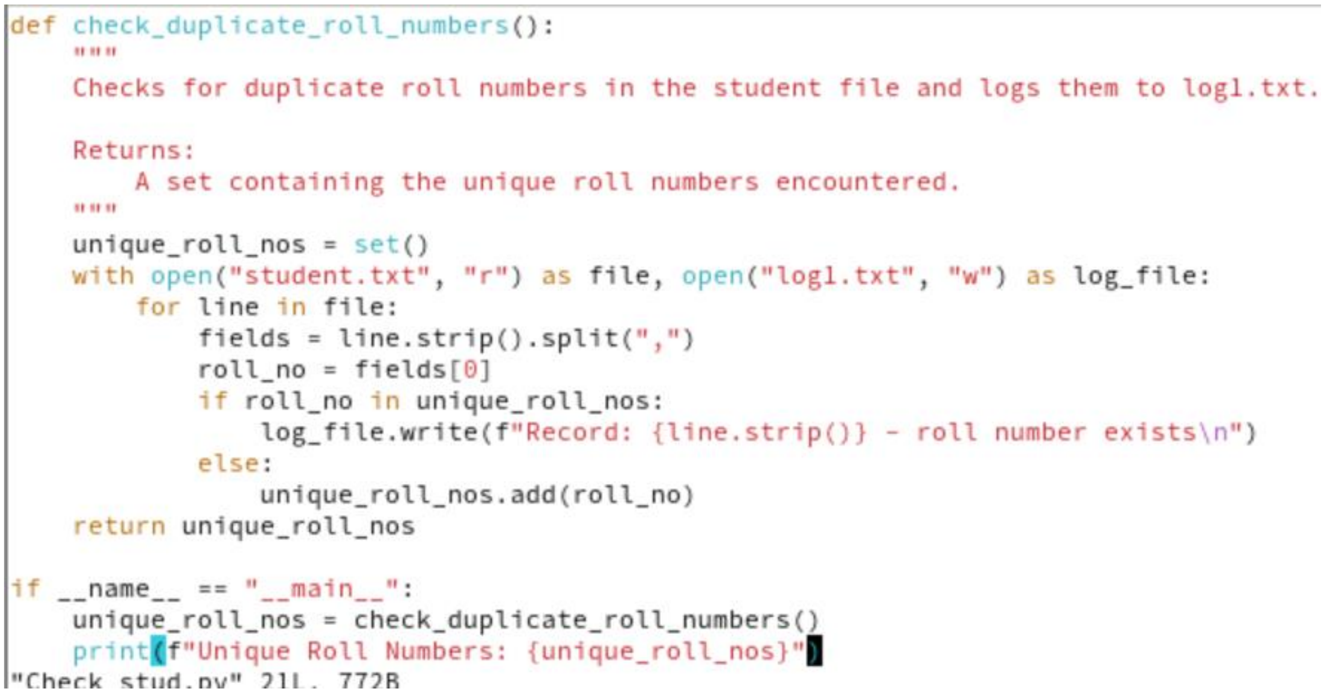
```
#!/bin/bash

if [[ $# -gt 1 ]]; then
  echo "Error: Only one argument allowed."
  exit 1
fi

file=$1

if [[ -f "$file" ]]; then
  echo "$file is a regular file."
elif [[ -d "$file" ]]; then
  echo "$file is a directory."
elif [[ -L "$file" ]]; then  # Corrected line
  echo "$file is a symbolic link."
else
  echo "$file is not a regular file, directory, or symbolic link."
fi
```



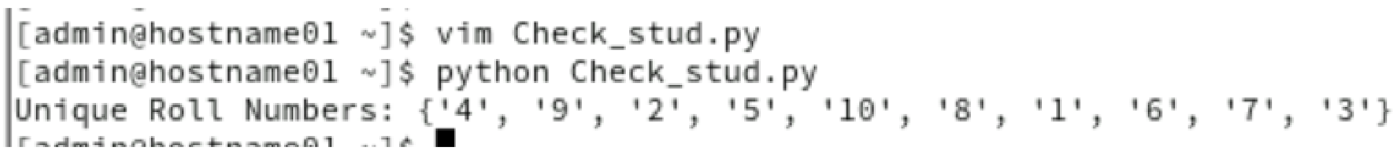
```
[admin@hostname01 ~]$ ./Assi.sh
 is not a regular file, directory, or symbolic link.
```

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”.



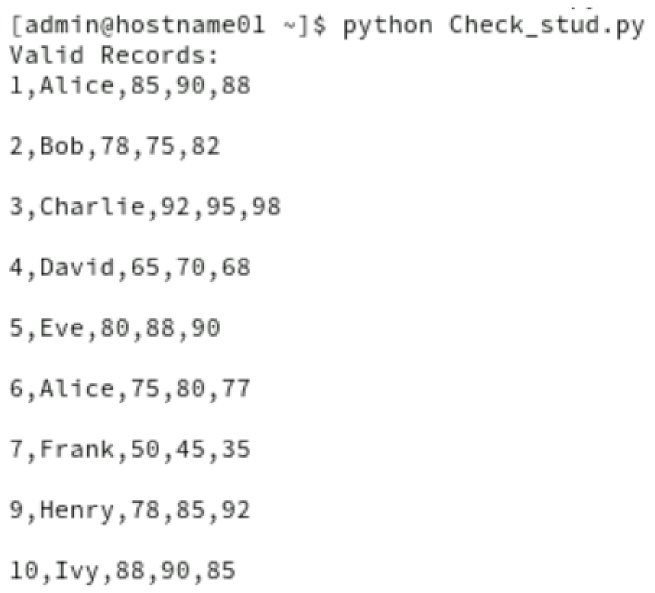
```
def check_duplicate_roll_numbers():
    """
    Checks for duplicate roll numbers in the student file and logs them to log1.txt.

    Returns:
        A set containing the unique roll numbers encountered.
    """
    unique_roll_nos = set()
    with open("student.txt", "r") as file, open("log1.txt", "w") as log_file:
        for line in file:
            fields = line.strip().split(",")
            roll_no = fields[0]
            if roll_no in unique_roll_nos:
                log_file.write(f"Record: {line.strip()} - roll number exists\n")
            else:
                unique_roll_nos.add(roll_no)
    return unique_roll_nos

if __name__ == "__main__":
    unique_roll_nos = check_duplicate_roll_numbers()
    print(f"Unique Roll Numbers: {unique_roll_nos}")
"Check_stud.py" 21L, 772B
```



```
[admin@hostname01 ~]$ vim Check_stud.py
[admin@hostname01 ~]$ python Check_stud.py
Unique Roll Numbers: {'4', '9', '2', '5', '10', '8', '1', '6', '7', '3'}
```

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"

```
def check_marks_range():
    valid_records = []
    with open("student.txt", "r") as file, open("log1.txt", "a") as log_file:
        for line in file:
            fields = line.strip().split(",")
            roll_no = fields[0]
            name = fields[1]
            try:
                marks_hindi = int(fields[2])
                marks_maths = int(fields[3])
                marks_physics = int(fields[4])
            except ValueError:
                log_file.write(f"Record: {line.strip()} - Invalid marks format\n")
                continue  # Skip to the next line if marks are not integers

            if 1 <= marks_hindi <= 99 and 1 <= marks_maths <= 99 and 1 <=
marks_physics <= 99:
                valid_records.append(line)
            else:
                log_file.write(f"Record: {line.strip()} - Marks out of range\n")

    return valid_records

if __name__ == "__main__":
    valid_records = check_marks_range()
    print("Valid Records:")
    for record in valid_records:
        print(record)
```



```
[admin@hostname01 ~]$ python Check_stud.py
Valid Records:
1,Alice,85,90,88

2,Bob,78,75,82

3,Charlie,92,95,98

4,David,65,70,68

5,Eve,80,88,90

6,Alice,75,80,77

7,Frank,50,45,35

9,Henry,78,85,92

10,Ivy,88,90,85
```

c. If the data is valid, the calculate total, percentage, grade and display on the terminal

```
def process_student_data():
    with open("student.txt", "r") as file:
        lines = file.readlines()
    with open("log1.txt", "w") as log_file:
        for line in lines:
            fields = line.strip().split(",")
            roll_no = fields[0]
            name = fields[1]
            marks_hindi = int(fields[2])
            marks_maths = int(fields[3])
            marks_physics = int(fields[4])


            if roll_no in roll_nos:
                log_file.write(f"Record: {line.strip()} - roll number exists\n")
            elif not (1 <= marks_hindi <= 99 and 1 <= marks_maths <= 99 and 1 <=
marks_physics <= 99):
                log_file.write(f"Record: {line.strip()} - marks out of range\n")
            else:
                total = marks_hindi + marks_maths + marks_physics
                percentage = (total / 300) * 100
            if percentage >= 90:
                grade = "A"
            elif percentage >= 80:
                grade = "B"
            elif percentage >= 70:
                grade = "C"
            elif percentage >= 60:
                grade = "D"
            else:
                grade = "F"

            print(f"Roll No: {roll_no}, Name: {name}")
            print(f"Total Marks: {total}")
            print(f"Percentage: {percentage:.2f}%")
            print(f"Grade: {grade}\n")

        roll_nos.add(roll_no)

if __name__ == "__main__":
    roll_nos = set()
    process_student_data()
```

-
  ```
  admin@hostname01 ~]$ python Check_stud.py
      Roll No: 1, Name: Alice
      Total Marks: 263
      Percentage: 87.67%
      Grade: B

      Roll No: 2, Name: Bob
      Total Marks: 235
      Percentage: 78.33%
      Grade: C

      Roll No: 3, Name: Charlie
      Total Marks: 285
      Percentage: 95.00%
  ```

```
Grade: A

Roll No: 4, Name: David
Total Marks: 203
Percentage: 67.67%
Grade: D

Roll No: 5, Name: Eve
Total Marks: 258
Percentage: 86.00%
Grade: B

Roll No: 6, Name: Alice
Total Marks: 232
Percentage: 77.33%
Grade: C

Roll No: 7, Name: Frank
Total Marks: 130
Percentage: 43.33%
Grade: F

Roll No: 9, Name: Henry
Total Marks: 255
Percentage: 85.00%
Grade: B

Roll No: 10, Name: Ivy
Total Marks: 263
Percentage: 87.67%
Grade: B
```