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

\*

\*\*

\*\*\*

\*\*\*\*

#!/bin/bash

for i in {1..4}

do

for j in $(seq 1 $i)

do

echo -n "\*"

done

echo

done

1. 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

# Accept first name, middle name, and last name

echo "Enter your first name:"

read fname

echo "Enter your middle name:"

read mname

echo "Enter your last name:"

read lname

# Greet the person using their full name

echo "Hello, $fname $mname $lname! Welcome!"

1. 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

# Get the list of files in the current directory along with their sizes in bytes

files=$(ls -l | grep -v ^d | awk '{print $9 " " $5}')

# Display all the files with their sizes

echo "Files and their sizes in bytes:"

echo "$files"

# Find the file with the maximum size

max\_file=$(echo "$files" | sort -k2 -n | tail -n 1)

# Find the file with the minimum size

min\_file=$(echo "$files" | sort -k2 -n | head -n 1)

# Display the file with maximum size

echo

echo "File with the maximum size: $max\_file"

# Display the file with minimum size

echo "File with the minimum size: $min\_file"

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

(Note: Working day Mon-Fri)

#!/bin/bash

# Get the current day of the week

day=$(date +%A)

# Check if the day is a working day (Monday to Friday)

if [[ "$day" == "Monday" || "$day" == "Tuesday" || "$day" == "Wednesday" || "$day" == "Thursday" || "$day" == "Friday" ]]; then

echo "Today is a working day."

else

echo "Today is not a working day."

fi

1. 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

# Accept the weight of the person

echo "Enter your weight (in kgs):"

read weight

# Check if the weight is within the acceptable range (30 to 250 kgs)

if [ "$weight" -ge 30 ] && [ "$weight" -le 250 ]; then

echo "Welcome to HP Health Club! Your membership is confirmed."

else

echo "Sorry, your weight is not within the acceptable range for membership."

fi

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

#!/bin/bash

# Get the current hour in 24-hour format

hour=$(date +%H)

# Greet the user based on the time of day

if [ "$hour" -ge 5 ] && [ "$hour" -lt 12 ]; then

echo "Good Morning!"

elif [ "$hour" -ge 12 ] && [ "$hour" -lt 17 ]; then

echo "Good Afternoon!"

elif [ "$hour" -ge 17 ] && [ "$hour" -lt 21 ]; then

echo "Good Evening!"

else

echo "Good Night!"

fi

1. 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

# Define the file containing the student records

file="file.txt"

# Function to modify a student's record

modify\_record() {

# Prompt the user for the roll number to modify

echo "Enter the roll number to modify:"

read rollno

# Search for the roll number in the file

record=$(grep "^$rollno:" "$file")

# Check if the record exists

if [ -z "$record" ]; then

echo "Roll No Not Found"

else

# Extract current details (rollno, name, and marks)

name=$(echo "$record" | cut -d':' -f2)

marks1=$(echo "$record" | cut -d':' -f3)

marks2=$(echo "$record" | cut -d':' -f4)

marks3=$(echo "$record" | cut -d':' -f5)

echo "Current details for Roll No $rollno:"

echo "Name: $name"

echo "Marks in Subject 1: $marks1"

echo "Marks in Subject 2: $marks2"

echo "Marks in Subject 3: $marks3"

# Prompt the user to modify the name

echo "Enter the new name (press Enter to keep the current name):"

read new\_name

if [ -n "$new\_name" ]; then

name="$new\_name"

fi

# Prompt the user to modify marks for 3 subjects

echo "Enter new marks for Subject 1 (current: $marks1):"

read new\_marks1

if [ -n "$new\_marks1" ]; then

marks1="$new\_marks1"

fi

echo "Enter new marks for Subject 2 (current: $marks2):"

read new\_marks2

if [ -n "$new\_marks2" ]; then

marks2="$new\_marks2"

fi

echo "Enter new marks for Subject 3 (current: $marks3):"

read new\_marks3

if [ -n "$new\_marks3" ]; then

marks3="$new\_marks3"

fi

# Modify the record in the file

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

echo "Record updated successfully!"

fi

}

# Run the function to allow the user to modify one record at a time

modify\_record

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

#!/bin/bash

# Check if roll number is passed as a command line argument

if [ $# -ne 1 ]; then

echo "Usage: $0 <RollNo>"

exit 1

fi

# Get the roll number from the command line argument

rollno=$1

# Define the file containing the student records

file="file.txt"

# Function to modify a student's record

modify\_record() {

# Search for the roll number in the file

record=$(grep "^$rollno:" "$file")

# Check if the record exists

if [ -z "$record" ]; then

echo "Roll No Not Found"

else

# Extract current details (rollno, name, and marks)

name=$(echo "$record" | cut -d':' -f2)

marks1=$(echo "$record" | cut -d':' -f3)

marks2=$(echo "$record" | cut -d':' -f4)

marks3=$(echo "$record" | cut -d':' -f5)

echo "Current details for Roll No $rollno:"

echo "Name: $name"

echo "Marks in Subject 1: $marks1"

echo "Marks in Subject 2: $marks2"

echo "Marks in Subject 3: $marks3"

# Prompt the user to modify the name

echo "Enter the new name (press Enter to keep the current name):"

read new\_name

if [ -n "$new\_name" ]; then

name="$new\_name"

fi

# Prompt the user to modify marks for 3 subjects

echo "Enter new marks for Subject 1 (current: $marks1):"

read new\_marks1

if [ -n "$new\_marks1" ]; then

marks1="$new\_marks1"

fi

echo "Enter new marks for Subject 2 (current: $marks2):"

read new\_marks2

if [ -n "$new\_marks2" ]; then

marks2="$new\_marks2"

fi

echo "Enter new marks for Subject 3 (current: $marks3):"

read new\_marks3

if [ -n "$new\_marks3" ]; then

marks3="$new\_marks3"

fi

# Modify the record in the file

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

echo "Record updated successfully!"

fi

}

# Run the function to allow the user to modify the record

modify\_record

1. 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

# Check if roll number is passed as a command line argument

if [ $# -ne 1 ]; then

echo "Usage: $0 <RollNo>"

exit 1

fi

# Get the roll number from the command line argument

rollno=$1

# Define the file containing the student records

file="file.txt"

# Function to delete a student's record

delete\_record() {

# Search for the roll number in the file

record=$(grep "^$rollno:" "$file")

# Check if the record exists

if [ -z "$record" ]; then

echo "Roll No Not Found"

else

# Display the current record

echo "Current details for Roll No $rollno:"

echo "$record"

# Ask for confirmation to delete

echo "Are you sure you want to delete this record? (yes/no)"

read confirmation

if [ "$confirmation" == "yes" ]; then

# Delete the record using sed and update the file

sed -i "/^$rollno:/d" "$file"

echo "Record for Roll No $rollno has been deleted successfully."

else

echo "Deletion aborted."

fi

fi

}

# Run the function to allow the user to delete the record

delete\_record

1. 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

# Check if exactly one argument is passed

if [ $# -ne 1 ]; then

echo "Error: Please provide exactly one argument."

exit 1

fi

# Get the file or directory from the argument

file=$1

# Check if the file or directory exists

if [ ! -e "$file" ]; then

echo "Error: $file does not exist."

exit 1

fi

# Report the type of the file

if [ -f "$file" ]; then

echo "$file is a regular file."

elif [ -d "$file" ]; then

echo "$file is a directory."

elif [ -l "$file" ]; then

echo "$file is a symbolic link."

elif [ -b "$file" ]; then

echo "$file is a block device file."

elif [ -c "$file" ]; then

echo "$file is a character device file."

elif [ -p "$file" ]; then

echo "$file is a named pipe (FIFO)."

elif [ -S "$file" ]; then

echo "$file is a socket."

else

echo "$file is of an unknown type."

fi

1. 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
   1. If the roll number already exists, then store the record and the following message   
      “roll number exists” in a log file “log1”.
   2. 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”
   3. If the data is valid, the calculate total, percentage, grade and display on the terminal

#!/bin/bash

# Define the student data file and log file

student\_file="student"

log\_file="log1"

# Function to check and add a student record

add\_student\_record() {

# Read the student details (RollNo, Name, Marks)

echo "Enter RollNo:"

read rollno

echo "Enter Name:"

read name

echo "Enter Marks in Hindi (1-99):"

read marks\_hindi

echo "Enter Marks in Maths (1-99):"

read marks\_maths

echo "Enter Marks in Physics (1-99):"

read marks\_physics

# Check if the roll number already exists

if grep -q "^$rollno:" "$student\_file"; then

echo "Roll number $rollno exists" >> "$log\_file"

echo "Roll number exists. Record saved to log1."

return

fi

# Check if marks are within the valid range (1-99)

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 "$rollno:$name:$marks\_hindi:$marks\_maths:$marks\_physics" >> "$log\_file"

echo "Marks out of range. Record saved to log1."

return

fi

# If data is valid, calculate total, percentage, and grade

total=$((marks\_hindi + marks\_maths + marks\_physics))

percentage=$((total / 3))

# Calculate grade based on percentage

if [ "$percentage" -ge 90 ]; then

grade="A"

elif [ "$percentage" -ge 70 ]; then

grade="B"

elif [ "$percentage" -ge 50 ]; then

grade="C"

else

grade="D"

fi

# Store valid student record in the student file

echo "$rollno:$name:$marks\_hindi:$marks\_maths:$marks\_physics" >> "$student\_file"

# Display total, percentage, and grade

echo "Record added successfully!"

echo "Total: $total"

echo "Percentage: $percentage%"

echo "Grade: $grade"

}

# Call the function to add a student record

add\_student\_record