

## Linux Programming: Assignment-9

1. Write a shell script using if...else to check if a number is even or odd.

Answer:

```
#!/bin/bash

read -p "Enter a number: " num

if [  $$(num \% 2)$  -eq 0]
then
    echo "$num is Even"
else
    echo "$num is Odd"
fi
```

2. Explain the difference between if and case statements in bash.

Answer: if is mainly used to test conditions that result in true or false, handling simple yes/no checks. case is better when dealing with multiple possible values for a variable, like checking many options or menu choices, making the code neater than many if-elif tests.

3. Write a script to find the largest of three numbers entered by the user.

Answer:

```
#!/bin/bash

read -p "Enter first number: " a
read -p "Enter second number: " b
read -p "Enter third number: " c

if [  $a -ge b$  ] && [  $a -ge c$  ]; then
    echo "$a is the largest"
elif [  $b -ge a$  ] && [  $b -ge c$  ]; then
    echo "$b is the largest"
else
    echo "$c is the largest"
fi
```

4. How do you use a for loop to traverse an array in bash? Give an example. The array is defined as arr= (123, "Abs", -2.3, 'A', 23.56, 0).

Answer:

```
#!/bin/bash
arr= (123 "Abs" -2.3 'A' 23.56 0)
for element in "${arr[@]}"
do
    echo "$element"
done
```

5. Write a shell script to loop through all files in the current directory and display their names.

Answer:

```
#!/bin/bash
for file in *
do
    echo "File: $file"
done
```

6. What is the difference between while and until loops in bash?

Answer:

A while loop continues running as long as its condition remains true. An until loop keeps going as long as its condition remains false and stops once the condition becomes true

7. Write a countdown timer script using a while loop.

Answer:

```
#!/bin/bash
read -p "Enter countdown time (in seconds): " time
while [ $time -gt 0]
do
    echo "Time left: $time seconds"
    sleep 1
    ((time--))
done
echo "Time's up!"
```

8. How do you use break and continue statements in loops? Give examples.

Answer:

- break stops the entire loop immediately
- continue skips the current loop iteration and moves to the next one.

Example:

```
for i in {1..5}
do
    if [ $i -eq 3 ]; then
        continue # skips 3
    fi
    if [ $i -eq 5 ]; then
        break # exits loop at 5
    fi
    echo "Number: $i"
done
```

9. Write a script to check if a file exists or not using the if and else loop.

Answer:

```
#!/bin/bash
read -p "Enter filename: " fname
if [ -f "$fname" ]; then
    echo "File '$fname' exists."
else
    echo "File '$fname' does not exist."
fi
```

10. Write a script to calculate factorial of a number using for loop.

Answer:

```
#!/bin/bash
read -p "Enter a number: " n
fact=1
for ((i=1; i<=n; i++))
```

do

fact=\$((fact \* i))

done

echo "Factorial of \$n is \$fact"

NAME:Sai charan S

USN:ENG24CY0153

ROLL NUMBER:25

SEC:3B