Name= shaik Raakin
Class =3rd sem
Branch = cybersecurity
usn=ENG24CY0192
Roll no:64
section:A
Assignment : 9

21Q)Write a shell script using if…else to check if a number is even or odd.
sol)
The shell script to check if a number is odd or even is :

```
#!/bin/bash
read -p "Enter a number: " num
if [ $((num%2)) -eq 0 ]
then
  echo "Even"
else
  echo "Odd"
fi
```

22Q)Explain the difference between if and case statements in bash.
sol)
The primary distinction between if statements and case statements in Bash lies in the manner in which conditions are evaluated. if statements allow you to evaluate complex expressions or results from commands, and let you include logical operations to manage multiple cases. This makes it particularly useful when you have a set of possible conditions (numbers, strings, command outputs).

Conversely, the case statement is specifically designed to evaluate a single variable or value against multiple possible fixed patterns, like menu options or file extensions. This allows the use of an if statement, it is simply shorter and easier to read in the case you have a number of specific options available. While if statements provide flexibility for greater logical evaluation, case statements succeed in matching one value and returning (or doing something else) for all potential known values/conditions.

23Q)Write a script to find the largest of three numbers entered by the user.
sol)
The shell script for this is :

```
#!/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 largest"
elif [ $b -ge $a ] && [ $b -ge $c ]
then
  echo "$b is largest"
else
  echo "$c is largest"
fi
```

24Q). 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).
sol)
The shell script to transverse a array in bash is:

```
#!/bin/bash
arr=(123, “Abs”, -2.3, ‘A’, 23.56, 0)
for i in "${arr[@]}";
do
	echo $i
done
```

25Q)Write a shell script to loop through all files in the current directory and display their names.
sol)
The shell script is:

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

26Q)What is the difference between while and until loops in bash?
sol)
The distinction between a while loop and an until loop occurs in Bash in how they evaluate their condition. A while loop evaluates true and continues as long as that true is testable, and they are normally useful to be used when you want to almost take action while something is true. When you use an until loop, that code will repeat, but only until your test condition is true or is met.

In summary, while repeats code while a condition is true, and until repeats code until a condition is true. The difference can be a choice as to whether you want your process to continue while something is true, or if you want the action to continue until something happens. This offers an elegant and clean way to write loops depending on what you want to test for to stop looping.

27Q)Write a countdown timer script using a while loop.
sol)
The shell script for the countdown timer is:

```
#!/bin/bash
count=10
while [ $count -gt 0 ]
do
  echo $count
  sleep 1
  count=$((count-1))
done
echo "Time's up!"
```

28Q)How do you use break and continue statements in loops? Give examples.
sol)
When it comes to controlling the flow of execution, the break and continue statements are used in Bash loops.

The break statement immediately breaks out of a loop, subsequent iterations will not be continued. This is useful for exiting (or breaking) out of a loop once your desired condition is met.

The continue statement skips to the next iteration in the remaining loop statements. This is helpful when you want to skip certain conditions but continue looping.

Example:
```
for i in {1..5}
do
  if [ $i -eq 3 ]; then
    continue  # Skip when i is 3
  fi
  if [ $i -eq 5 ]; then
    break     # Exit loop when i is 5
  fi
  echo $i
done
```

29Q)Write a script to check if a file exists or not using the if and else loop.
sol)
The shell script is:

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

30Q)Write a script to calculate factorial of a number using for loop.
sol)
The shell script to calculate factorial is:

```
#!/bin/bash
read -p "Enter a number: " num
fact=1
for (( i=1; i<=num; i++ ))
do
  fact=$((fact * i))
done
echo "Factorial of $num is $fact"
```