Linux Programming: Assignment-9

NAME: SANDEEP
USN:ENG24CY0047
ROLL NO:40
SECTION:B

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

A:
```
echo "Enter a number:"

read num

if [ $((num % 2)) -eq 0 ]

then

   echo "The number $num is even."

else

   echo "The number $num is odd."

fi
```

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

A: 1. IF statement

Used if you wish to test conditions (such as numbers, strings, comparisons).

It will work well for complicated logical tests (such as >, <, ==, !=, etc.).

Example:

```
if [ $num -gt 10 ]

then

   echo "Number is greater than 10"

else

   echo "Number is 10 or smaller"

fi
```

2. CASE statement

Used when a single variable has numerous possible fixed values.It's similar to a menu — it selects a value from among many.

Example:

```
case $day in
```

```
   "Mon") echo "Start of the week" ;;
   "Fri") echo "Weekend is close!" ;;
   "Sun") echo "It's holiday!" ;;
   *) echo "Just another day" ;;
Esac
```

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

A:
```
echo "Enter first number:"
read a
echo "Enter second number:"
read b
echo "Enter third number:"
read c
f [ $a -ge $b ] && [ $a -ge $c ]
then
   echo "The biggest number is: $a"
elif [ $b -ge $a ] && [ $b -ge $c ]
then
   echo "The biggest number is: $b"
else
echo "The largest  number is: $c"
fi
```

24. 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).

A:
```
arr=(123 "Abs" -2.3 'A' 23.56 0)
echo "The elements of the array are:"
for item in "${arr[@]}"
do
 echo "$item"
done
```

ash

Copy code

```
arr=(123 "Abs" -2.3 'A' 23.56 0)
```

"${arr[@]}" provides all elements of the array.

The for loop loops through all the elements individually.

Every value is outputted with echo.

Output:

Sql   Copy code

```
The values in the array are:
123   Abs  -2.3  A  23.56
```

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

A: echo "Files in the present directory are:"

```
for file in *
do
   echo "$file"
done
```

The for loop iterates over each item one at a time.

echo "$file" prints out the name of each folder or file.

Example Output:

sql

Copy code

```
Files in the present directory are:
file1.txt
file2.sh
image.png
notes.docx
```

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

A: 1. while loop

The while loop continues as long as the condition is true.

As soon as the condition turns to false, it finishes.

Example:

```
count=1
while [ $count -le 5 ]
do
  echo "Count is $count"
  ((count++))
done
```

This loop continues while $count is less than or equal to 5.

2. until loop

The until loop continues until the condition turns true.

That is, it keeps on looping when the condition is false.

Example:

```
count=1
until [ $count -gt 5 ]
do
  echo "Count is $count"
  ((count++))
done
```

This loop will continue running until $count is greater than 5

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

A:
```
echo "Enter the number of seconds to countdown:"
read time
while [ $time -gt 0 ]
do
    echo "Time remaining: $time seconds"
    sleep 1
        ((time--))
```

```
echo "Time's up!"
```

The user input how many seconds to countdown.The while loop executes while time is more than 0.sleep 1 waits for 1 second each time.

((time--)) decrements the time by 1.

When the time becomes 0, it prints "Time's up!"

Example Output:

Sql   Copy code

```
Input the number of seconds to count down:

5

Time left: 5 seconds

Time left: 4 seconds

Time left: 3 seconds

Time left: 2 seconds

Time left: 1 second

Time's up!
```

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

A: 1. break statement

The break statement is utilized to terminate a loop absolutely.If the loop reaches break, it breaks out and terminates running — even if it has additional steps to go.

Example:for num in 1 2 3 4 5

```
do

   if [ $num -eq 3 ]

   then

break

fi

echo "Number: $num"

done
```

Output:Number: 1

```
Number: 2
```

2. continue statement:The continue keyword bypasses the remaining code for that single round of the loop.The loop continues on to the next item rather than terminating.Example:for num in 1 2 3 4 5

```
do
   if [ $num -eq 3 ]
   then
      continue
   fi
   echo "Number: $num"
done
```

Output:Number: 1

```
Number: 2
Number: 4
Number: 5
```

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

A: echo "Enter the file name:"

```
read filename
if [ -e "$filename" ]
then
   echo "Yes, the file '$filename' exists."
else
   echo "Sorry, the file '$filename' does not exist."
fi
```

The script requests you to input a file name.The -e option is used to check whether the file is present in the present directory.If it is present → it displays a message that the file exists.If not → it outputs that the file doesn't exist.

Case 1: File exists

pgsql

Copy code

```
Enter the file name:
```

```
notes.txt
Yes, the file 'notes.txt' exists.
```

Case 2: File does not exist

nginx

Copy code

```
Enter the file name:
hello.txt
Sorry, the file 'hello.txt' does not exist.
```

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

A: cho "Enter a number:"

```
read num
fact=1
for (( i=1; i<=num; i++ ))
do
   fact=$((fact * i))  # Multiply fact by i every time
done
echo "The factorial of $num is: $fact"
```

The user inputs a number (for example, 5).

fact begins at 1.The for loop increments fact by each number from 1 to the input number.When the loop finishes, fact has the factorial.

Example Output:

yaml

Copy code

```
Enter a number:
5
The factorial of 5 is: 120
```