Remedial assignment Day-2

Conditional Statements | Shell Script:

→ There are five conditional statements which can be used in bash programming.

1) If Statement
2) If-else Statement
3) If... elif... else.. fi Statements
4) If... then... else.. if... then.. fi (Nested if)
5) Switch Statement

1) **If Statement:-** In this Statement the block will process if specified condition is true.

Syntax:-

```
if [ Expression ]
then
   Statement
fi
```

Example:

```
a=20
b=30
if [$a==$b]
then
  echo "a is equal to b"
fi

if [$a != $b]
then
  echo "a is not equal to b"
 fi
```

Output:

a is not equal to b.

2) If-else Statement:- In this statement if specified condition is not true in "if" part then else part will execute

Syntax

```
if [Expression]
then
  Statement1
else
  Statement2
fi
```

Example:

```
a=20
b=20
if [$a ==$b]
then
   echo "a is equal to b"
else
   echo "a is not equal to b"
fi
```

OUTPUT:

a is equal to b

3) If.. Elif.. Else.. fi Statement (Else if ladder): To use multiple conditions in one if-else block, then elif keyword is used in shell. if Expression 1 is true then it executes Statement 1 and 2, and this process continues. If none of the condition is true then it processes Else Part.

Syntax:

```
if [Expression1]
then
   Statement 1
   Statement 2
    :
elif [Expression2]
then
   Statement 3
   Statement 4
    :
else
   Statement 5
fi
```

4) If then Else if then fi fi (Nested if): Nested if-else block can be used when, one condition is satisfies then it again checks another condition. In Syntax, if Expression1 is false then it processes Else part, and again Expression2 will be check.

Syntax:

```
if [Expression1]
then
```

```
    Statement 1
    Statement 2
    .
  Else
    if [Expression2]
     then
        Statement 3
        :
     fi
  fi
```

5) Switch Statement : In this if specified value match with the Pattern then it will execute a block of that particular pattern this case will be terminated when the last command is executed.

Syntax:

```
Case in
    Pattern 1) Statement 1;;
    Pattern n) Statement n;;
esac
```

Example:

```
CARS = "bmw"

Case "$CARS" in

"mercedes") echo "Headquarters - ABC, Germany";

"audi") echo "Headquarters - Ingo, Germany";

"bmw") echo "Headquarters - Chennai, Tamil Nadu, India";
esac
```

OUTPUT:

```
Headquarters - Chennai, Tamil Nadu, India
```

Loops in Shell scripts:

Bash Supports:

- for loop
- while loop
- until loop

To alter the flow of loop statements, two commands are used they are:

1) Break
2) continue

- for Statement: The for loop operate on list of items, it repeats a set of commands for every item in a list.

Syntax: for var in word1, word2.... wordn
do
Statement to be Executed
done

- while Statement: Here, the command is evaluated and based on the result loop will Executed, if command raise to false then loop will be terminated

Syntax: while command
do
Statement to be Executed
done

until Statement: The until loop is executed as many as times the condition/commands evaluates to false. The loop terminates when the condition/commands become true.

Syntax:

```
until command
do
   Statement to be executed until true
done
```

Example:

```
a=0
until [$a -gt 10]
do
   echo $a
   a=`expr $a+1`
done
```

Example (while loop):

```
a=0
while [$a -lt 10]
do
   echo $a
   a=`expr $a+1`
done
```

Example (for loop):

```
COLORS="green red blue"
for COLOR in $COLORS
do
   echo "COLOR: $COLOR"
done
```