BASH SCRIPTING ASSIGNMENT

1.An example demonstrating the use of double parentheses for arithmetic operations in a Bash shell script.

Step 1: Create an "**arithmetic.sh"** script file using touch command

Step 2: Create a nano file to write the code

```
root@serverid:~# touch arithmetic.sh
root@serverid:~# nano arithmetic.sh
root@serverid:~# chmod +x arithmetic.sh
```

Step 3: Write the Code in nano arithmetic.sh script file

```
root@serverid: ~
  GNU nano 7.2                                                    arithmetic.sh *
#!/bin/bash
x=12
y=3
echo "x=12, y=3"
echo "Addition of x & y"
echo $(( $x + $y ))
echo "Subtraction of x & y"
echo $(( $x - $y ))
echo "Multiplication of x & y"
echo $(( $x * $y ))
echo "Division of x by y"
echo $(( $x / $y ))
echo "Exponentistion of x,y"
echo $(( $x ** $y ))
echo "Modular Division of x,y"
echo $(( $x % $y ))
echo "Incrementing x by 5, then x="
(( x += 5 ))
echo $x
echo "Decrementing x by 5, then x="
(( x -= 5 ))
echo $x
echo "Multiply of x by 5, then x="
(( x *= 5 ))
echo $x
echo "Dividing x by 5, then x="
(( x /= 5 ))
echo $x
echo "Remainder of Dividing x by 5, then x="
(( x %= 5 ))
echo $x
```

Step 4: Output

```
root@serverid:~# ./arithmetic.sh
x=12, y=3
Addition of x & y
15
Subtraction of x & y
9
Multiplication of x & y
36
Division of x by y
4
Exponentistion of x,y
1728
Modular Division of x,y
0
Incrementing x by 5, then x=
17
Decrementing x by 5, then x=
12
Multiply of x by 5, then x=
60
Dividing x by 5, then x=
12
Remainder of Dividing x by 5, then x=
2
root@serverid:~#
```

2. Shell Program to demonstrate the use of let command in a Bash script.

Step 1: Create an "**arithmetic1.sh"** script file using touch command

Step 2: Create a nano file to write the code

```
root@serverid:~# touch arithmetic1.sh
root@serverid:~# nano arithmetic1.sh
root@serverid:~# chmod +x arithmetic1.sh
```

Step 3: Write the Code in nano arithmetic1.sh script file

```
root@serverid: ~
  GNU nano 7.2                                                    arithmetic1.sh
#! /bin/bash
x=10
y=6
z=0
echo "Addition"
let "z=$(( x + y ))"
echo "z=$z"
echo "Subtraction"
let "z=$(( x - y ))"
echo "z=$z"
echo "Multiplication"
let "z=$(( x * y ))"
echo "z=$z"
echo "Division"
let "z=$(( x / y ))"
echo "z=$z"
echo "Exponentiation"
let "z=$(( x ** y ))"
echo "z=$z"
echo "Modular Division"
let "z=$(( x % y ))"
echo "z=$z"
let "x += 5"
echo "Incrementing x by 5, then x="
echo $x
let "x -= 5"
echo "Decrementing x by 5, then x="
echo $x
let "x *= 5"
echo "Multiply of x by 5, then x="
echo $x
let "x /= 5"
echo "Dividing x by 5, x="
echo $x
let "x %= 5"
echo "Remainder of Dividing x by 5, x="
echo $x
```

Step 4: Output

```
root@serverid:~# ./arithmetic1.sh
Addition
z=16
Subtraction
z=4
Multiplication
z=60
Division
z=1
Exponentiation
z=1000000
Modular Division
z=4
Incrementing x by 5, then x=
15
Decrementing x by 5, then x=
10
Multiply of x by 5, then x=
50
Dividing x by 5, x=
10
Remainder of Dividing x by 5, x=
0
```

3. Shell Program to demonstrate the use of backticks and expr in a Bash script.

Step 1: Create an "**arithmetic2.sh**" script file using touch command

Step 2: Create a nano file to write the code

```
root@serverid:~# touch arithmetic2.sh
root@serverid:~# nano arithmetic2.sh
root@serverid:~# chmod +x arithmetic2.sh
```

Step 3: Write the Code in nano arithmetic2.sh script file

```
root@serverid: ~
  GNU nano 7.2                                                    arithmetic2.sh
#!/bin/bash
#Basic arithmetic using expr
echo "a=10, b=3"
echo "c is the value of addition c=a+b"
a=10
b=3
echo "c= `expr $a + $b`"
```

Step 4: Output

```
root@serverid:~# ./arithmetic2.sh
a=10, b=3
c is the value of addition c=a+b
c= 13
root@serverid:~#
```

4. Shell Program to take a user-input of any number and check if the value is greater than 125.

Step 1: Create an "**conditions.sh**" script file using touch command

Step 2: Create a nano file to write the code

```
root@serverid:~# touch conditions.sh
root@serverid:~# nano conditions.sh
root@serverid:~# chmod conditions.sh
```

Step 3: Write the Code in nano conditions.sh script file

```
root@serverid: ~
  GNU nano 7.2                                          conditions.sh
#!/bin/bash
read -p " Enter number : " number
if [ $number -gt 125 ]
then
echo "Value is greater than 125"
fi
```

Step 4: Output

```
root@serverid:~# ./conditions.sh
 Enter number : 149
Value is greater than 125
```

5. Shell Program to demonstrate the usage of if statement with a simple scenario of comparing two strings

Step 1: Create an "**conditions1.sh**" script file using touch command

Step 2: Create a nano file to write the code

```
root@serverid:~# touch conditions1.sh
root@serverid:~# nano conditions1.sh
root@serverid:~# chmod +x conditions1.sh
```

Step 3: Write the Code in nano conditions1.sh script file

```
root@serverid: ~
  GNU nano 7.2                                          conditions1.sh *
#!/bin/bash
# if condition is true
if [ "myfile" == "myfile" ];
then
echo "true condition"
fi
# if condition is false
if [ "myfile" == "yourfile" ];
then
echo "false condition"
fi
```

Step 4: Output

```
root@serverid:~# ./conditions1.sh
true condition
```

6. Shell Program to demonstrate how to compare numbers by using the if statement.

Step 1: Create an "**conditions2.sh**" script file using touch command

Step 2: Create a nano file to write the code

```
root@serverid:~# touch conditions2.sh
root@serverid:~# nano conditions2.sh
root@serverid:~# chmod +x conditions2.sh
```

Step 3: Write the Code in nano conditions2.sh script file

```
root@serverid: ~
  GNU nano 7.2                                                    conditions2.sh *
#!/bin/bash
#if condition (greater than) is true
if [ 10 -gt 3 ];
then
echo "10 is greater than 3."
fi
#if condition (greater than) is false
if [ 3 -gt 10 ];
then
echo "3 is not greater than 10."
fi
#if condition (lesser than) is true
if [ 3 -lt 10 ];
then
echo "3 is less than 10."
fi
#if condition (lesser than) is false
if [ 10 -lt 3 ];
then
echo "10 is not less than 3."
fi
#if condition (equal to) is true
if [ 10 -eq 10 ];
then
echo "10 is equal to 10."
fi
#if condition (equal to) is false
if [ 10 -eq 9 ];
then
echo "10 is not equal to 9"
fi
```

Step 4: Output

```
root@serverid:~# ./conditions2.sh
10 is greater than 3.
3 is less than 10.
10 is equal to 10.
root@serverid:~#
```


7. Shell Program to demonstrate the use of AND operator to include multiple conditions in if expression.

Step 1: Create an "**conditions3.sh"** script file using touch command

Step 2: Create a nano file to write the code

```
root@serverid:~# touch conditions3.sh
root@serverid:~# nano conditions3.sh
root@serverid:~# chmod +x conditions3.sh
```

Step 3: Write the Code in nano conditions3.sh script file

```
root@serverid: ~
  GNU nano 7.2                                                    conditions3.sh *
#!/bin/bash
# TRUE && TRUE
if [ 8 -gt 6 ] && [ 10 -eq 10 ];
then
echo "Conditions are true"
fi
# TRUE && FALSE
if [ "mylife" == "mylife" ] && [ 3 -gt 10 ];
then
echo "Conditions are false"
fi
```

Step 4: Output

```
root@serverid:~# ./conditions3.sh
Conditions are true
```

8. Shell Program to demonstrate the use of OR operator to include multiple conditions in if expression.

Step 1: Create an "**conditions4.sh"** script file using touch command

Step 2: Create a nano file to write the code

```
root@serverid:~# touch conditions4.sh
root@serverid:~# nano conditions4.sh
root@serverid:~# chmod +x conditions4.sh
```

Step 3: Write the Code in nano conditions4.sh script file

```
root@serverid: ~
  GNU nano 7.2                                                    conditions4.sh *
#!/bin/bash
# TRUE || FALSE
if [ 8 -gt 7 ] || [ 10 -eq 3 ];
then
echo " Condition is true. "
fi
# FALSE || FALSE
if [ "mylife" == "yourlife" ] || [ 3 -gt 10 ];
then
echo " Condition is false. "
fi
```

Step 4: Output

```
root@serverid:~# ./conditions4.sh
 Condition is true.
```

9. Shell Program to demonstrate the use of AND and OR operators in the if expression.

Step 1: Create an "**conditions5.sh**" script file using touch command

Step 2: Create a nano file to write the code

```
root@serverid:~# touch conditions5.sh
root@serverid:~# nano conditions5.sh
root@serverid:~# chmod +x conditions5.sh
```

Step 3: Write the Code in nano conditions5.sh script file

```
root@serverid: ~
  GNU nano 7.2                                                    conditions5.sh *
#!/bin/bash
# TRUE && FALSE || FALSE || TRUE
if [[ 10 -eq 10 && 5 -gt 4 || 3 -eq 4 || 3 -lt 6 ]];
then
echo "Condition is true."
fi
# TRUE && FALSE || FALSE
if [[ 8 -eq 8 && 8 -gt 10 || 9 -lt 5 ]];
then
echo "Condition is false"
fi
```

Step 4: Output

```
root@serverid:~# ./conditions5.sh
Condition is true.
```

10. Shell Program to demonstrate the use of nested if expression.

Step 1: Create an "**conditions6.sh**" script file using touch command

Step 2: Create a nano file to write the code

```
root@serverid:~# touch conditions6.sh
root@serverid:~# nano conditions6.sh
root@serverid:~# chmod +x conditions6.sh
```

Step 3: Write the Code in nano conditions6.sh script file

```
root@serverid: ~
  GNU nano 7.2                                                    conditions6.sh *
#!/bin/bash
#Nested if statement
if [ $1 -gt 50 ]
then
echo "Number is greater than 50."
if (( $1 % 2 == 0 ))
then
echo "and it is an even number."
fi
fi
```

Step 4: Output

```
root@serverid:~# ./conditions6.sh 60
Number is greater than 50.
and it is an even number.
```

11. Shell Program to demonstrate the use of if else in Bash Script.

Step 1: Create an "**conditions7.sh**" script file using touch command

Step 2: Create a nano file to write the code

```
root@serverid:~# touch conditions7.sh
root@serverid:~# nano conditions7.sh
root@serverid:~# chmod +x conditions7.sh
```

Step 3: Write the Code in nano conditions7.sh script file

```
root@serverid: ~
  GNU nano 7.2                                          conditions7.sh *
#!/bin/bash
#when the condition is true
if [ 10 -gt 3 ];
then
echo "10 is greater than 3."
else
echo "10 is not greater than 3."
fi
#when the condition is false
if [ 3 -gt 10 ];
then
echo "3 is greater than 10."
else
echo "3 is not greater than 10."
fi
```

Step 4: Output

```
root@serverid:~# ./conditions7.sh
10 is greater than 3.
3 is not greater than 10.
```

12. Shell Program to demonstrate the use of if else with logical operators in Bash script.

Step 1: Create an "**conditions8.sh**" script file using touch command

Step 2: Create a nano file to write the code

```
root@9a4a8a5799315e0:~# touch conditions8.sh
root@9a4a8a5799315e0:~# nano conditions8.sh
root@9a4a8a5799315e0:~# chmod +x conditions8.sh
```

Step 3: Write the Code in nano conditions8.sh script file

```
root@9a4a8a5799315e0: ~
  GNU nano 7.2                                          conditions8.sh *
#!/bin/bash
# When condition is true
# TRUE && FALSE || FALSE || TRUE
if [[ 10 -gt 9 && 10 == 9 || 2 -lt 1 || 25 -gt 20 ]];
then
echo "Given condition is true."
else
echo "Given condition is false."
fi
# When condition is false
#TRUE && FALSE || FALSE || TRUE
if [[ 10 -gt 9 && 10 == 8 || 3 -gt 4 || 8 -gt 8 ]];
then
echo "Given condition is true."
else
echo "Given condition is not true."
fi
```

Step 4: Output

```
root@9a4a8a5799315e0:~# ./conditions8.sh
Given condition is true.
Given condition is not true.
```

13. Shell Program to demonstrate the use of if else statement in a single line in Bash script.

Step 1: Create an "**conditions9.sh**" script file using touch command

Step 2: Create a nano file to write the code

```
root@9a4a8a5799315e0: ~
Given condition is not true.
root@9a4a8a5799315e0:~# touch conditions9.sh
root@9a4a8a5799315e0:~# nano conditions9.sh
root@9a4a8a5799315e0:~# chmod +x conditions9.sh
```

Step 3: Write the Code in nano conditions9.sh script file

```
root@9a4a8a5799315e0: ~
  GNU nano 7.2                                          conditions9.sh
#!/bin/bash
read -p "Enter a value:" value
if [ $value -gt 9 ];
then
echo "The value you typed is greater than 9.";
else
echo "The value you typed is not greater than 9.";
fi
```

Step 4: Output

```
root@9a4a8a5799315e0:~# ./conditions9.sh
Enter a value:24
The value you typed is greater than 9.
```

14. Shell Program to demonstrate the use of nested if-else statement in a Bash script.

Step 1: Create an "**conditions10.sh**" script file using touch command

Step 2: Create a nano file to write the code

```
root@9a4a8a5799315e0:~# touch conditions10.sh
root@9a4a8a5799315e0:~# nano conditions10.sh
root@9a4a8a5799315e0:~# chmod +x conditions10.sh
```

Step 3: Write the Code in nano conditions10.sh script file

```
root@9a4a8a5799315e0: ~
  GNU nano 7.2                                          conditions10.sh *
#!/bin/bash
read -p "Enter a value:" value
if [ $value -gt 9 ];
then
if [ $value -lt 11 ];
then
echo "$value>9, $value<11"
else
echo "The value you typed is greater than 9."
fi
else echo "The value you typed is not greater than 9."
fi
```

Step 4: Output

```
root@9a4a8a5799315e0:~# ./conditions10.sh
Enter a value:10
10>9, 10<11
```

15. Shell Program to demonstrate the use of else-if statement in a Bash script.

Step 1: Create an "**conditions11.sh**" script file using touch command

Step 2: Create a nano file to write the code

```
root@9a4a8a5799315e0:~# touch conditions11.sh
root@9a4a8a5799315e0:~# nano conditions11.sh
root@9a4a8a5799315e0:~# chmod +x conditions11.sh
```

Step 3: Write the Code in nano conditions11.sh script file

```
root@9a4a8a5799315e0: ~
  GNU nano 7.2                                          conditions11.sh *
#!/bin/bash
read -p "Enter a number of quantity:" num
if [ $num -gt 100 ];
then
echo "Eligible for 10% discount"
elif [ $num -lt 100 ];
then
echo "Eligible for 5% discount"
else
echo "Lucky Draw Winner"
echo "Eligible to get the item for free"
fi
```

Step 4: Output

```
root@9a4a8a5799315e0:~# ./conditions11.sh
Enter a number of quantity:110
Eligible for 10% discount
```

16. Shell Program to demonstrate the use of multiple conditionswith else-if statement in Bash script.

Step 1: Create an "**conditions12.sh"** script file using touch command

Step 2: Create a nano file to write the code

```
root@9a4a8a5799315e0:~# touch conditions12.sh
root@9a4a8a5799315e0:~# nano conditions12.sh
root@9a4a8a5799315e0:~# chmod +x conditions12.sh
```

Step 3: Write the Code in nano conditions12.sh script file

```
root@9a4a8a5799315e0: ~
  GNU nano 7.2                                   conditions12.sh *
#!/bin/bash
read -p "Enter a number of quantity:" num
if [ $num -gt 200 ];
then
echo "Eligible for 20% discount"
elif [[ $num == 200 || $num == 100 ]];
then
echo "Lucky Draw Winner"
echo "Eligible to get the item for free"
elif [[ $num -gt 100 && $num -lt 200 ]];
then
echo "Eligible for 10% discount"
elif [ $num -lt 100 ];
then
echo "No discount"
fi
```

Step 4: Output

```
root@9a4a8a5799315e0:~# ./conditions12.sh
Enter a number of quantity:100
Lucky Draw Winner
Eligible to get the item for free
```