

Bash Arithmetic Operators Code:

```
jith123@a6cf8aee28ce504: ~
GNU nano 7.2 bash_arithmetic_operators
#!/bin/bash
x=8
y=2
echo "x=8, y=2"
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 "Exponentiation 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, x= "
(( x /= 5 ))
echo $x
echo "Remainder of Dividing x by 5, x="
(( x %= 5 ))
echo $x
```

Output:

```
jith123@a6cf8aee28ce504: ~
./bash_arithmetic_operators: line 5: @: command not found
./bash_arithmetic_operators: line 6: + : syntax error: operand expected (error token is "+ ")
./bash_arithmetic_operators: line 7: @: command not found
./bash_arithmetic_operators: line 8: - : syntax error: operand expected (error token is "- ")
./bash_arithmetic_operators: line 9: @: command not found
./bash_arithmetic_operators: line 10: * : syntax error: operand expected (error token is "** ")
./bash_arithmetic_operators: line 11: @: command not found
./bash_arithmetic_operators: line 12: / : syntax error: operand expected (error token is "/" ")
./bash_arithmetic_operators: line 13: @: command not found
./bash_arithmetic_operators: line 14: ** : syntax error: operand expected (error token is "*** ")
./bash_arithmetic_operators: line 15: @: command not found
./bash_arithmetic_operators: line 16: % : syntax error: operand expected (error token is "% ")
./bash_arithmetic_operators: line 17: @: command not found
./bash_arithmetic_operators: line 18: syntax error near unexpected token `('
./bash_arithmetic_operators: line 18: `@ (( x += 5 )) '
jith123@a6cf8aee28ce504:~$ nano bash_arithmetic_operators
jith123@a6cf8aee28ce504:~$ ./bash_arithmetic_operators
x=8, y=2
Addition of x & y
10
Subtraction of x & y
6
Multiplication of x & y
16
Division of x by y
4
Exponentiation of x,y
64
Modular Division of x,y
0
Incrementing x by 5, then x=
13
Decrementing x by 5, then x=
8
Multiply of x by 5, then x=
40
Dividing x by 5, x=
8
Remainder of Dividing x by 5, x=
3
jith123@a6cf8aee28ce504:~$
```

If-else statements with eq ,gt,lt,&&,||, operators :

Output of if-else programs in same file:

```
jith123@a6cf8aee28ce504: ~  
jith123@a6cf8aee28ce504:~$ nano if_else  
jith123@a6cf8aee28ce504:~$ chmod +x if_else  
jith123@a6cf8aee28ce504:~$ ./if_else  
Enter number to know whether it is greater than 125 or not : 30  
true condition  
10 is greater than 3.  
3 is less than 10.  
10 is equal to 10.  
Conditions are true  
Condition is true.  
Condition is true.  
./if_else: line 103: [: -gt: unary operator expected  
./if_else: line 113: unexpected EOF while looking for matching `"  
jith123@a6cf8aee28ce504:~$ nano if_else  
jith123@a6cf8aee28ce504:~$ nano if_else  
jith123@a6cf8aee28ce504:~$ ./if_else  
Enter number to know whether it is greater than 125 or not : 20  
true condition  
10 is greater than 3.  
3 is less than 10.  
10 is equal to 10.  
Conditions are true  
Condition is true.  
Condition is true.  
end of learning if loops  
jith123@a6cf8aee28ce504:~$
```

Actual code:

Value greater than 125:

```
Select jith123@a6cf8aee28ce504: ~  
GNU nano 7.2 if_else  
#!/bin/bash  
read -p "Enter number to know whether it is greater than 125 or not : " number  
if [ $number -gt 125 ]  
then  
echo "Value is greater than 125"  
fi
```

Numerical comparison:

```
#numerical comparison
#!/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."
```

String comparison:

```
#!/bin/bash
#string comparison
#!/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
```

Logical OR:

```
#logical or
#!/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
```

Logical And

```
#use logical and
#!/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
```

Nested if:

```
#nested if's
#!/bin/bash
# Nested if statement
if [ 20 -gt 50 ];
then
    echo "Number is greater than 50."
    if (( 20 % 2 == 0 ));
    then
        echo "and it is an even number."
    fi
fi
#end
echo "end of learning if loops"
```

Combining Logical operators:

```
#!/bin/bash
# combining logical operators
# 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
```

All codes in one file:

```
jith123@a6cf8ee28ce504: ~
GNU nano 7.2 if else
#!/bin/bash
echo "3 is less than 10."
if [ 3 -lt 10 ];
then
    echo "3 is less than 10."
fi
# if condition (less 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
#use logical and
#!/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
#use logical or
#!/bin/bash
```

```
jith123@a6cf8bee28ce504: ~
GNU nano 7.2 if else
#logical or
#!/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

#combining logical operators
#!/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

#nested if's
#!/bin/bash
# Nested if statement
if [ 20 -gt 50 ];
then
    echo "Number is greater than 50."
    if (( 20 % 2 == 0 ));
    then
        echo "and it is an even number."
    fi
fi
echo "end of learning if loops"
```

```
jith123@a6cf8bee28ce504: ~
GNU nano 7.2 if else

# FALSE || FALSE
if [ "mylife" == "yourlife" ] || [ 3 -gt 10 ];
then
    echo "Condition is false."
fi

#combining logical operators
#!/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

#nested if's
#!/bin/bash
# Nested if statement
if [ 20 -gt 50 ];
then
    echo "Number is greater than 50."
    if (( 20 % 2 == 0 ));
    then
        echo "and it is an even number."
    fi
fi
echo "end of learning if loops"
```

Bash if -else Example-1:

```
GNU nano 7.2 bash_if-else ex-1
#!/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
```

Bash if -else Example-2:

```
GNU nano 7.2 bash_if-else ex-2
#!/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
```

Bash if -else statement in single line E-3:

```
jith123@a6cf8ee28ce504: ~
GNU nano 7.2 bash_if-else ex-3
#!/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
```

Bash Nested if -else Example-4:

```
GNU nano 7.2 bash_if-else ex-4
#!/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
```

Bash else-if Example-5:

```
GNU nano 7.2 bash if-else ex-5
#!/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
```

Bash else-if Example-6:

```
GNU nano 7.2 bash if-else ex-6
#!/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
```

Output of all Bash if -else programs from Ex-1 to 6:

```
l1th123@a6cf8aee28ce504:~$ ./bash_if-else_ex-1
10 is greater than 3.
3 is not greater than 10.
l1th123@a6cf8aee28ce504:~$ ./bash_if-else_ex-2
Given condition is true.
Given condition is not true.
l1th123@a6cf8aee28ce504:~$ ./bash_if-else_ex-3
Enter a value: 47
The value you typed is greater than 9.
l1th123@a6cf8aee28ce504:~$ ./bash_if-else_ex-4
Enter a value:30
The value you typed is greater than 9.
l1th123@a6cf8aee28ce504:~$ ./bash_if-else_ex-5
Enter a number of quantity:10
Eligible for 5% discount
l1th123@a6cf8aee28ce504:~$ ./bash_if-else_ex-6
Enter a number of quantity:25
No discount
```