

Name : Md. Sohag Hossain  
ID : IT-17060  
Lab report no : 05  
**lab report name : Introduction To Python**

### **objectives:**

Python language supports the following types of operators.

- Arithmetic Operators
- Comparison (Relational) Operators
- Assignment Operators
- Logical Operators
- Bitwise Operators
- Membership Operators
- Identity Operators

### **Theory:**

Python is an interpreted, object-oriented, high-level programming language with dynamic semantics. Its high-level built in data structures, combined with dynamic typing and dynamic binding, make it very attractive for Rapid Application Development, as well as for use as a scripting or glue language to connect existing components together. Python's simple, easy to learn syntax emphasizes readability and therefore reduces the cost of program maintenance. Python supports modules and packages, which encourages program modularity and code reuse. The Python interpreter and the extensive standard library are available in source or binary form without charge for all major platforms, and can be freely distributed.

### **Python Arithmetic Operators:**

Assume variable a holds 10 and variable b holds 20, then –

| Operator         | Description   | Example   |
|------------------|---|---|
| + Addition       | Adds values on either side of the operator.   | $a + b = 30$  |
| - Subtraction    | Subtracts right hand operand from left hand operand.  | $a - b = -10$   |
| * Multiplication | Multiplies values on either side of the operator  | $a * b = 200$   |
| / Division       | Divides left hand operand by right hand operand   | $b / a = 2$   |
| % Modulus        | Divides left hand operand by right hand operand and returns remainder   | $b \% a = 0$  |
| ** Exponent      | Performs exponential (power) calculation on operators   | $a ** b = 10$ to the power 20   |
| //               | Floor Division - The division of operands where the result is the quotient in which the digits after the decimal point are removed. But if one of the operands is negative, the result is floored, i.e., rounded away from zero (towards negative infinity) – | $9 // 2 = 4$ and $9.0 // 2.0 = 4.0$ , - $11 // 3 = -4$ , - $11.0 // 3 = -4.0$ |

## Python Comparison Operators:

Assume variable a holds 10 and variable b holds 20, then –

| Operator                     | Description   | Example   |
|------------------------------|---|---|
| =                            | Assigns values from right side operands to left side operand                            | <code>c = a + b</code><br>assigns value of <code>a + b</code> into <code>c</code> |
| <code>+=</code> Add AND      | It adds right operand to the left operand and assign the result to left operand         | <code>c += a</code> is equivalent to <code>c = c + a</code>                       |
| <code>-=</code> Subtract AND | It subtracts right operand from the left operand and assign the result to left operand  | <code>c -= a</code> is equivalent to <code>c = c - a</code>                       |
| <code>*=</code> Multiply AND | It multiplies right operand with the left operand and assign the result to left operand | <code>c *= a</code> is equivalent to <code>c = c * a</code>                       |
| <code>/=</code> Divide AND   | It divides left operand with the right operand and assign the result to left operand    | <code>c /= a</code> is equivalent to <code>c = c / a</code>                       |

|                               |  |                                     |
|-------------------------------|--|-------------------------------------|
| <b>%= Modulus<br/>AND</b>     | It takes modulus using two operands and assign the result to left operand                  | c %= a is equivalent to c = c % a   |
| <b>**= Exponent<br/>AND</b>   | Performs exponential (power) calculation on operators and assign value to the left operand | c **= a is equivalent to c = c ** a |
| <b>//= Floor<br/>Division</b> | It performs floor division on operators and assign value to the left operand               | c //= a is equivalent to c = c // a |

## **Python Bitwise Operators**

Bitwise operator works on bits and performs bit by bit operation. Assume if a = 60; and b = 13; Now in the binary format their values will be 0011 1100 and 0000 1101 respectively. Following table lists out the bitwise operators supported by Python language with an example each in those, we use the above two variables (a and b) as operands –

a = 0011 1100

b = 0000 1101

-----

a&b = 0000 1100

a|b = 0011 1101

a^b = 0011 0001

~a = 1100 0011

## Arithmetic Operators:

Just as any other programming languages, the addition, subtraction, multiplication, and division operators can be used with numbers.

### Code:

```
x = 15
y = 4
print('x + y =', x+y)
print('x - y =', x-y)
print('x * y =', x*y)
print('x / y =', x/y)
print('x // y =', x//y)
print('x ** y =', x**y)
```

### output:

---

```
"C:\Users\sohag\PycharmI
```

```
x + y = 19
```

```
x - y = 11
```

```
x * y = 60
```

```
x / y = 3.75
```

```
x // y = 3
```

```
x ** y = 50625
```

Another operator available is the modulo (%) operator, which returns the integer remainder of the division. dividend % divisor = remainder.

**Code:**

```
remainder = 11 % 3  
print("remainder : ", remainder)
```

**output:**

---

```
"C:\Users\sohag\PycharmI  
remainder : 2
```

Using two multiplication symbols makes a power relationship.

**Code:**

```
squared = 7 ** 2  
cubed = 2 ** 3  
print("squared : ", squared)  
print("cubed : ", cubed)
```

**output:**

```
"C:\Users\sohag\PycharmI
squared : 49
cubed : 8
```

### Comparison operators:

Comparison operators are used to compare values. It returns either `True` or `False` according to the condition.

#### Code:

```
x = 10
y = 12
print('x > y is',x>y)
print('x < y is',x<y)
print('x == y is',x==y)
print('x != y is',x!=y)
print('x >= y is',x>=y)
print('x <= y is',x<=y)
```

#### output:

```
"C:\Users\sohag\PycharmI
x > y is False
x < y is True
x == y is False
x != y is True
x >= y is False
x <= y is True
```

## Logical Operators in Python:

### Code:

```
x = True
y = False
print('x and y is',x and y)
print('x or y is',x or y)
print('not x is',not x)
```

### output:

```
"C:\Users\sohag\PycharmF
x and y is False
x or y is True
not x is False
```

## Identity operators in Python:

### Code:

```
x1 = 5
y1 = 5
x2 = 'Hello'
y2 = 'Hello'
x3 = [1,2,3]
y3 = [1,2,3]
print(x1 is not y1)
print(x2 is y2)
print(x3 is y3)
```

### output:



---

```
"C:\Users\sohag\Py
False
True
False
```

### **Membership operators:**

#### **Code:**

```
x = 'Hello world'
y = {1:'a',2:'b'}
print('H' in x)
print('hello' not in x)
print(1 in y)
print('a' in y)
```

#### **output:**

---

```
"C:\Users\sohag\PycharmF
True
True
True
False
```

## Using Operators with Strings:

Python supports concatenating strings using the addition operator:

### Code:

```
helloworld = "hello" + " " + "world"  
print(helloworld)
```

### output:

---

```
"C:\Users\sohag\PycharmI  
hello world"
```

## Using Operators with Lists

Lists can be joined with the addition operators:

### Code:

```
even_numbers = [2,4,6,8]  
odd_numbers = [1,3,5,7]  
all_numbers = odd_numbers + even_numbers  
print("All Numbers : ",all_numbers)
```

### output:

---

```
"C:\Users\sohag\PycharmProjects\computer  
All Numbers : [1, 3, 5, 7, 2, 4, 6, 8]
```

### **Conclusion:**

In this tutorial, I learn to run Python on my computer. Once I do that, I shall also write our first Python program.

Python is a cross-platform programming language, which means that it can run on multiple platforms like Windows, macOS, Linux, and has even been ported to the Java and .NET virtual machines. It is free and open-source.