#OPERATORS IN PYTHON

1) Arithematic Operators(+, -, *, /, %)

2) Assignment Operator(=, +=, -=, *=, /=)

3) Relational Operator(>, <, <=, >=, ==)

4) Logical Operators(and, or)

#WRITING BIGGER CODES

<1> Sum of 2 numbers, taking input from the user.

```
x=int(input("Enter 1st number"))
y=int(input("Enter 2nd number"))
z=x+y
print("Sum=",z)
z+=4            #Assignment operator "+=" is used, z+=4 is nothing but z=z+4, similarly you can use
                "-=", "*=", "/=" operators also
print(z)
```

<2> **Swapping two numbers with 2 diffrent methods:**

<2.1>**Using 3rd Variable**

```
x=int(input("Enter 1st number"))
y=int(input("Enter 2nd number"))
temp=x
x=y
```

```
y=temp

print("Your swapped numbers are",x,y)
```

<2.2>**Without using 3rd variable**

```
x=int(input("Enter 1st number"))

y=int(input("Enter 2nd number"))

x,y=y,x             #This is a unique and simple feature in python for swapping
                    numbers without using 3rd variable

print("Your swapped numbers are",x,y)
```

<3> **Taking String and float as input from user**

```
x=input("Enter any string")

y=float(input("Enter any decimal number"))


print(x)

print(y)

print(type(x)) #belongs to class String ; output- <class 'string'>

print(type(y)) #belong to class float ; output- <class 'float'>
```

#**CONDITIONAL STATEMENTS**

There are 3 conditional statements **- if, else, elif.**

<1> Using **if** statement

```
x=4
```

```
y=3
if x>y:
	print(x,"is greater")
print("Bye")
```

<2> Using **else** statement

```
x=4
y=3
if x>y:						#Relational operator(">") is used here.
	print(x,"is greater")
else:
	print(y,"is greater")
```

<3> Using **elif** statement : elif is nothing but "else if", it is used to check more than 2 conditions.

```
x=int(input("Enter any number between 0 and 4"))
if x==1:
	print("one")
elif x==2:					#Relational operator("==") is used here.
	print("two")
elif x==3:
	print("three")
else:
	print("Invalid Input")
```

#LOOPING STATEMENTS

keyword "range" syntax = range(start,stop,increment/decrement)

<1> Using **"for" loop** statement

```
for i in range(1,11):      # range(1,11) will take starting value as 1, not mentioning increment/dec
                           value means it will increment the value by 1(default)
        print(i)           #output - 1
                                     2
                                     3
                                     4
                                     5
                                     6
                                     7
                                     8
                                     9
                                     10        #Last printed value is = 10, therefore arguement in range
takes value till 'n-1', here n=11
```

<2> **Example 2 (for loop)**

```
for i in range(0,11,2):
        print(i,end=" ") # end="" will print the values of 'i' on the same line instead of going to new
line.
                           #output - 0 2 4 6 8 10
```

<3> **Using "while loop" statement**

```
i=1                     #initialization
while i<=5:             #condition
        print(i,end=" ")
        i=i+1           #increment/decrement
                #output- 1 2 3 4 5
```

<4> **Nested "while loop"**

```
i=1
while i<=3:
        print("Hello",end=" ")
        j=1
        while j<=4:
                print("Rocks",end=" ")
                j=j+1
        i=i+1
        print()   #Used for new line

#output - Hello Rocks Rocks Rocks Rocks
          Hello Rocks Rocks Rocks Rocks
          Hello Rocks Rocks Rocks Rocks
```

<5> **Printing Patterns using nested "for loop"**

```
#
# #
# # #
# # # #
```

```
for i in range(1,5):
        for j in range(1, i+1):
                print("#",end=" ")
        print()
```

<2> **Print the Pattern**

```
1234
1234
1234
1234
```

```
for i in range(4):
        for j in range(4):
                print(j+1,end="")
        print()
```