PRESENTER

Ola El-Shiekh

AI Researcher

mystro

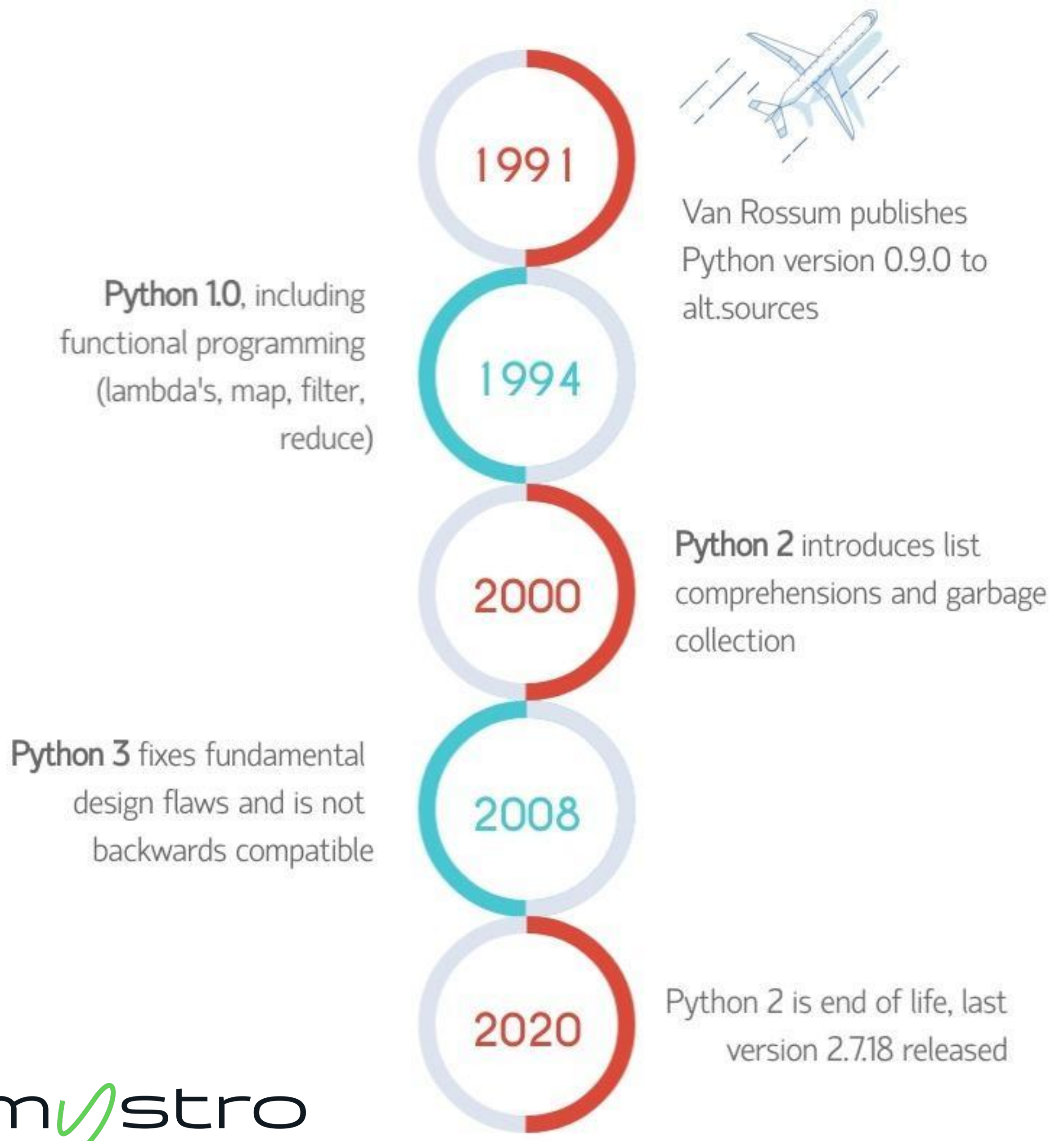
INTRODUCTION TO AI & SOCIAL ENTREPRENEURSHIP

mystro

AGENDA

1. Using python in Artificial Intelligence
2. Data Types
3. Variables
4. Control structures
5. If - else if - nested if
6. For loop
7. While loop

PYTHON HISTORY



1991

Van Rossum publishes Python version 0.9.0 to alt.sources

1994

Python 1.0, including functional programming (lambda's, map, filter, reduce)

2000

Python 2 introduces list comprehensions and garbage collection

2008

Python 3 fixes fundamental design flaws and is not backwards compatible

2020

Python 2 is end of life, last version 2.7.18 released

PYTHON NOW

- High-level
- General-purpose
- Simple Syntax
- Big Demand on Market

PYTHON PROGRAMMING

- Artificial Intelligence

- Web Development

APPS BUILT IN PYTHON

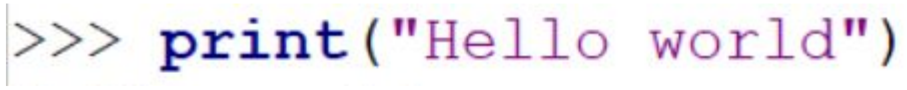
- Netflex
- Spotify
- Uber
- Instagram

I ENVIRONMDE – INTEGRATED DEVELOPMENTENT

- Contains : Text Editor, Compiler, Linker, Debugger, ...
- Functions.
  - ✔ Writing source code
  - ✔ Debugging
  - ✔ Tracing value of a variable is possible.

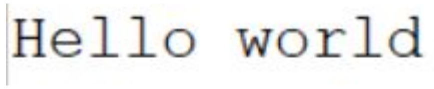
LET’S WRITE OUR FIRST PROGRAM

Printing Hello World !



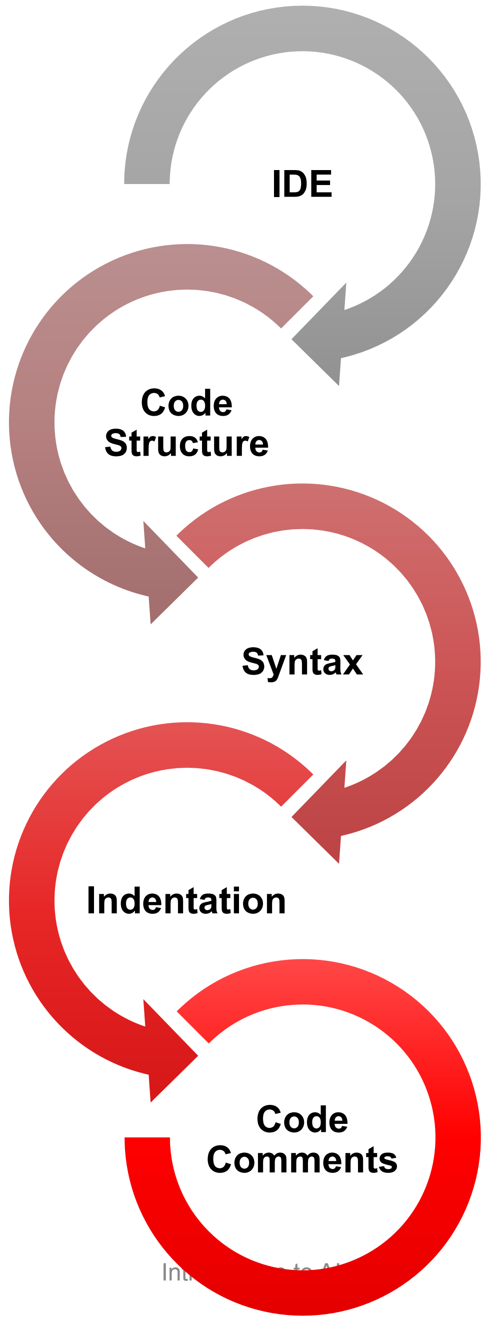
```
>>> print("Hello world")
```

Output



```
Hello world
```

BASIC CONCEPTS



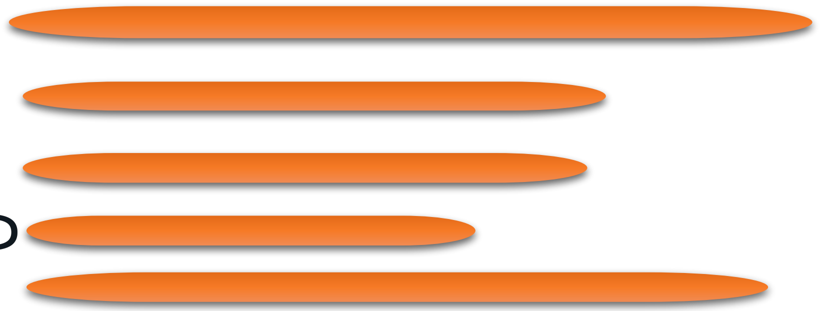
CODE STRUCTURE

- **Statement**

- Statement is one of **expression** unit for computer to understand our thoughts and to make it execute specific action.
- Computer program is a collection of a number of these statements.

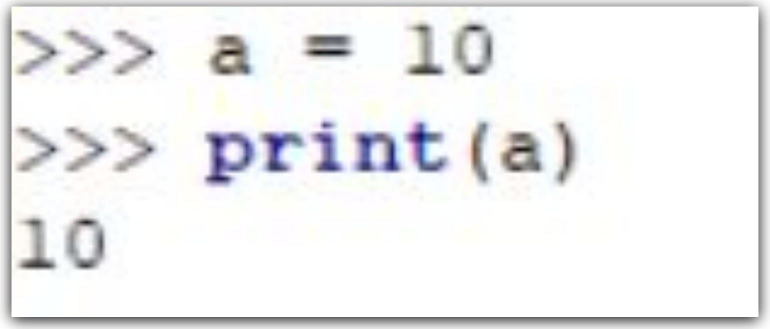
- **Block**

- In computer language, statements that mean similar property or action are expressed in one collection, and it's called block structure.
- In other words, it handles a number of statements by binding these into one block



VARIABLES

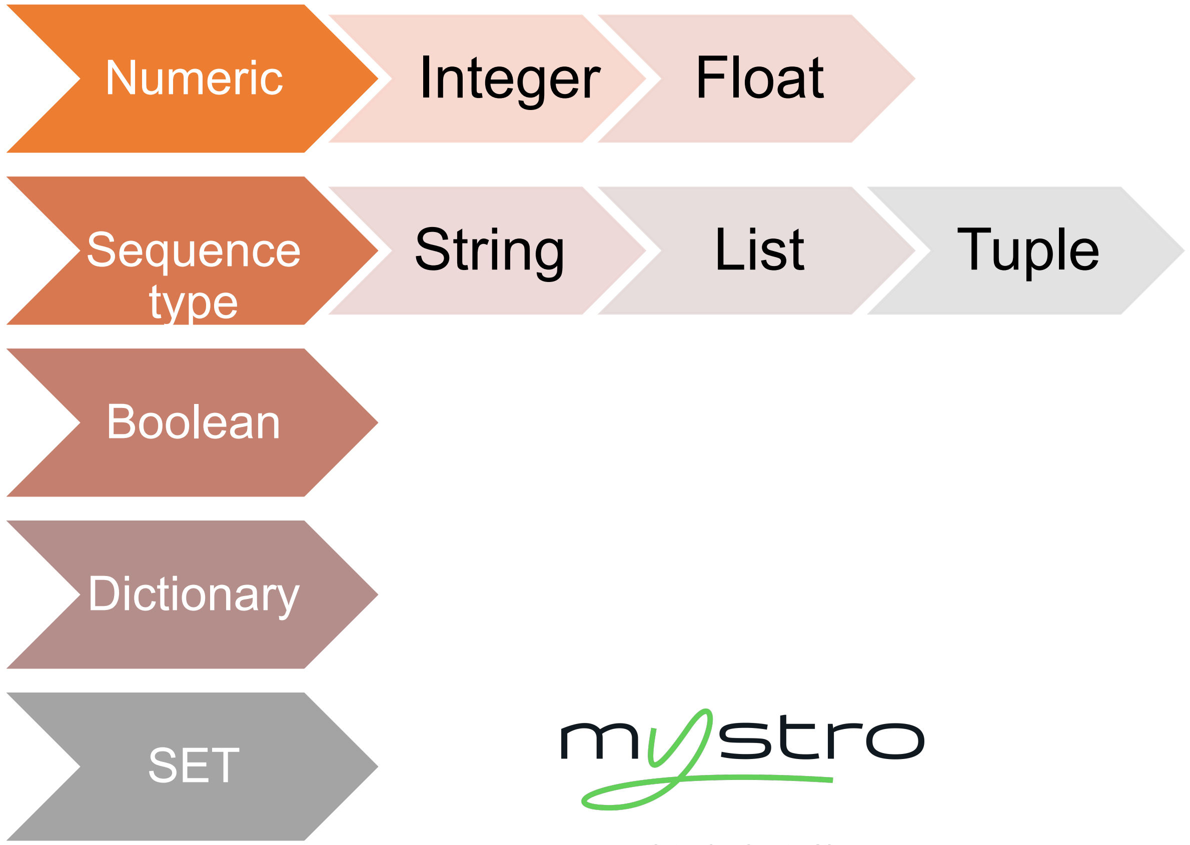
- Python Variable is containers which store values.
- Dynamic – No need for declaration
- Python Variable is containers which store values.



```
>>> a = 10
>>> print(a)
10
```

DATA TYPES

- **Variables can store data of different types**



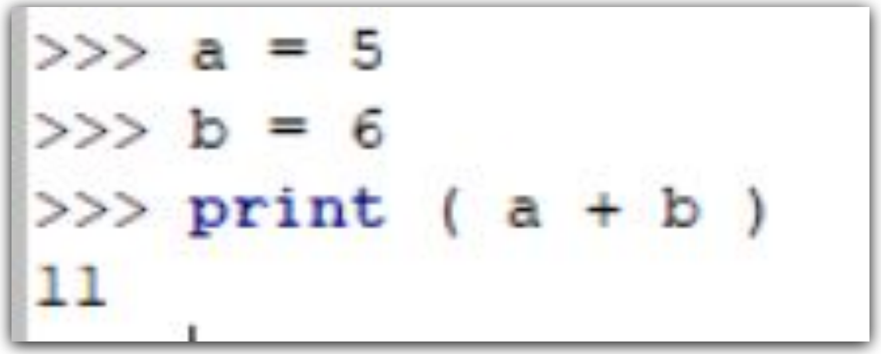
- containers which store values

ARITHMETIC OPERATION

- Addition (sum; ‘+’)
- Subtraction (difference; -)
- Multiplication (product; ×) (*)
- Division (÷) (/)
- Division reminder (Modulus ; %)
- Exponentiation (**)
- Floor division

ADDITION OPERATOR

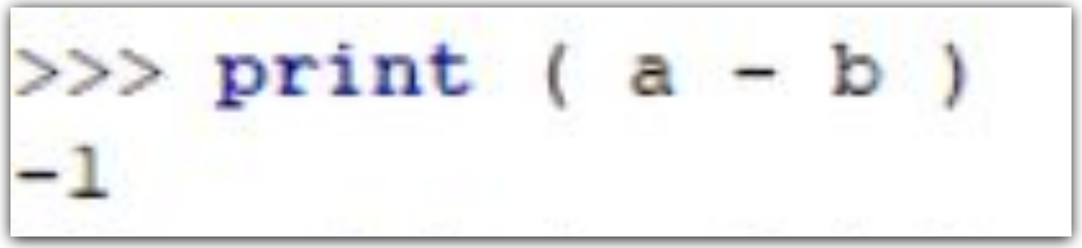
- In Python, (+) is the addition operator. It is used to add 2 values.



```
>>> a = 5
>>> b = 6
>>> print ( a + b )
11
```

SUBSTRACTION OPERATOR

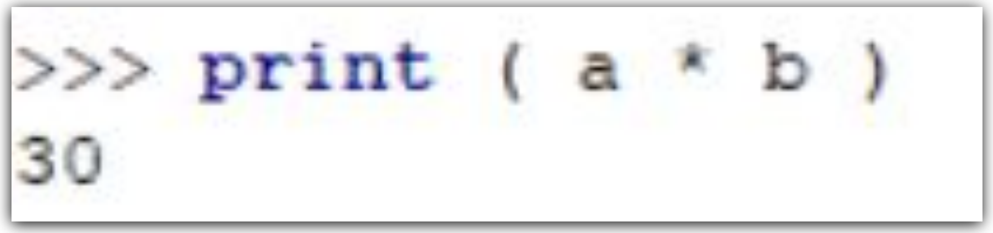
- In Python, – is the subtraction operator. It is used to subtract the second value from the first value..



```
>>> print ( a - b )
-1
```

MULIPLICATION OPERATOR

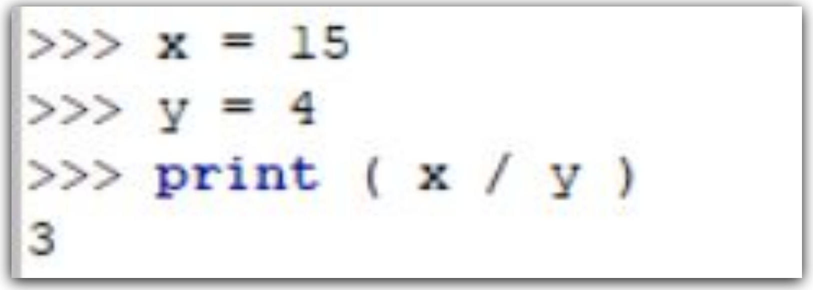
- In Python, * is the multiplication operator. It is used to find the product of 2 values.



```
>>> print ( a * b )
30
```

DIVISION OPERATOR

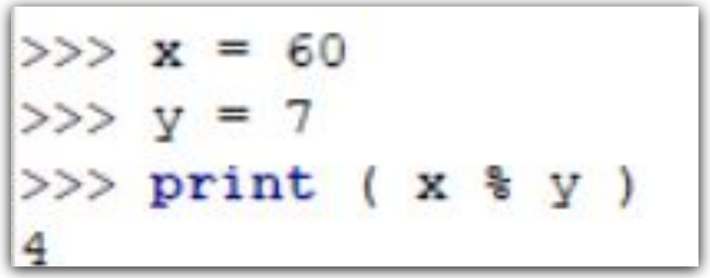
- In Python, / is the division operator. It is used to find the quotient when first operand is divided by the second.



```
>>> x = 15
>>> y = 4
>>> print ( x / y )
3
```

MODULUS OPERATOR

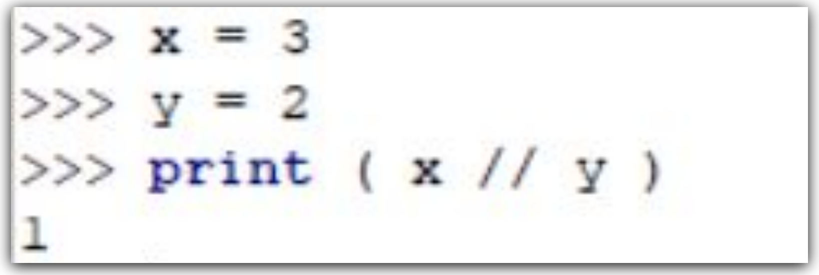
- In Python, % is the modulus operator. It is used to find the remainder when first operand is divided by the second.



```
>>> x = 60
>>> y = 7
>>> print ( x % y )
4
```

FLOOR DIVISION

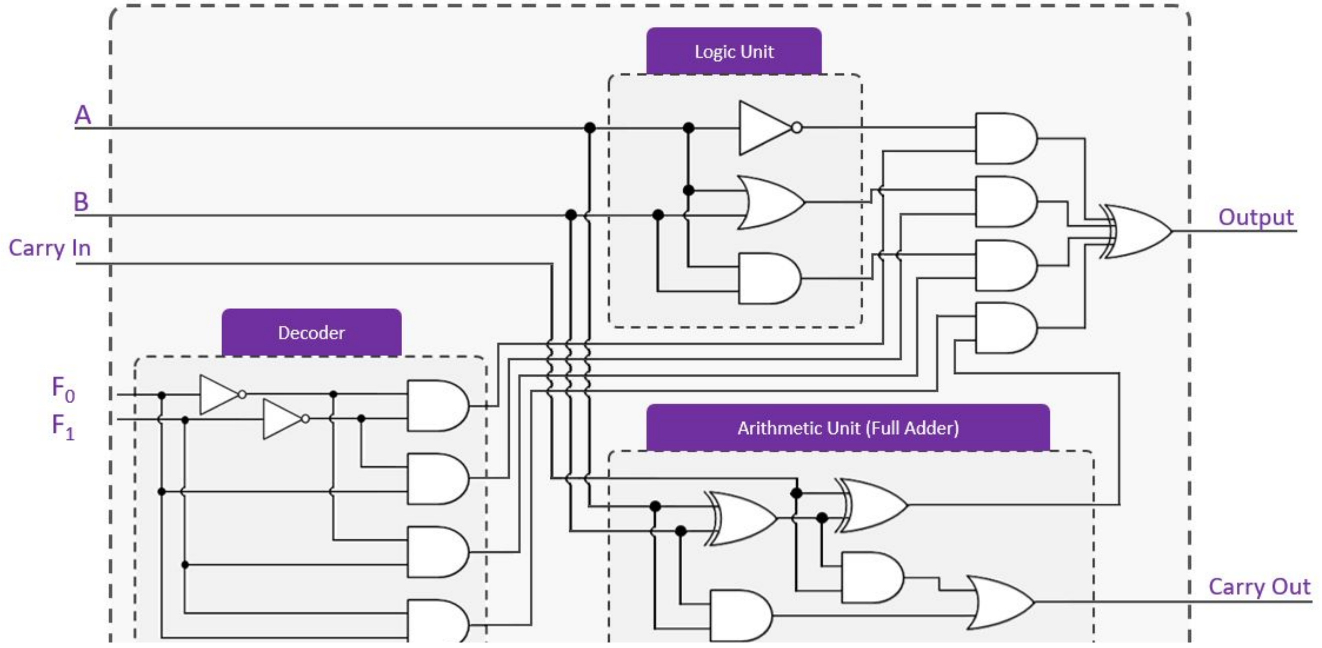
- In Python, // is used to conduct the floor division. It is used to find the floor of the quotient when first operand is divided by the second.



```
>>> x = 3
>>> y = 2
>>> print ( x // y )
1
```

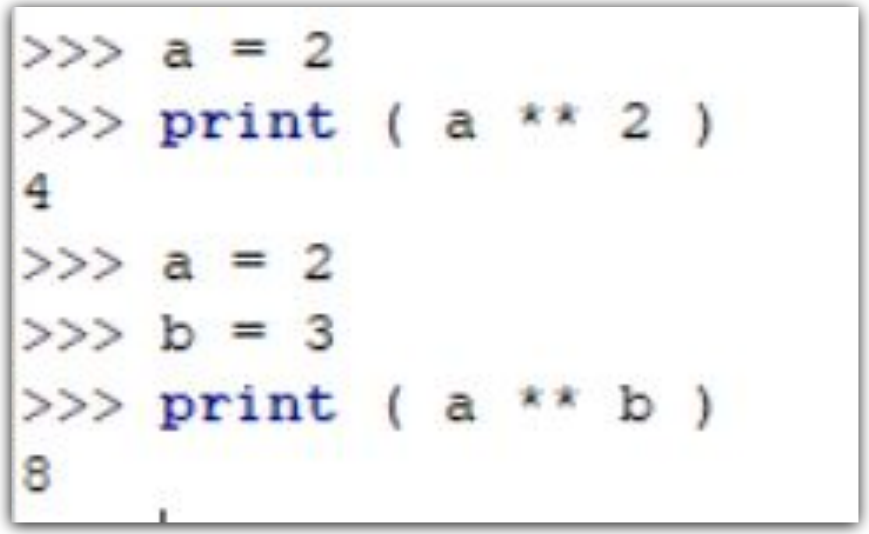
LOGIC GATES

- A logic gate is a simple switching circuit that determines whether an input pulse can pass through to the output in digital circuits.



EXPONENTIATION OPERATOR

- In Python, ** is the exponentiation operator. It is used to raise the first operand to power of second.



```
>>> a = 2
>>> print ( a ** 2 )
4
>>> a = 2
>>> b = 3
>>> print ( a ** b )
8
```

TYPES OF LOGIC GATES

- AND
- OR
- NOT
- NOR
- NAND
- XOR
- XNOR

TRUTH TABLE

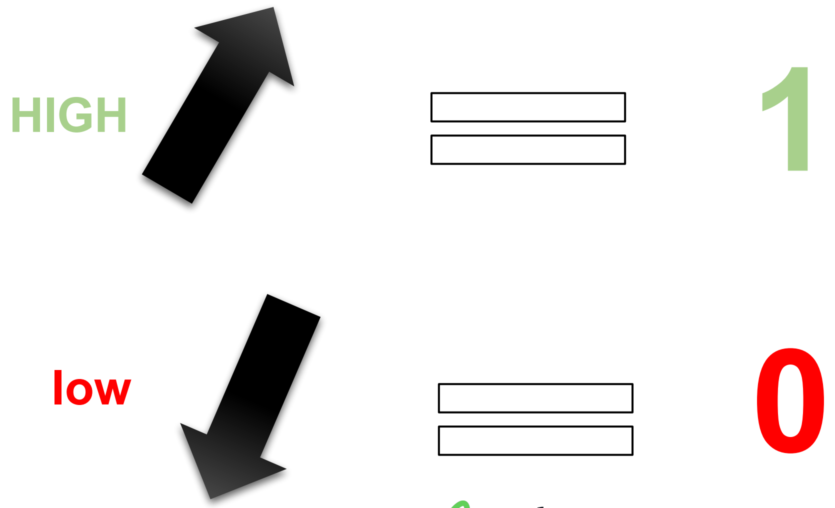
- Boolean algebra is a type of logical algebra in which symbols represent logic levels.
- 1 Means High
- 0 Means Low

Cases	Input 1	Input 2	Output
CASE 1	Low (0)	Low (0)	Based on the Logic Gate used !
CASE 2	Low (0)	High (1)	
CASE 3	High (1)	Low (0)	
CASE 4	High (1)	High (1)	

BOOLEAN ALGEBRA

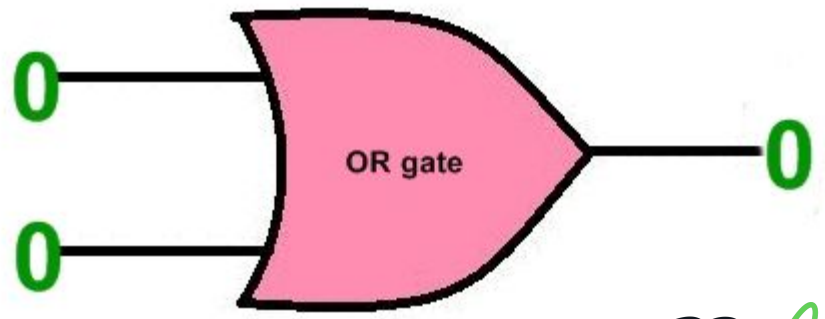
BOOLEAN ALGEBRA

- Boolean algebra is a type of logical algebra in which symbols represent logic levels.
- 1 Means High
- 0 Means Low



(OR) LOGIC GATES

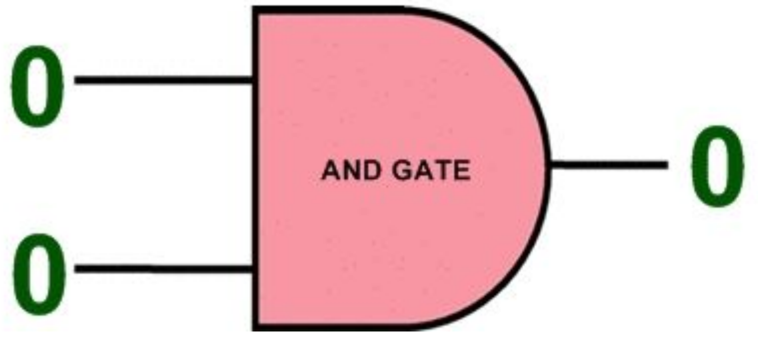
- **The OR gate gives an output of 1 if either of the two inputs are 1, it gives 0 otherwise.**



Cases	Input 1	Input 2	Output
CASE 1	0	0	0
CASE 2	0	1	1
CASE 3	1	0	1
CASE 4	1	1	1

(AND) LOGIC GATES

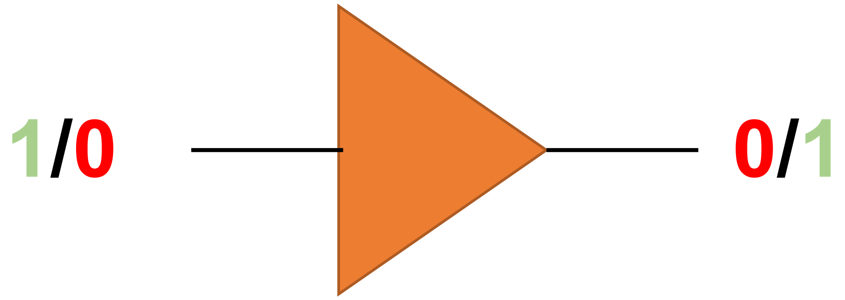
- **The AND gate gives an output of 1 if both the two inputs are 1, it gives 0 otherwise.**



Cases	Input 1	Input 2	Output
CASE 1	0	0	0
CASE 2	0	1	0
CASE 3	1	0	0
CASE 4	1	1	1

(NOT) LOGIC GATES

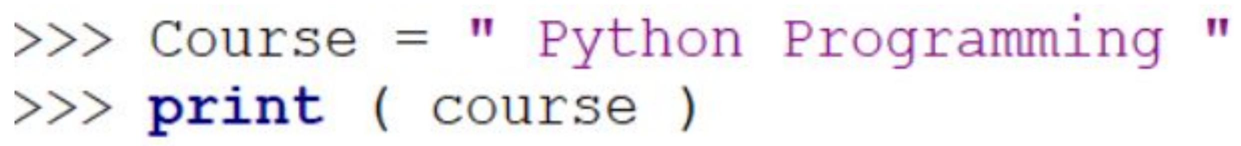
- It acts as an inverter. It takes only one input. If the input is given as 1, it will invert the result as 0 and vice-versa.



Cases	Input	Output
CASE 1	0	1
CASE 2	1	0

VARIABLES IN PYTHON

- Once an object is assigned to a variable, it can be referred to by that name. We can say that Variable in Python is containers that store values.
- The value stored in a variable can be changed during program execution.



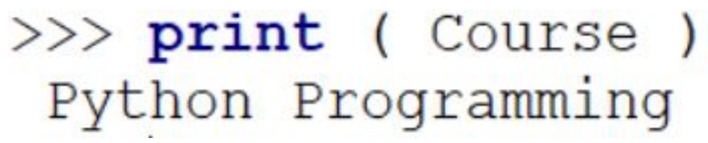
```
>>> Course = " Python Programming "
>>> print ( course )
```

VARIABLES

- Python Variable is containers that
- store values.
- Python is not “statically typed”.
- We do not need to declare variables before using them or declare their type.
- A variable is created the moment we first assign a value to it.
- A Python variable is a name given to a memory location.
- It is the basic unit of storage in a program.

VARIABLES IN PYTHON

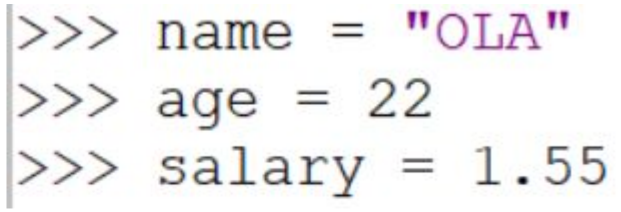
- A Variables in Python is only a name given to a memory location, all the operations done on the variable effects that memory location.



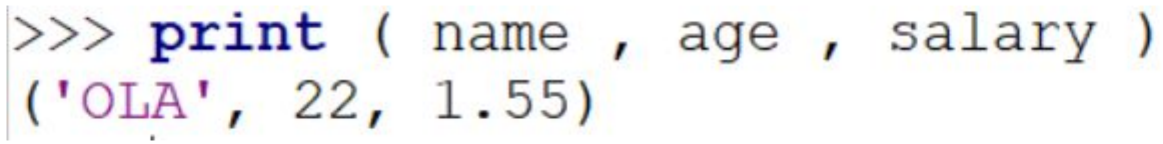
```
>>> print ( Course )
 Python Programming
```

VARIABLES IN PYTHON

- Assigning different values like name (String value) , age (integer value) and salary (float number).



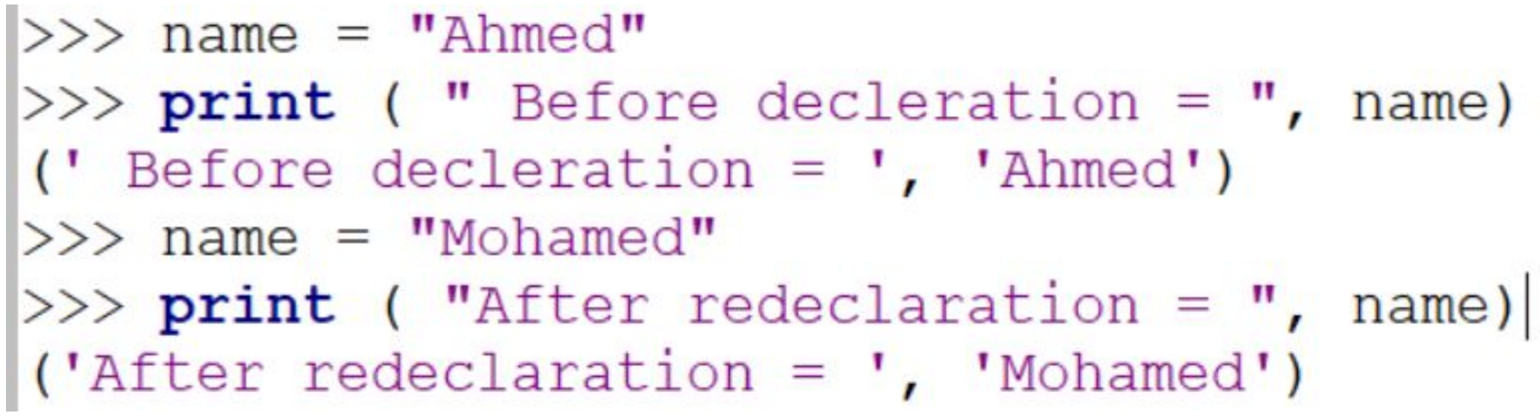
```
>>> name = "OLA"
>>> age = 22
>>> salary = 1.55
```



```
>>> print ( name , age , salary )
('OLA', 22, 1.55)
```

VARIABLES IN PYTHON

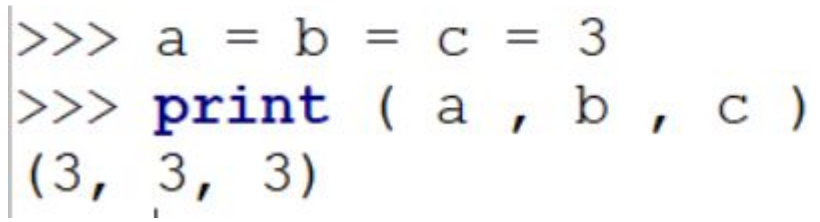
- We can re-declare the Python variable once we have declared the variable already..



```
>>> name = "Ahmed"
>>> print ( " Before decleration = ", name)
(' Before decleration = ', 'Ahmed')
>>> name = "Mohamed"
>>> print ( "After redeclaration = ", name)
('After redeclaration = ', 'Mohamed')
```

VARIABLES IN PYTHON

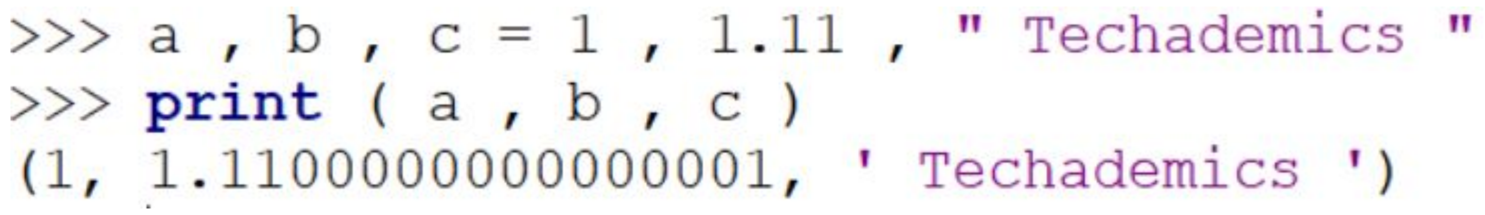
- Python allows assigning a single value to several variables simultaneously with “=” operators.



```
>>> a = b = c = 3
>>> print ( a , b , c )
(3, 3, 3)
```

VARIABLES IN PYTHON

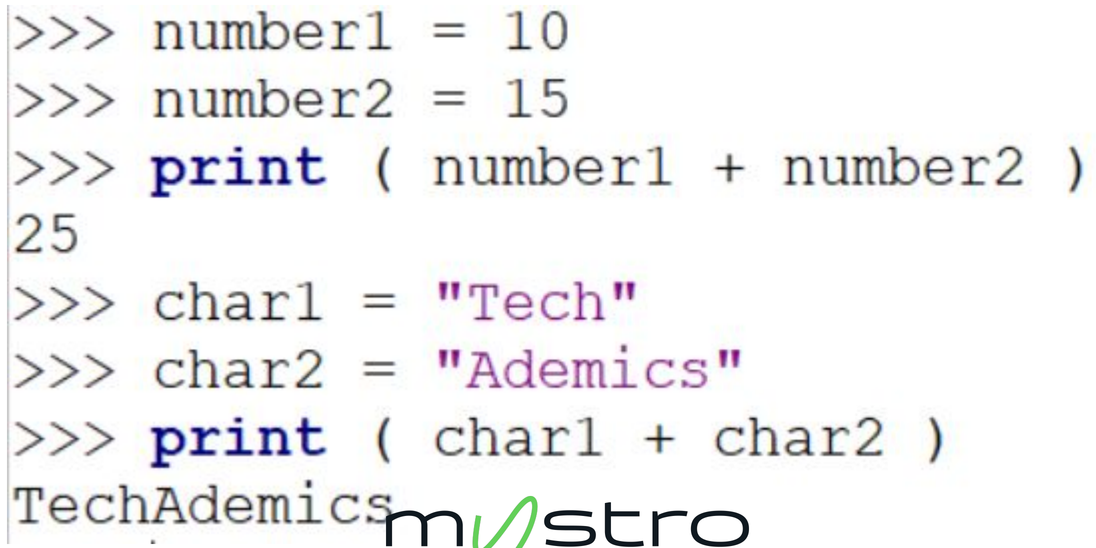
- Python allows adding different values in a single line with “,” operators.



```
>>> a , b , c = 1 , 1.11 , " Techademics "
>>> print ( a , b , c )
(1, 1.1100000000000001, ' Techademics ')
```

STRING CONCATENATION

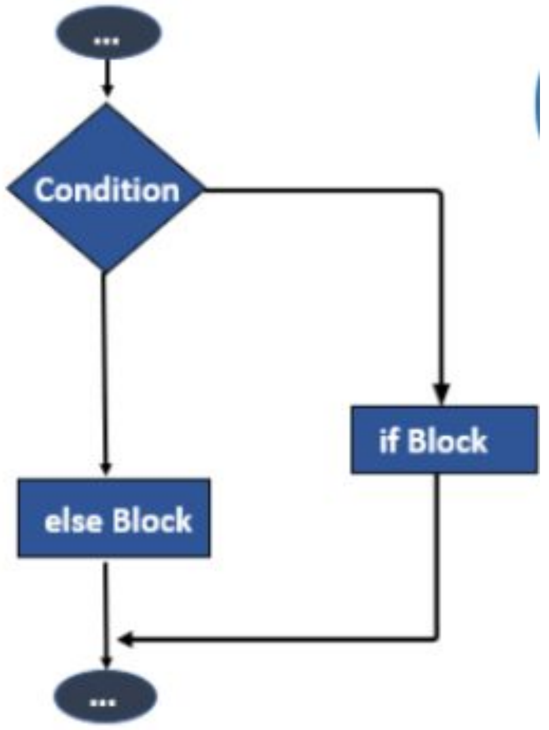
- The Python plus operator + provides a convenient way to add a value if it is a number and concatenate if it is a string. If a variable is already created it assigns the new value back to the same variable.



```
>>> number1 = 10
>>> number2 = 15
>>> print ( number1 + number2 )
25
>>> char1 = "Tech"
>>> char2 = "Ademics"
>>> print ( char1 + char2 )
TechAdemics
```

IF STATEMENTS IN PYTHON

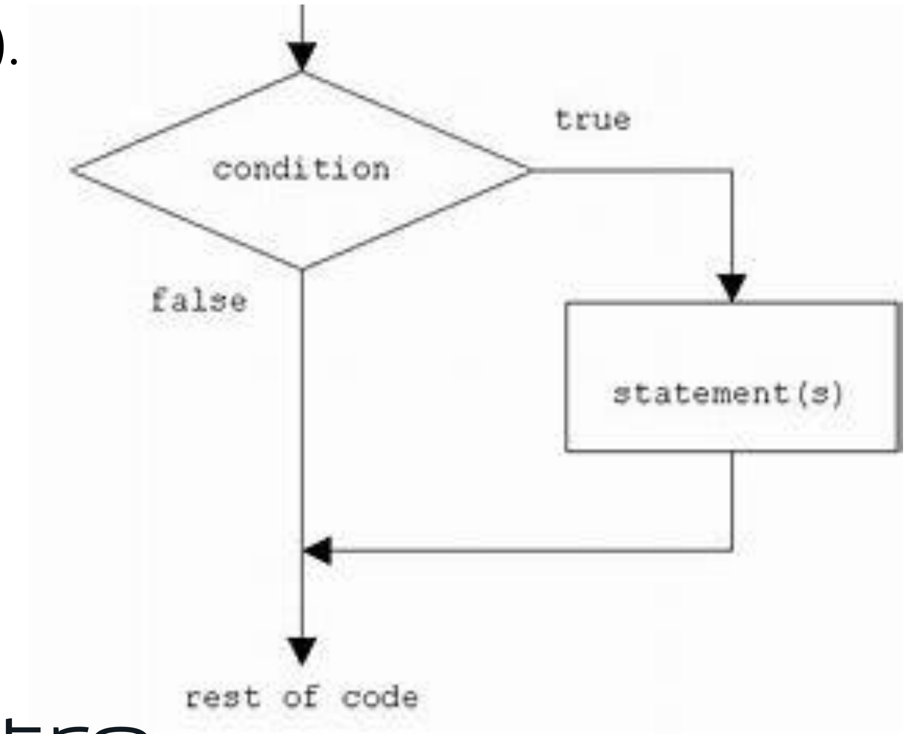
- . A program sometimes may have to make choices. These choices can execute different code depending on certain condition.



IF STATEMENTS IN PYTHON

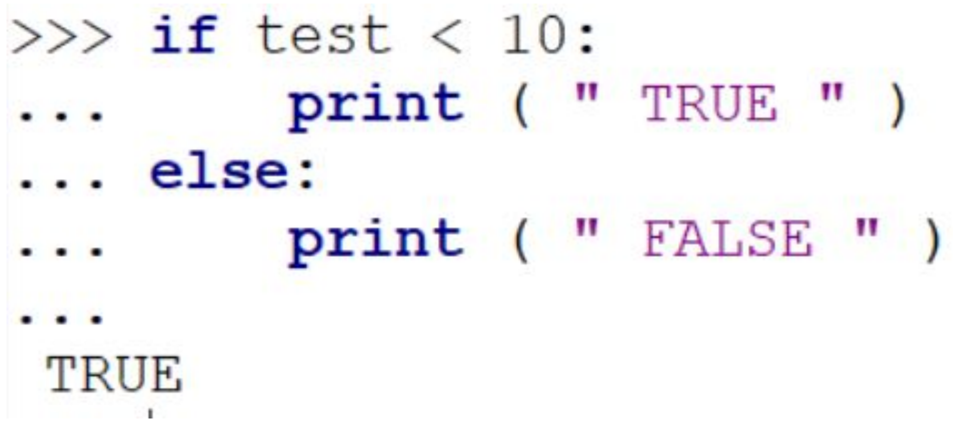
- The if statement may be combined with certain operator such as equality (==), greater than (>=), smaller than (<=) and not equal (!=).

.



IF STATEMENTS IN PYTHON

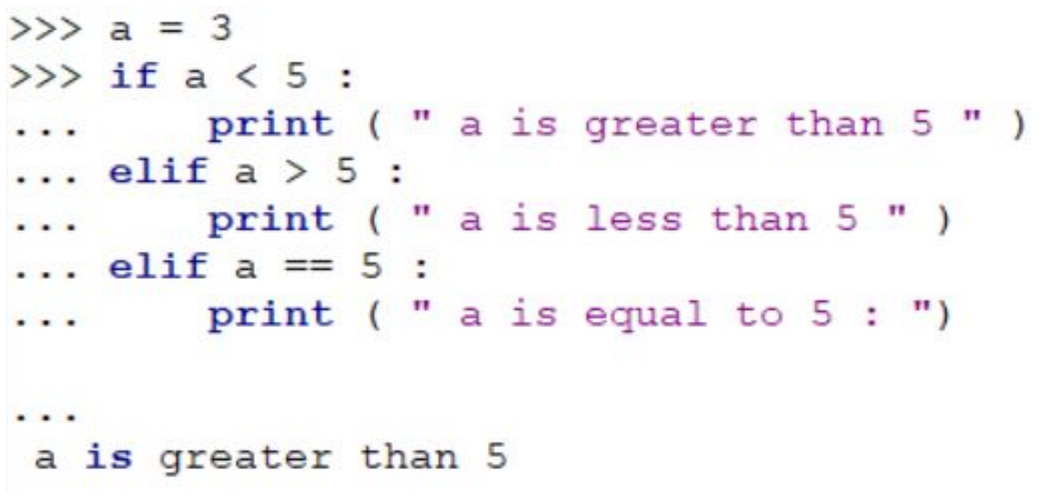
- In Python the if statement is used for conditional execution or branching. An if statement is one of the control structures. (A control structure controls the flow of the program.).



```
>>> if test < 10:
...     print ( " TRUE " )
... else:
...     print ( " FALSE " )
...
 TRUE
```

ELIF STATEMENTS IN PYTHON

- If you want to evaluate several cases, you can use the elif clause. elif is short for else if. .



```
>>> a = 3
>>> if a < 5 :
...       print ( " a is greater than 5 " )
... elif a > 5 :
...       print ( " a is less than 5 " )
... elif a == 5 :
...       print ( " a is equal to 5 : ")

...
 a is greater than 5
```

GETTING INFORMATION FROM THE USER

Gets the input from the user

$\downarrow$

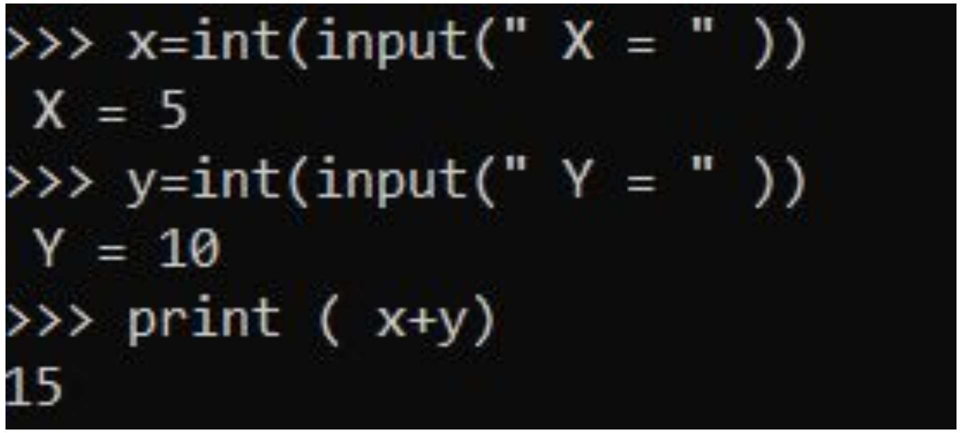
```
X = input(“Please enter your name”)
```

$\uparrow$

Stores what the user entered in the variable “X”

GETTING INFORMATION FROM THE USER

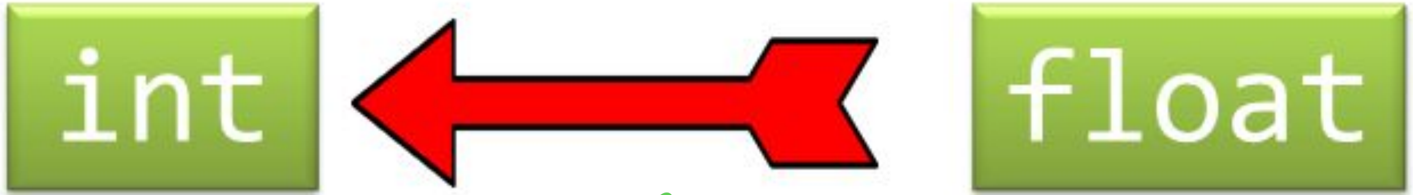
- In Python the input function return string values only.
- To take integer values need to do Type Casting.



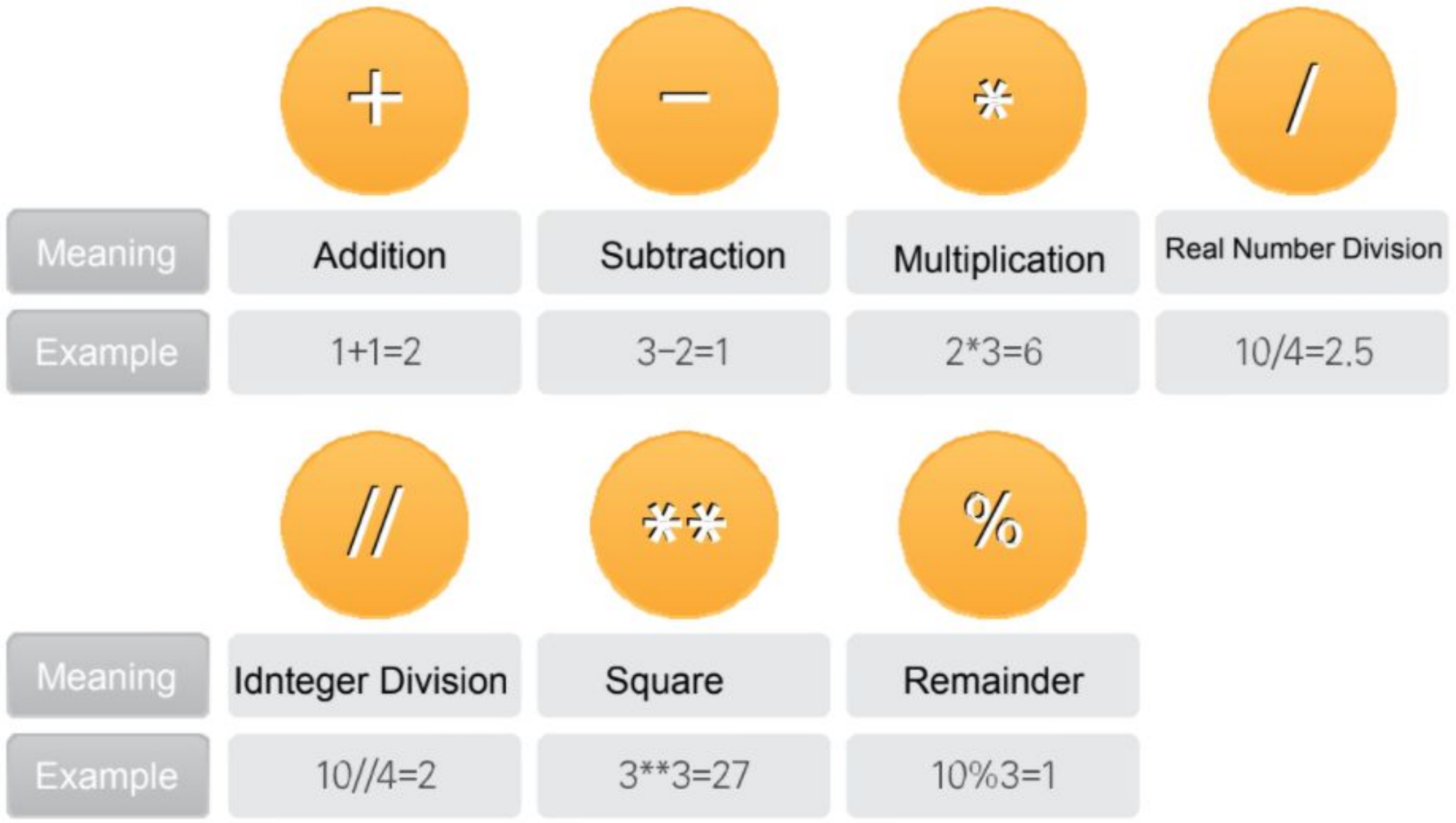
```
>>> x=int(input(" X = " ))
 X = 5
>>> y=int(input(" Y = " ))
 Y = 10
>>> print ( x+y)
15
```

TYPE CASTING

- Conversion from one date type to anc
- Syntax: DataType(Variable/Value)
- Example: int(2.8)



ARITHMETIC OPERATORS



	+	−	*	/
Meaning	Addition	Subtraction	Multiplication	Real Number Division
Example	1+1=2	3−2=1	2*3=6	10/4=2.5

	//	**	%
Meaning	Idnteger Division	Square	Remainder
Example	10//4=2	3**3=27	10%3=1

CHALLENGE

Build a simple Calculator

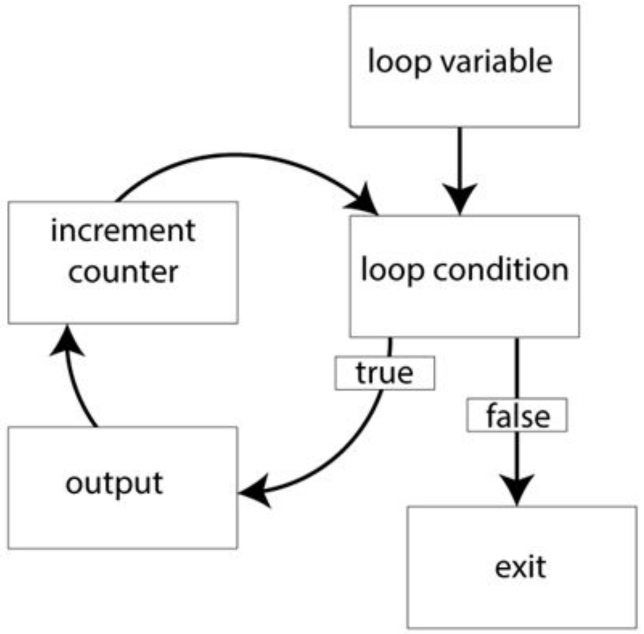
- Take 2 inputs from user
- Do all the Arithmetic operations we discussed on them
- In Pow (User Choose power)

LOOPs IN PYTHON

There are four main components :

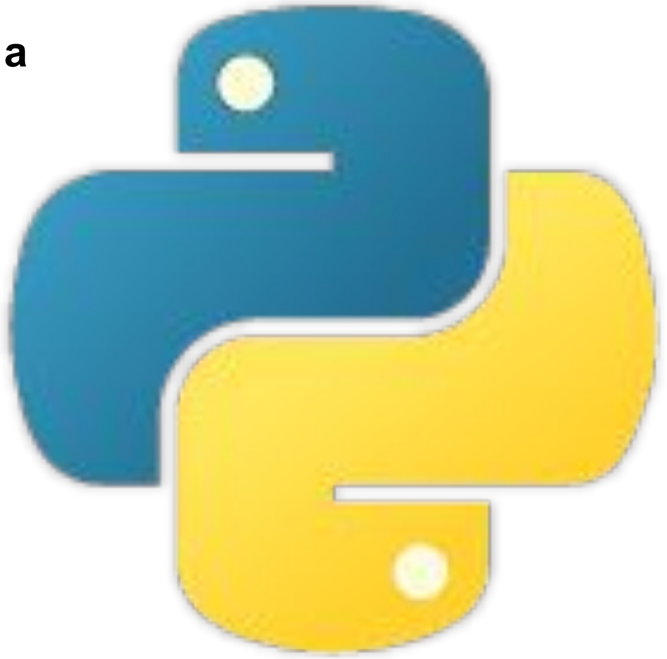
- ❑ Initial value
- ❑ Conditional expression
- ❑ Repetition structure
- ❑ Increment and decrement operator

•



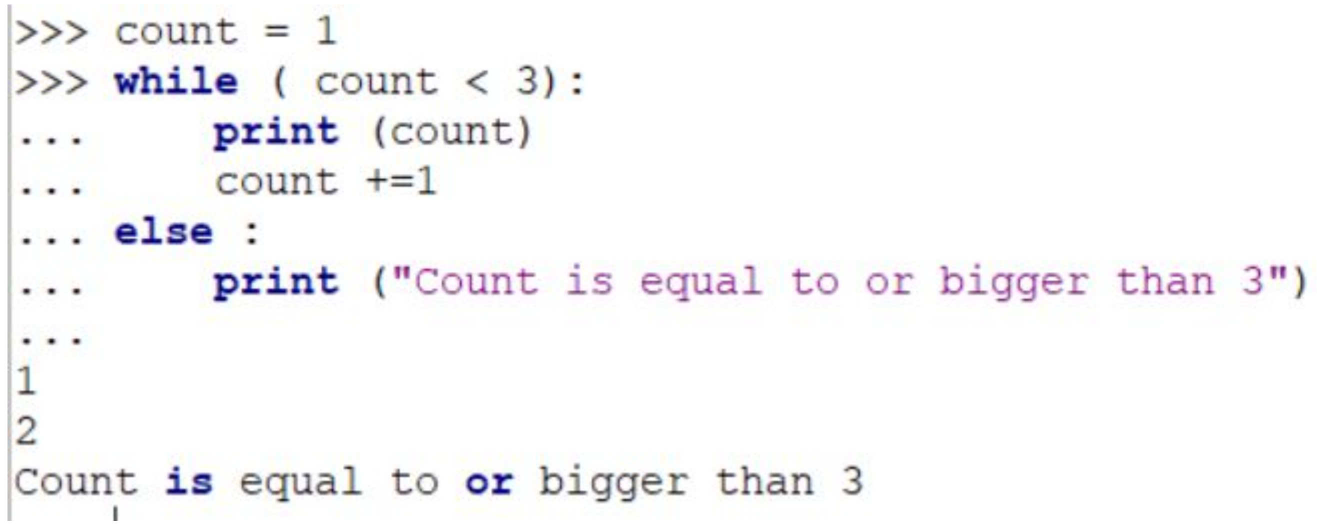
WHILE LOOP IN PYTHON

- In python, a while loop is used to execute a block of statements repeatedly until a given condition is satisfied.
- Python uses INDENTATION as its method of grouping statements.

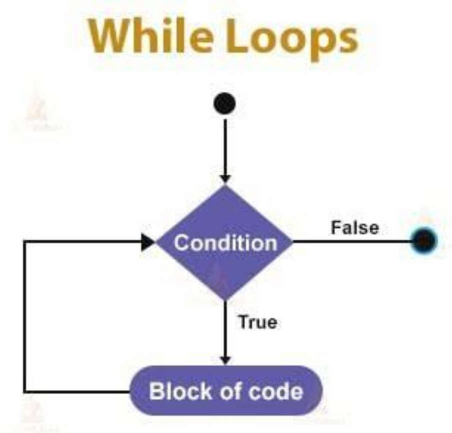


WHILE LOOP IN PYTHON

- The ELSE clause is only executed when your while condition becomes FALSE. If you break out of the loop, or if an exception is raised, it won't be executed.

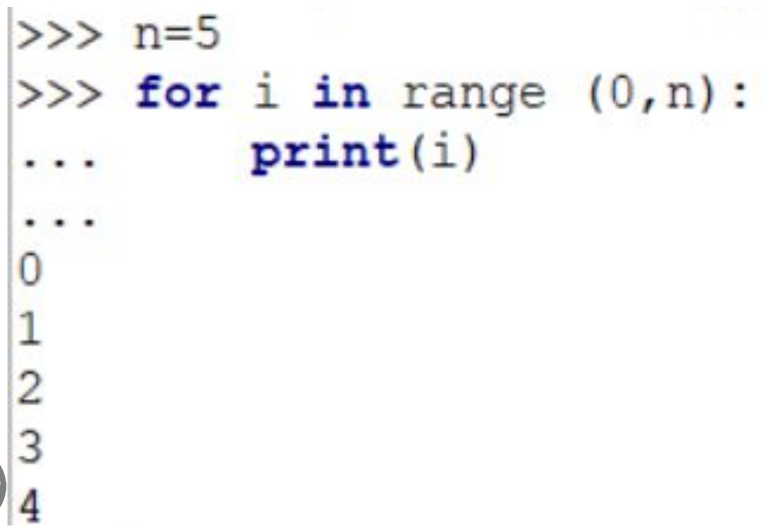


```
>>> count = 1
>>> while ( count < 3):
...       print (count)
...       count +=1
... else :
...       print ("Count is equal to or bigger than 3")
...
1
2
Count is equal to or bigger than 3
```

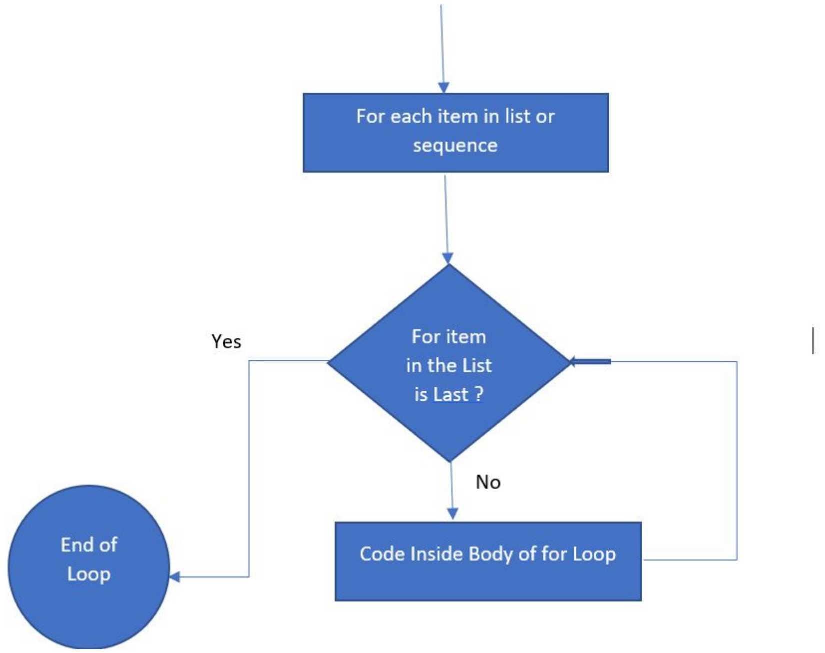


FOR LOOP IN PYTHON

- For loops are used for sequential traversal. For example: traversing a list or string or array etc.

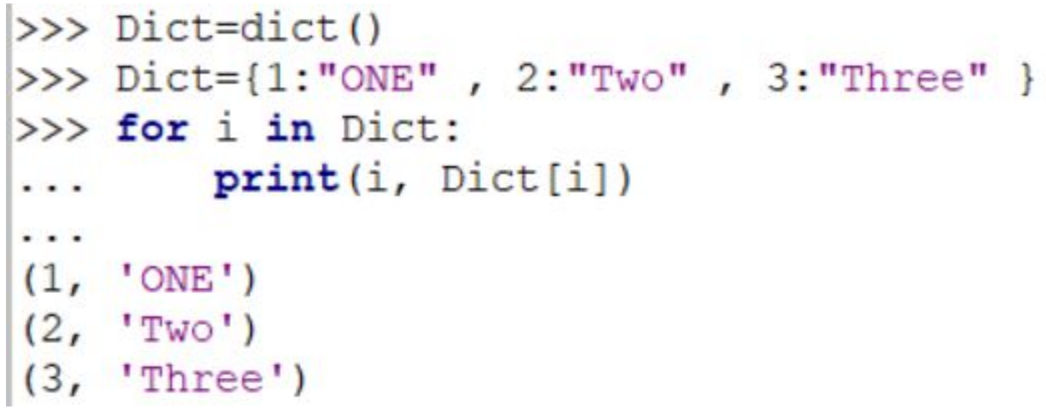


```
>>> n=5
>>> for i in range (0,n):
...         print(i)
...
0
1
2
3
4
```


For each item in list or sequence
For item in the List is Last ?
Yes
No
Code Inside Body of for Loop
End of Loop

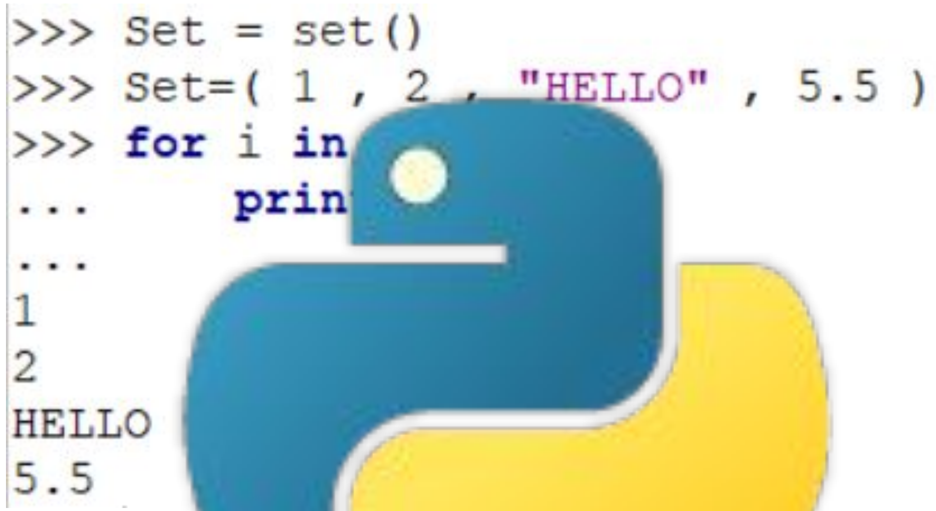
FOR LOOP IN PYTHON

- ❑ Dict Iteration



```
>>> Dict=dict()
>>> Dict={1:"ONE" , 2:"Two" , 3:"Three" }
>>> for i in Dict:
...     print(i, Dict[i])
...
(1, 'ONE')
(2, 'Two')
(3, 'Three')
```

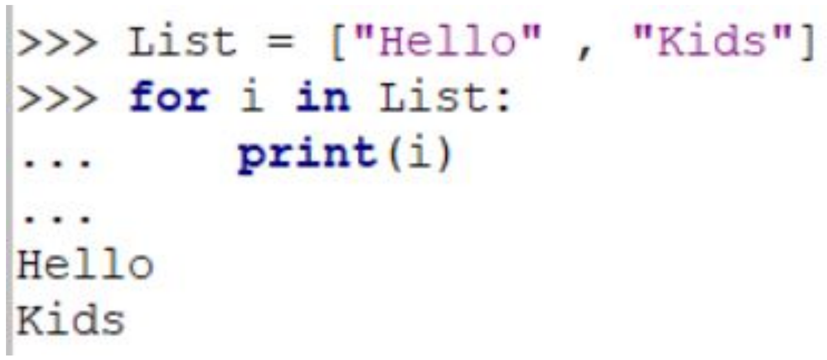
- ❑ Set Iteration



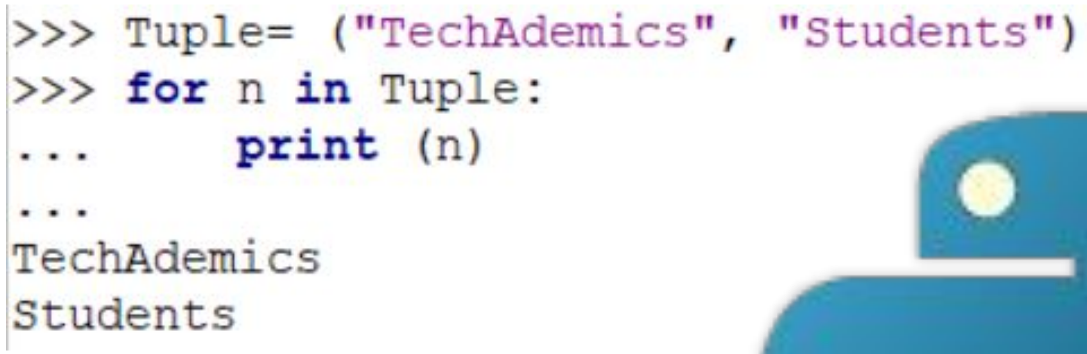
```
>>> Set = set()
>>> Set=( 1 , 2 , "HELLO" , 5.5 )
>>> for i in
...     prin
...
1
2
HELLO
5.5
```

FOR LOOP IN PYTHON

- ❑ List Iteration
- ❑ Tuple Iteration



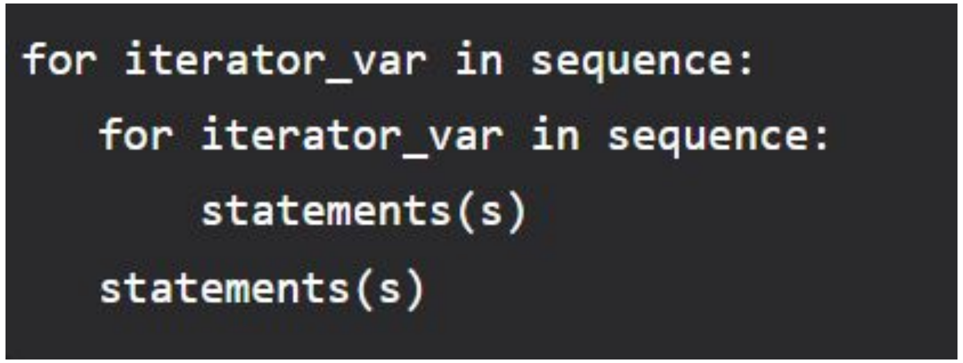
```
>>> List = ["Hello" , "Kids"]
>>> for i in List:
...     print(i)
...
Hello
Kids
```



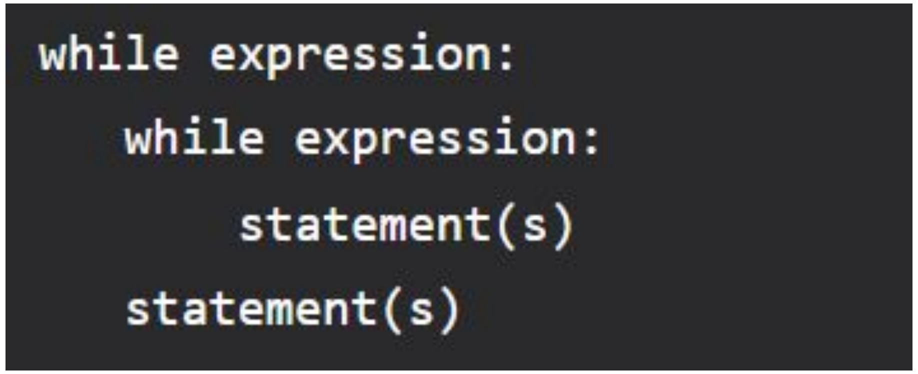
```
>>> Tuple= ("TechAdemics", "Students")
>>> for n in Tuple:
...     print (n)
...
TechAdemics
Students
```

NESTED LOOPS

- Python programming language allows to use one loop inside another loop



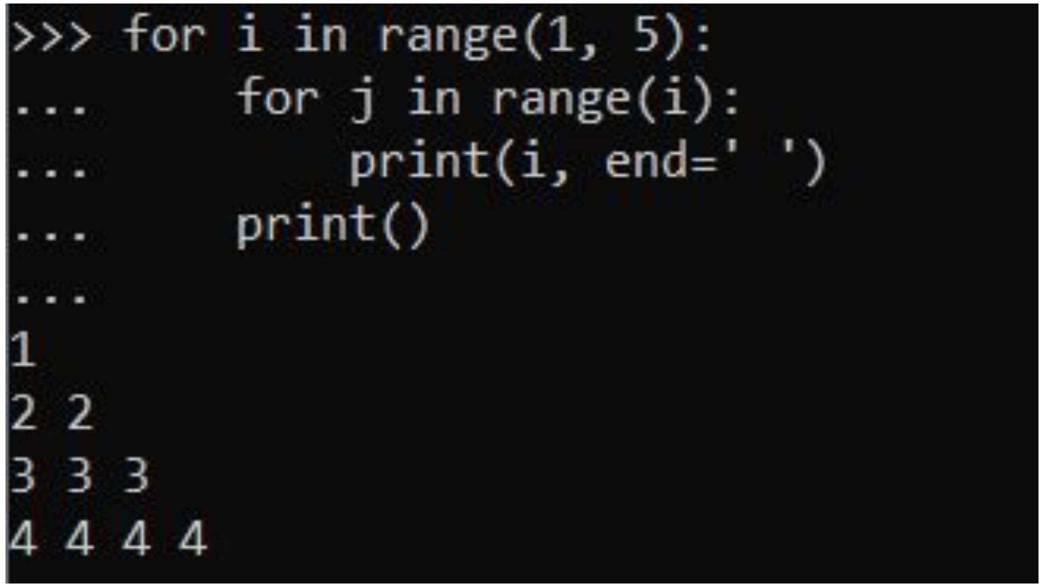
```
for iterator_var in sequence:
   for iterator_var in sequence:
       statements(s)
   statements(s)
```



```
while expression:
   while expression:
       statement(s)
   statement(s)
```

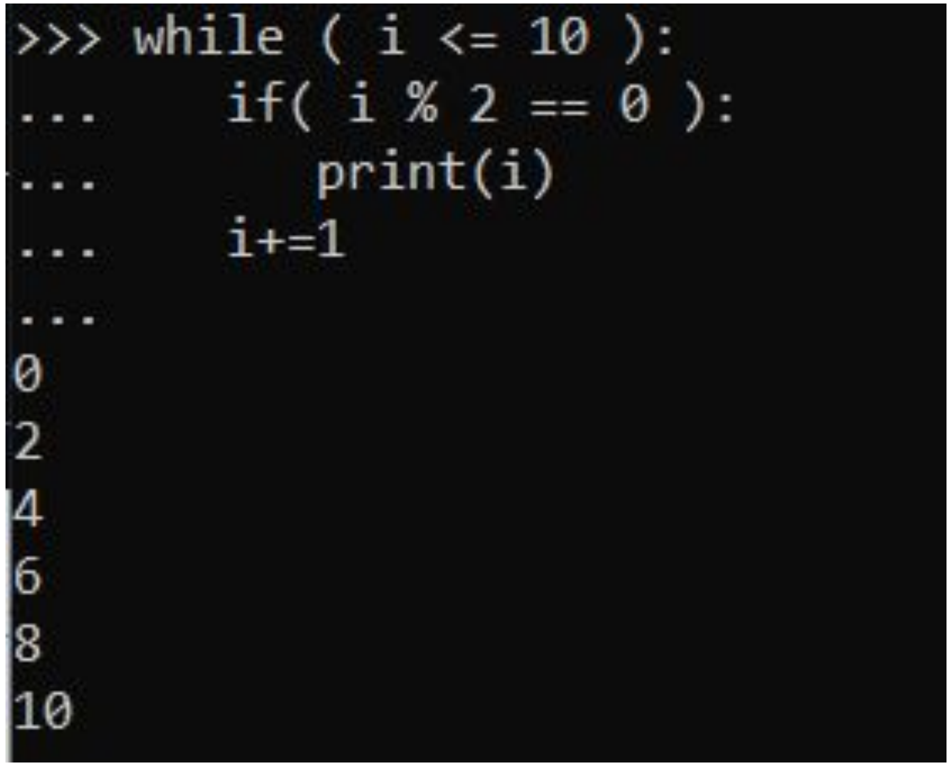
FOR LOOP IN PYTHON

- For Nesting



```
>>> for i in range(1, 5):
...     for j in range(i):
...         print(i, end=' ')
...     print()
...
1
2 2
3 3 3
4 4 4 4
```

- While Nesting



```
>>> while ( i <= 10 ):
...     if( i % 2 == 0 ):
...         print(i)
...     i+=1
...
0
2
4
6
8
10
```

Try this in your CMD

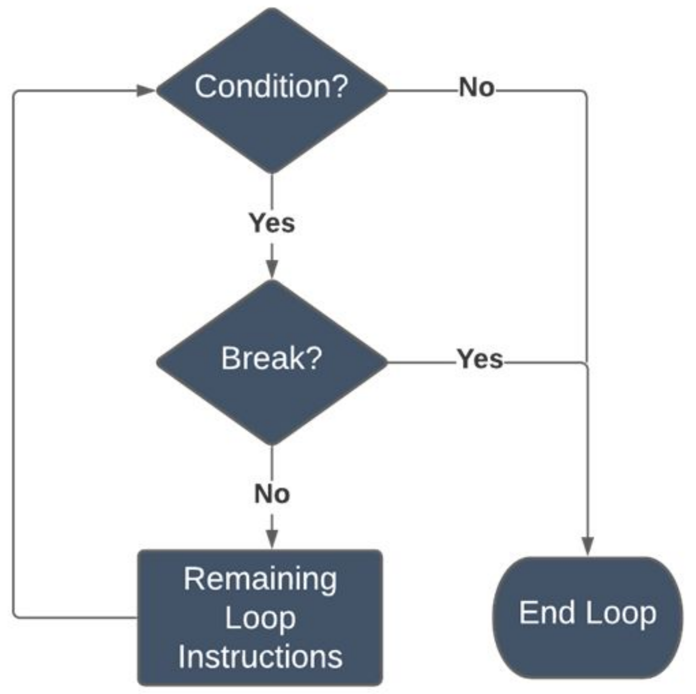
AI-ROS

mystro

OLA EL-SHIEKH

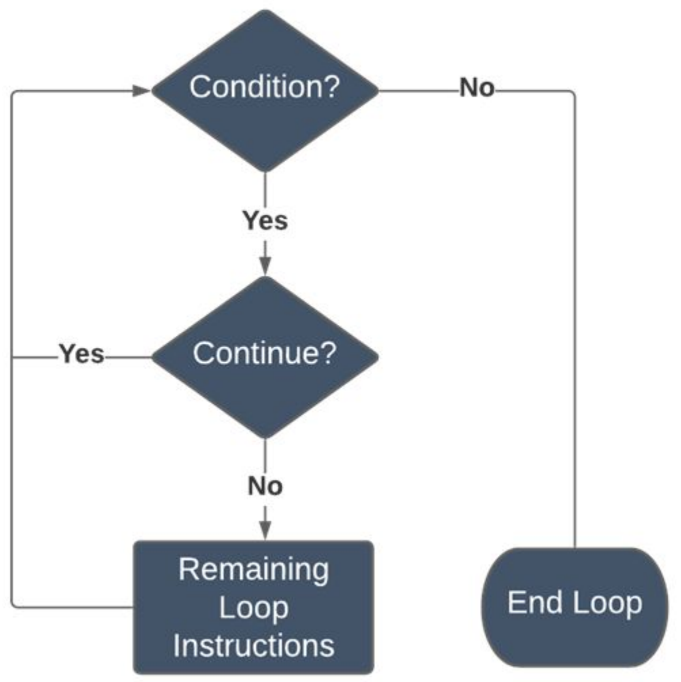
BREAK KEYWORD

- The break statement is used to terminate the loop or statement in which it is present
- After that, the control will pass to the statements that are present after the break statement, if available.



CONTINUE KEYWORD

- Continue is also a loop control statement just like the break statement.
- Continue statement is opposite to that of break statement, instead of terminating the loop, it forces to execute the next iteration of the loop



Break

7 : 15 Sharp

THANK YOU

ANY QUESTIONS ?

OLA EL-SHIEKH

AI-ROS

ola.elshiekh71@gmail.com

LINKEDIN