TECHNO INDIA
SILIGURI INSTITUTE
OF TECHNOLOGY
A Satyam Roychowdhury Initiative

# Introduction to Python Programming

MCA 1st Semester, MCAN101

## Why to study Python?

- Development productivity
- Software Quality
- Program portability
- Support libraries
- Component integration
- Enjoyment

## What can you do with python?

- Graphical User Interface
- Internet Scripting
- Database Programming
- Numeric and Scientific programming
- Components integration

# Python’s Technical Strength

- It’s Free
- It’s Object Oriented
- It’s easy to learn
- It’s mixable

## Python’s Data Types

- Python takes data in the form of objects
- Objects, either built-in object provided by Python or created using Python
- Objects are most fundamental notion of Python programming

# Python’s Core Data Types

| Object Type | Examples |
|---|---|
| Numbers | 1234, 3.145,3+6j |
| Strings | ‘spam’, “guido” |
| Lists | [1,2,3],[1,[2,3]] |
| Dictionaries | {a:1,b:3,4:c} |
| Tuples | (1,2,”ram”) |
| Files | Flptr=open(“test.txt”,’r’) |
| Sets | Set(‘abc’),{‘a’,’b’,’c’} |

# Python’s Statement

Statements are the things written to tell Python what your program should do

| Statement | Role | Example |
|---|---|---|
| Assignment | Creating references | a,b=‘good’,’bad’ |
| Calls and other expression | Running functions | log.write(“spam,ham”) |
| Print calls | Printing objects | print(“hi’) |
| If/elif/else | Selecting actions | if a>0: print(‘hi’) |
| For/else | Sequence iteration | for i in range(5):print(i) |
| While/else | General loop | while x>y: print(‘hello’) |
| def | Function and method | def fun(a,b): pass |
| class | Building objects | class student: pass |
| Import | Module access | import numpy |
| from | Attribute access | from module import fun |

# Special in Python

Python removed parenthesis

Python removed braces

Python added indentation with colon (:)

End of line statement

Semicolon can be added for multiple statement in one line

End of indentation is end of block

# Invention of Python Language

- Developed by Guido van Rossum in 1991 at the National Research Institute for Mathematics and Computer Science.
- Python creator Guido van Rossum named it after the BBC comedy series Monty Python's Flying Circus

## Structure of Python program

```
def fun1(…):

……….

def fun2():

………

def fun3(…..):

………..

statement1

statement 2

statement3
```

- Interpreter execute statements from top to bottom
- Function definitions are digested for future use
- Actual computation starts from Statement1

## Assignment Statement

- Assign a value to a name
  - ➢ i=5
  - ➢ j=2*5
  - ➢ k=i*j
- Left hand side is a name
- Right hand side is an expression

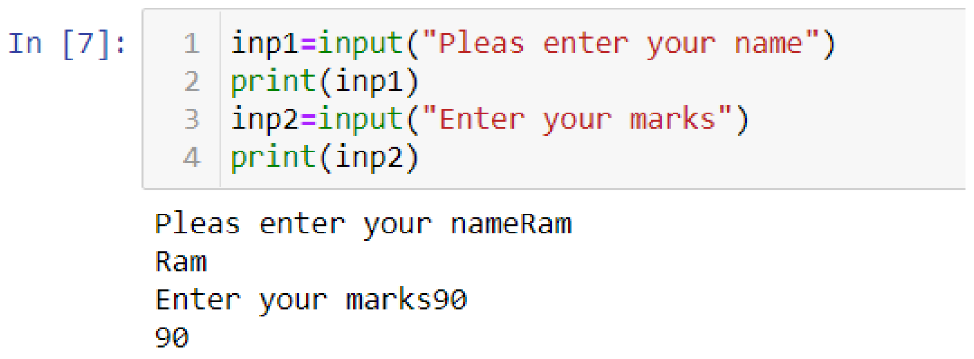
➢Operations in expression depends on type of value

## Comments and Tokens

- Everything after "#" on a line is ignored.
- Multiline comment start and end with’’’
- Allowed characters: az-AZ-09 underscore, and must begin with a letter or underscore.
- Names and identifiers are case sensitive.
- Identifiers can be of unlimited length.
- Special names, customizing, etc. Usually begin and end in double underscores.
- Special name classes Single and double underscores.
- Naming convention-not rigid

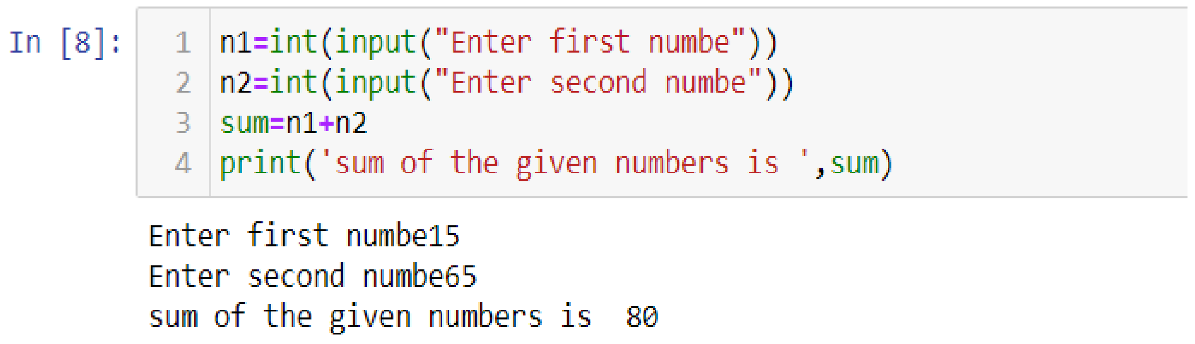
# Input and Output

- The print() function display the output in the console.
- The input() function accepts and returns the user’s input as a string and store it in the variable  which is assigned with the assignment operator.
- Use appropriate conversion function to work with numeric values.



```
In [7]:
inp1=input("Pleas enter your name")
print(inp1)
inp2=input("Enter your marks")
print(inp2)
```

```
Pleas enter your nameRam
Ram
Enter your marks90
90
```

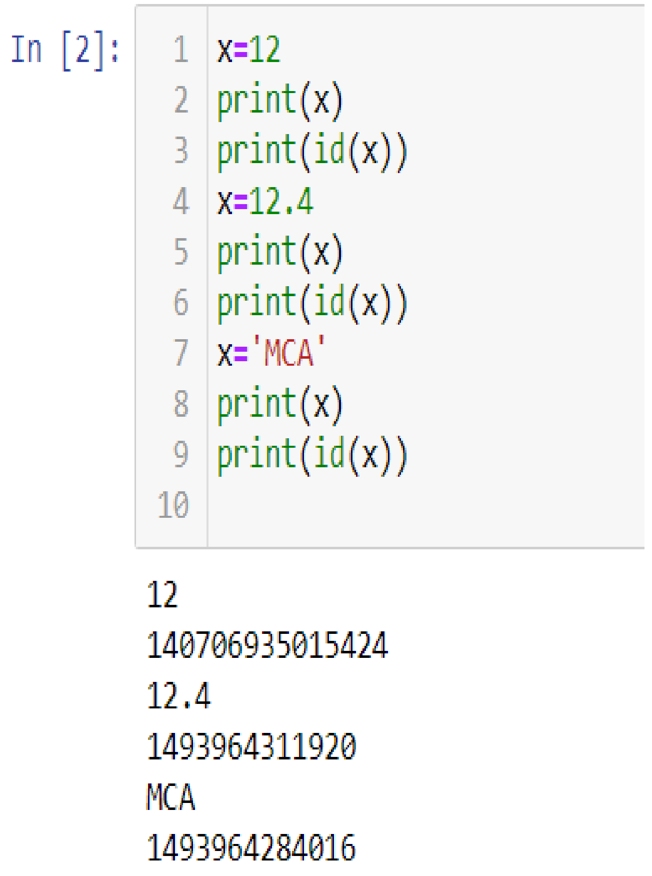


```
In [8]:
n1=int(input("Enter first numbe"))
n2=int(input("Enter second numbe"))
sum=n1+n2
print('sum of the given numbers is ',sum)
```

```
Enter first numbe15
Enter second numbe65
sum of the given numbers is  80
```

# Mutable and immutable types

- Variables whose values can be changed after they are created and assigned are called mutable.
- Variables whose values can not be changed are called immutable.
- When an attempt is made to update the value of an immutable variable the old variable is destroyed and a new variable is by the same name.
- Mutable objects: list, dictionary, set etc.
- Immutable objects: int, float, complex, bool, string, tuple etc.

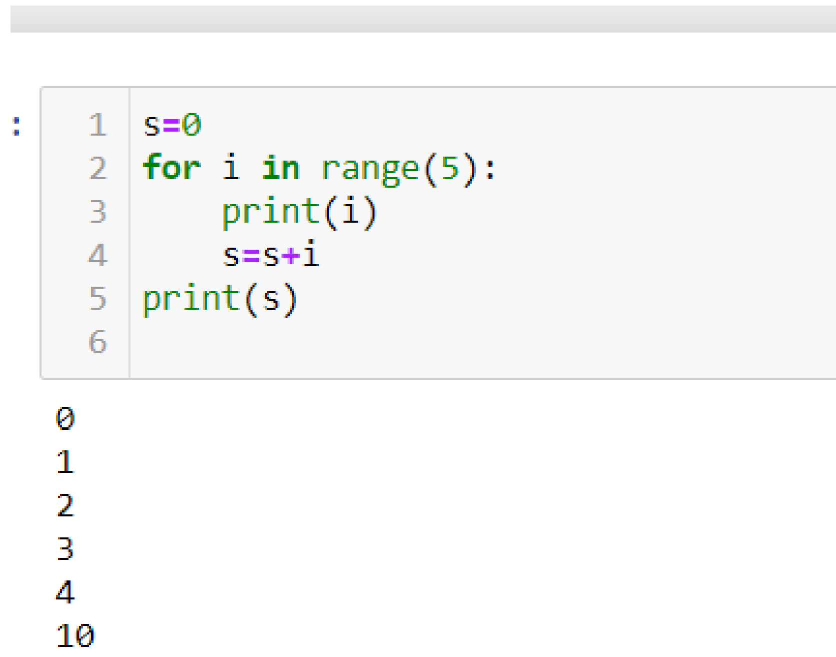


```
In [2]:
x=12
print(x)
print(id(x))
x=12.4
print(x)
print(id(x))
x='MCA'
print(x)
print(id(x))
```

```
12
140706935015424
12.4
1493964311920
MCA
1493964284016
```

# Blocks and indentation

- Python represents block structure and nested block structure with indentation, not with begin and end brackets.
- The empty block Use--the pass no-op statement.
- Reduces work. Only need to get the indentation correct, not both indentation and brackets.
- Reduces clutter. Eliminates all the curly brackets.
- If it looks correct, it is correct. Indentation cannot fool the reader.

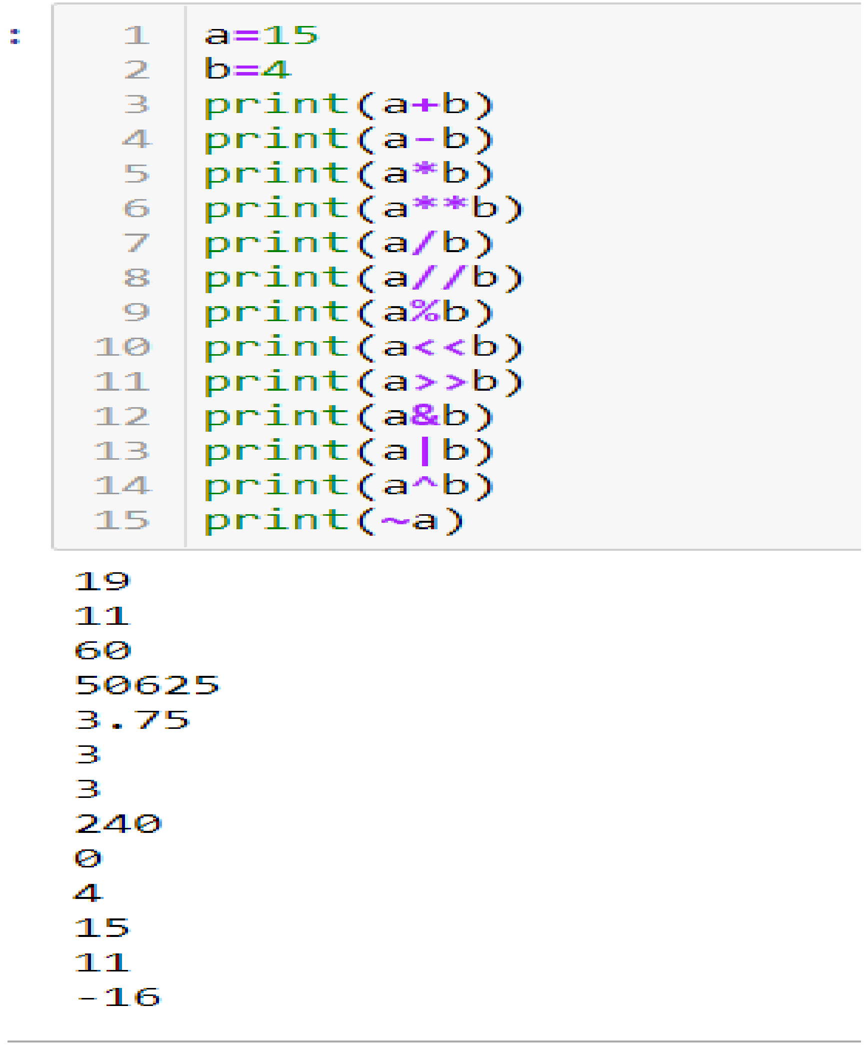


```python
s=0
for i in range(5):
    print(i)
    s=s+i
print(s)
```

```
0
1
2
3
4
10
```

# Operators in Python

- Python defines the following operators
- + ,-,*, **, /, //, %, <<, >>, &, | ,^ ,~, < ,>, <=, >=, ==, !=, <>
- Logical Operators
- and, or, is, not, in



```
a=15
b=4
print(a+b)
print(a-b)
print(a*b)
print(a**b)
print(a/b)
print(a//b)
print(a%b)
print(a<<b)
print(a>>b)
print(a&b)
print(a|b)
print(a^b)
print(~a)
```

```
19
11
60
50625
3.75
3
3
240
0
4
15
11
-16
```

# Thank You