

Unit 1: Introduction and syntax of python programming

Python is developed by **Guido van Rossum**. Guido van Rossum started implementing Python in 1989.

Features of python –

Python is Interactive –

You can actually sit at a Python prompt and interact with the interpreter directly to write your programs.

Python is Object-Oriented –

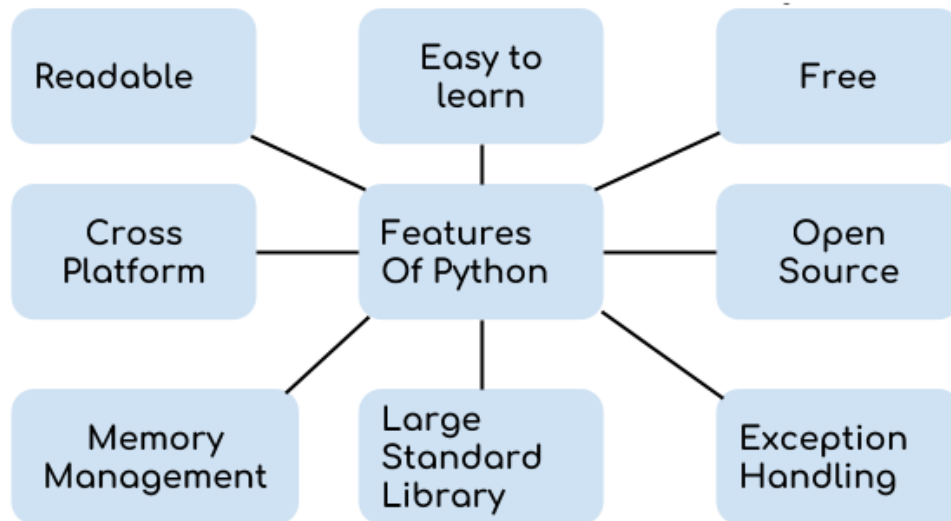
Python supports object oriented language and concepts of classes and objects come into existence.

Python is Interpreted

Python is an interpreted language i.e. interpreter executes the code line by line at a time. This makes debugging easy and thus suitable for beginners.

Python is Platform Independent

Python can run equally on different platforms such as Windows, Linux, Unix and Macintosh etc. So, we can say that Python is a portable language.



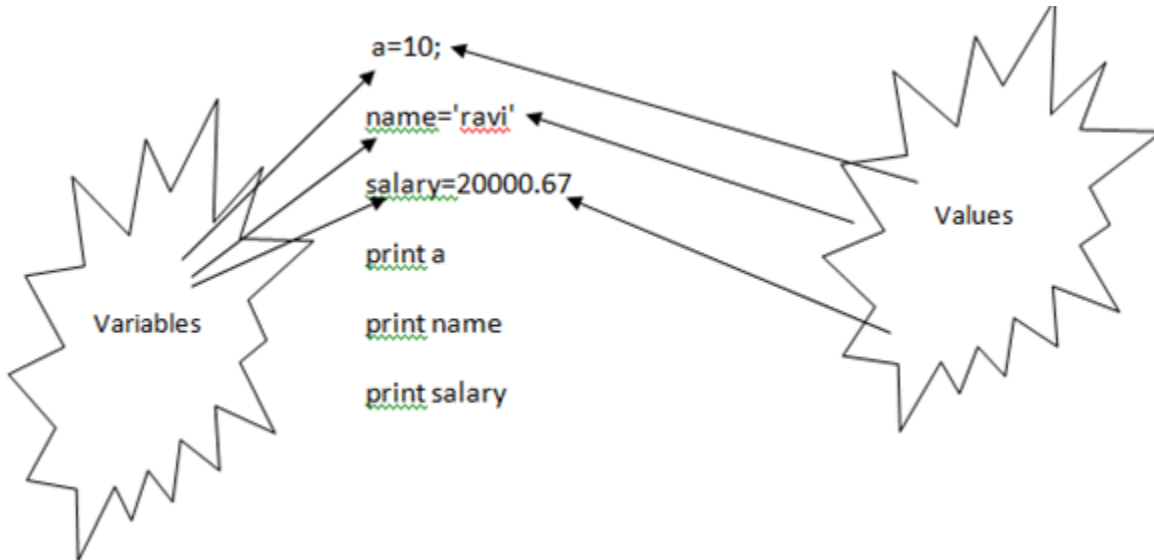
Python building blocks

Python Identifiers

Variable name is known as identifier.

The rules to name an identifier are given below.

- The first character of the variable must be an alphabet or underscore (_).
- All the characters except the first character may be an alphabet of lower-case(a-z), upper-case (A-Z), underscore or digit (0-9).
- Identifier name must not contain any white-space, or special character (!, @, #, %, ^, &, *).
- Identifier name must not be similar to any keyword defined in the language.
- Identifier names are case sensitive for example my name, and MyName is not the same.
- Examples of valid identifiers : a123, _n, n_9, etc.
- Examples of invalid identifiers: 1a, n%4, n 9, etc.



Assigning single value to multiple variables

Eg: `x=y=z=50`

Assigning multiple values to multiple variables:

Course Outcome (CO): Display message on screen using Python Script on IDE.

Eg: a,b,c=5,10,15

Reserved Words

The following list shows the Python keywords. These are reserved words and cannot use them as constant or variable or any other identifier names. All the Python keywords contain lowercase letters only.

| | | |
|----------|---------|--------|
| and | exec | not |
| assert | finally | or |
| break | for | pass |
| class | from | print |
| continue | global | raise |
| def | if | return |
| del | import | try |
| elif | in | while |
| else | is | with |
| except | lambda | yield |

Indentation

Python provides no braces to indicate blocks of code for class and function definitions or flow control. Blocks of code are denoted by line indentation, which is compulsory.

The number of spaces in the indentation is variable, but all statements within the block must be indented the same amount. For example –

```
if True:
    print "True"
else:
    print "False"
```

Thus, in Python all the continuous lines indented with same number of spaces would form a block.

Course Outcome (CO): Display message on screen using Python Script on IDE.

Variable Types

Variables are used to store data, they take memory space based on the type of value we assigning to them. Creating variables in Python is simple, you just have write the variable name on the left side of = and the value on the right side.

Python Variable Example

```
num = 100
str = "BeginnersBook"
print(num)
print(str)
```

Multiple Assignment Examples

We can assign multiple variables in a single statement like this in Python.

```
x = y = z = 99
print(x)
print(y)
print(z)
```

Another example of multiple assignments

```
a, b, c = 5, 6, 7
print(a)
print(b)
print(c)
```

Plus and concatenation operation on the variables

```
x = 10
y = 20
print(x + y)

p = "Hello"
q = "World"
print(p + " " + q)
```

Course Outcome (CO): Display message on screen using Python Script on IDE.

output:

30

Hello World

Comments

Use the hash (#) symbol to start writing a comment.

```
1. #This is a comment
2. #print out Hello
3. print('Hello')
```

Multi-line comments

use triple quotes, either ''' or """".

eg:

```
1. """This is also a
2. perfect example of
3. multi-line comments"""
```

Data Types

A data type defines the type of data, for example 123 is an integer data while “hello” is a String type of data. The data types in Python are divided in two categories:

1. Immutable data types – Values cannot be changed.
2. Mutable data types – Values can be changed

Immutable data types in Python are:

1. Numbers
2. String
3. Tuple

Mutable data types in Python are:

1. List
2. Dictionaries
3. Sets

Python Environment Setup-Installation and Working Of IDE

Course Outcome (CO): Display message on screen using Python Script on IDE.

Install Python on any operating system such as Windows, Mac OS X, Linux/Unix and others.

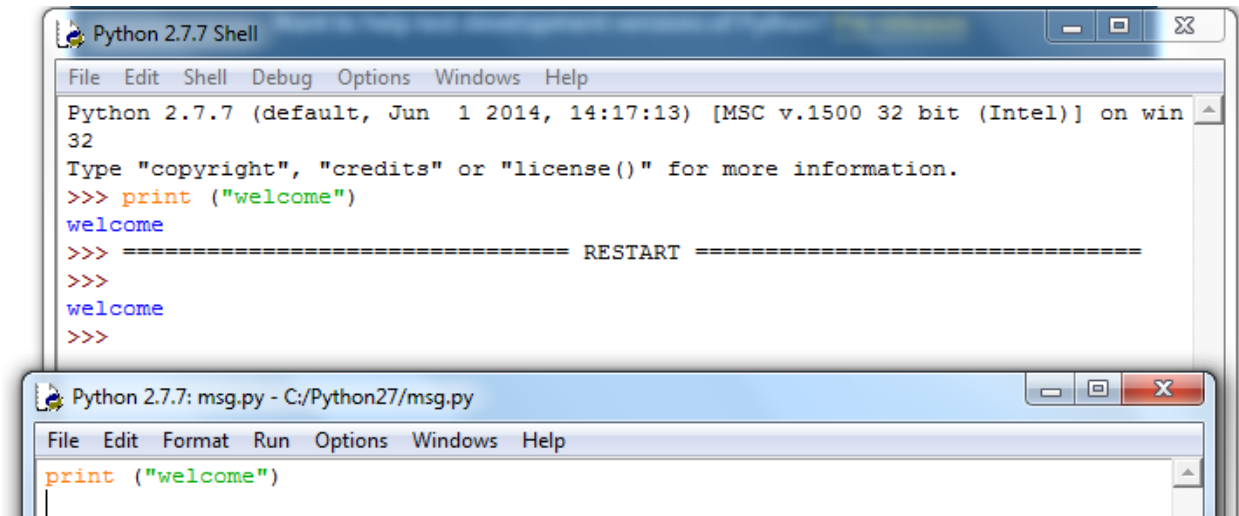
To install the Python on your operating system, go to this link: <https://www.python.org/downloads/>. You will see a screen like this.



1. On Windows 7 and earlier, IDLE is easy to start—it's always present after a Python install, and has an entry in the Start button menu for Python in Windows 7 and earlier.
2. Select it by right-clicking on a Python program icon, and launch it by clicking on the icon for the files idle.pyw or idle.py located in the idlelib subdirectory of Python's Lib directory. In this mode, IDLE is a clickable Python script that lives in C:\Python3.6\..

Running Simple Python Scripts To Display 'Welcome' Message

Course Outcome (CO): Display message on screen using Python Script on IDE.



Python data types : numbers ,string, tuples, lists, dictionary.

Python Data Types

- 1 Python Data Type – Numeric
- 2 Python Data Type – String
- 3 Python Data Type – List
- 4 Python Data Type – Tuple
- 5 Dictionary

Declaration and use of data types

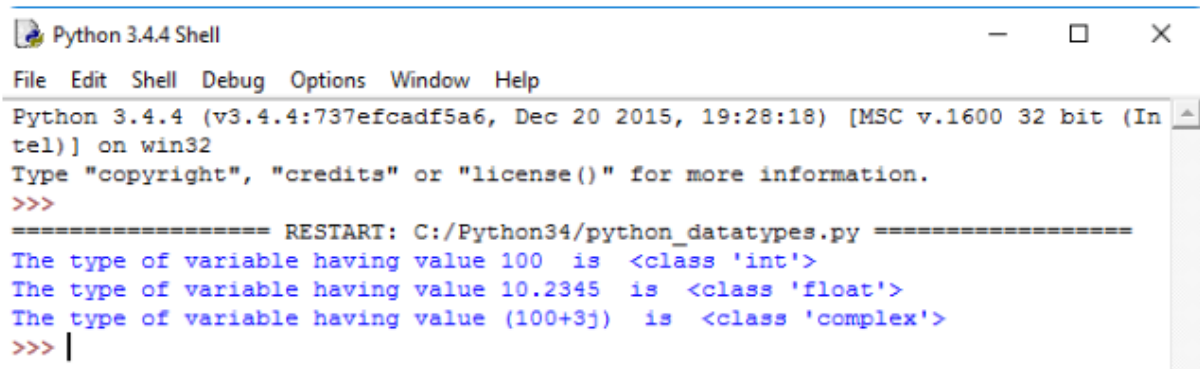
Python Data Type – Numeric

Python numeric data type is used to hold numeric values like;

- int – holds signed integers of non-limited length.
- long- holds long integers(exists in Python 2.x, deprecated in Python 3.x).
- float- holds floating precision numbers and it's accurate upto 15 decimal places.
- complex- holds complex numbers.

```
#create a variable with integer value.  
a=100  
print("The type of variable having value", a, " is ", type(a))  
  
#create a variable with float value.  
b=10.2345  
print("The type of variable having value", b, " is ", type(b))  
  
#create a variable with complex value.  
c=100+3j  
print("The type of variable having value", c, " is ", type(c))
```

If you run the above code you will see output like the below image.



```
Python 3.4.4 Shell  
File Edit Shell Debug Options Window Help  
Python 3.4.4 (v3.4.4:737efcadf5a6, Dec 20 2015, 19:28:18) [MSC v.1600 32 bit (Intel)] on win32  
Type "copyright", "credits" or "license()" for more information.  
>>>  
===== RESTART: C:/Python34/python_datatypes.py =====  
The type of variable having value 100 is <class 'int'>  
The type of variable having value 10.2345 is <class 'float'>  
The type of variable having value (100+3j) is <class 'complex'>  
>>> |
```

Python Data Type – String

The string is a sequence of characters. Python supports Unicode characters. Generally, strings are represented by either single or double quotes.

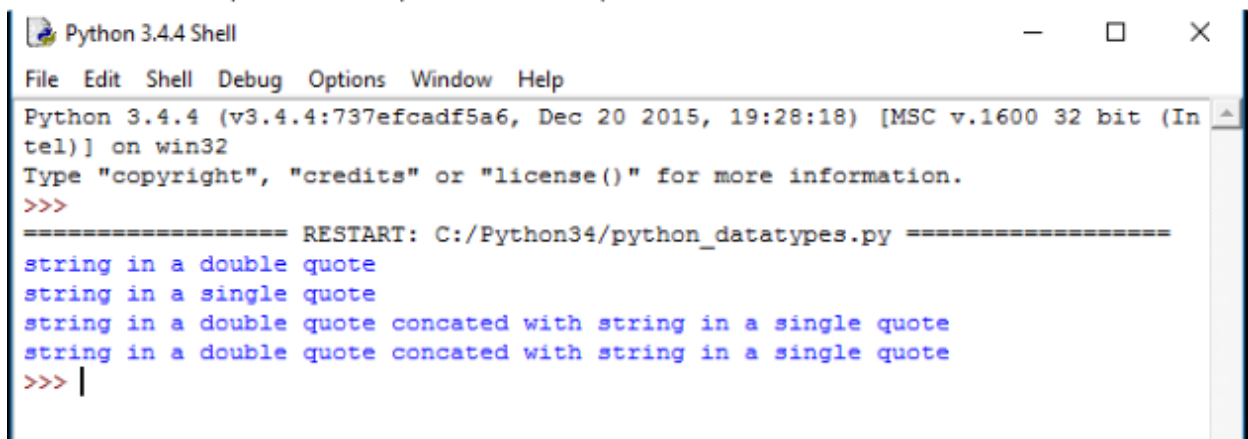
Course Outcome (CO): Display message on screen using Python Script on IDE.

```
a = "string in a double quote"
b= 'string in a single quote'
print(a)
print(b)

# using ',' to concatenate the two or several strings
print(a,"concatenated with",b)

#using '+' to concate the two or several strings
print(a+" concated with "+b)
```

The above code produce output like below picture-

A screenshot of a Python 3.4.4 Shell window. The window title is "Python 3.4.4 Shell". The menu bar includes "File", "Edit", "Shell", "Debug", "Options", "Window", and "Help". The status bar at the bottom shows "Python 3.4.4 (v3.4.4:737efcadf5a6, Dec 20 2015, 19:28:18) [MSC v.1600 32 bit (Intel)] on win32". The main text area shows the following output:

```
Type "copyright", "credits" or "license()" for more information.
>>>
===== RESTART: C:/Python34/python_datatypes.py =====
string in a double quote
string in a single quote
string in a double quote concated with string in a single quote
string in a double quote concated with string in a single quote
>>> |
```

Python Data Type – List

List is an ordered sequence of some data written using square brackets ([]) and commas (,).

Course Outcome (CO): Display message on screen using Python Script on IDE.

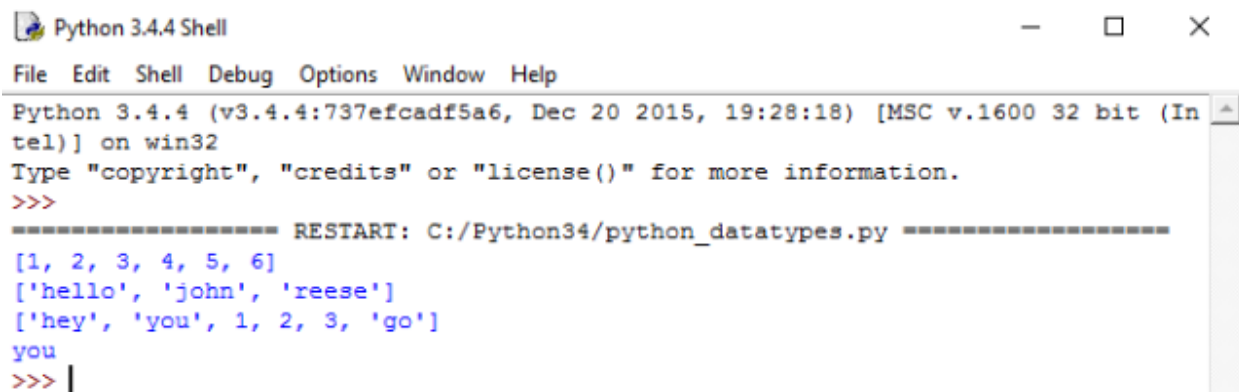
```
#list of having only integers
a= [1,2,3,4,5,6]
print(a)

#list of having only strings
b=["hello","john","reese"]
print(b)

#list of having both integers and strings
c= ["hey","you",1,2,3,"go"]
print(c)

#index are 0 based. this will print a single character
print(c[1]) #this will print "you" in list c
```

The above code will produce output like this-



```
Python 3.4.4 Shell
File Edit Shell Debug Options Window Help
Python 3.4.4 (v3.4.4:737efcadf5a6, Dec 20 2015, 19:28:18) [MSC v.1600 32 bit (Intel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>>
===== RESTART: C:/Python34/python_datatypes.py =====
[1, 2, 3, 4, 5, 6]
['hello', 'john', 'reese']
['hey', 'you', 1, 2, 3, 'go']
you
>>> |
```

Python Data Type – Tuple

Tuple is another data type which is a sequence of data similar to list. But it is immutable. That means data in a tuple is write protected. Data in a tuple is written using parenthesis and commas.

Course Outcome (CO): Display message on screen using Python Script on IDE.

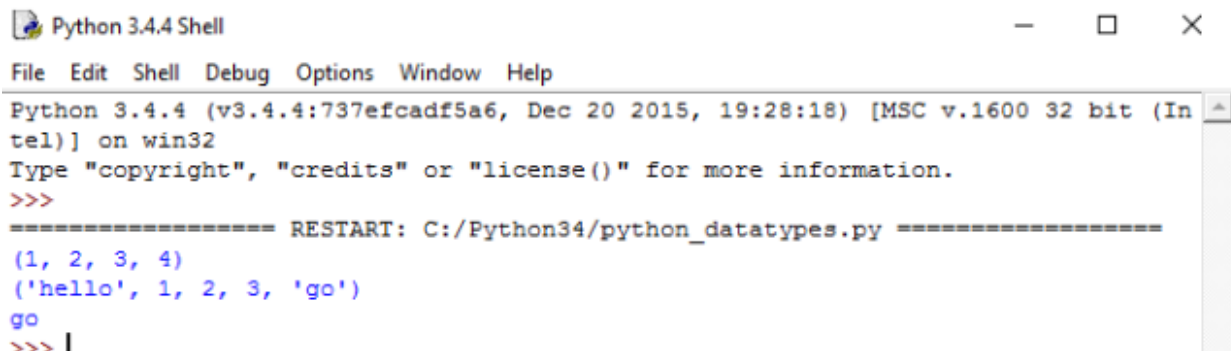
```
#tuple having only integer type of data.
a=(1,2,3,4)
print(a) #prints the whole tuple

#tuple having multiple type of data.
b=("hello", 1,2,3,"go")
print(b) #prints the whole tuple

#index of tuples are also 0 based.

print(b[4])
```

The output of this above python data type tuple example code will be like below image.



```
Python 3.4.4 Shell
File Edit Shell Debug Options Window Help
Python 3.4.4 (v3.4.4:737efcadf5a6, Dec 20 2015, 19:28:18) [MSC v.1600 32 bit (Intel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>>
===== RESTART: C:/Python34/python_datatypes.py =====
(1, 2, 3, 4)
('hello', 1, 2, 3, 'go')
go
>>> |
```

Dictionary

Python Dictionary is an unordered sequence of data of key-value pair form. It is similar to the hash table type. Dictionaries are written within curly braces in the form **key: value.**

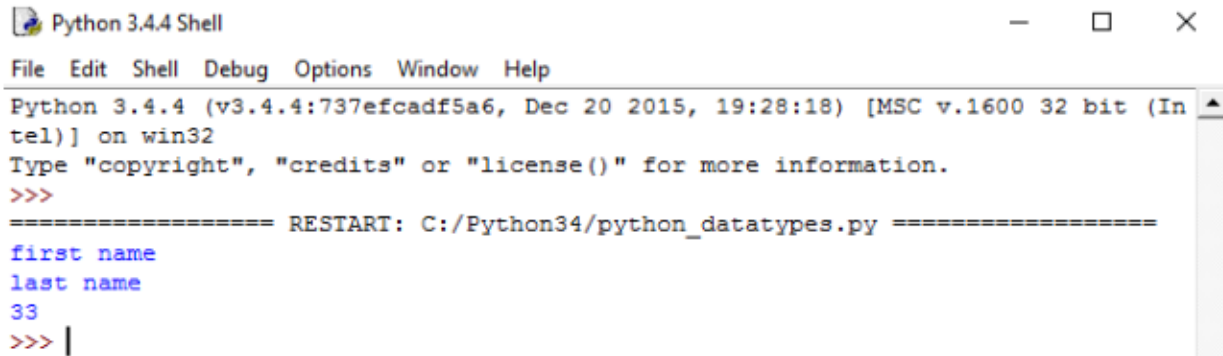
Course Outcome (CO): Display message on screen using Python Script on IDE.

```
#a sample dictionary variable

a = {1:"first name",2:"last name", "age":33}

#print value having key=1
print(a[1])
#print value having key=2
print(a[2])
#print value having key="age"
print(a["age"])
```

If you run this python dictionary data type example code, output will be like below image.

A screenshot of a Python 3.4.4 Shell window. The window title is "Python 3.4.4 Shell". The menu bar includes "File", "Edit", "Shell", "Debug", "Options", "Window", and "Help". The status bar at the bottom shows "Python 3.4.4 (v3.4.4:737efcadf5a6, Dec 20 2015, 19:28:18) [MSC v.1600 32 bit (Intel)] on win32". The main text area shows the output of a script: "Type 'copyright', 'credits' or 'license()' for more information.", followed by a restart message "===== RESTART: C:/Python34/python_datatypes.py =====", and then the printed values "first name", "last name", and "33". The prompt ">>>" is visible at the end of the line.

```
Python 3.4.4 Shell
File Edit Shell Debug Options Window Help
Python 3.4.4 (v3.4.4:737efcadf5a6, Dec 20 2015, 19:28:18) [MSC v.1600 32 bit (Intel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>>
===== RESTART: C:/Python34/python_datatypes.py =====
first name
last name
33
>>> |
```