

A stack of books with a blue background. The books are stacked on the left side of the image, with their spines facing right. The pages are a light brown color, and the spines are a darker brown. The background is a solid light blue color.

# Python 3.4

Learn Python With [Technical.com](http://Technical.com)



# Course Content

- Introduction to Python
- Application and Advantages of Python
- Keywords, Identifiers and Operators in Python
- Data Types in Python
- Control Structure and Looping in Python
- Functions in Python
- String, List, Tuples and Dictionaries in Python
- OOP programming in Python
- File Handling in Python
- Exception Handling and GUI Programming in Python



A stack of books with a blue background. The books are stacked on the left side of the image, with their spines facing right. The pages are a light brown color, and the spines are a darker brown. The background is a solid light blue color.

# Python Introduction



# What is Python ?

Python is a general purpose, dynamic, high level and interpreted programming language. It supports Object Oriented programming approach to develop applications.

Python supports multiple programming pattern, including object oriented, imperative and functional or procedural programming styles.

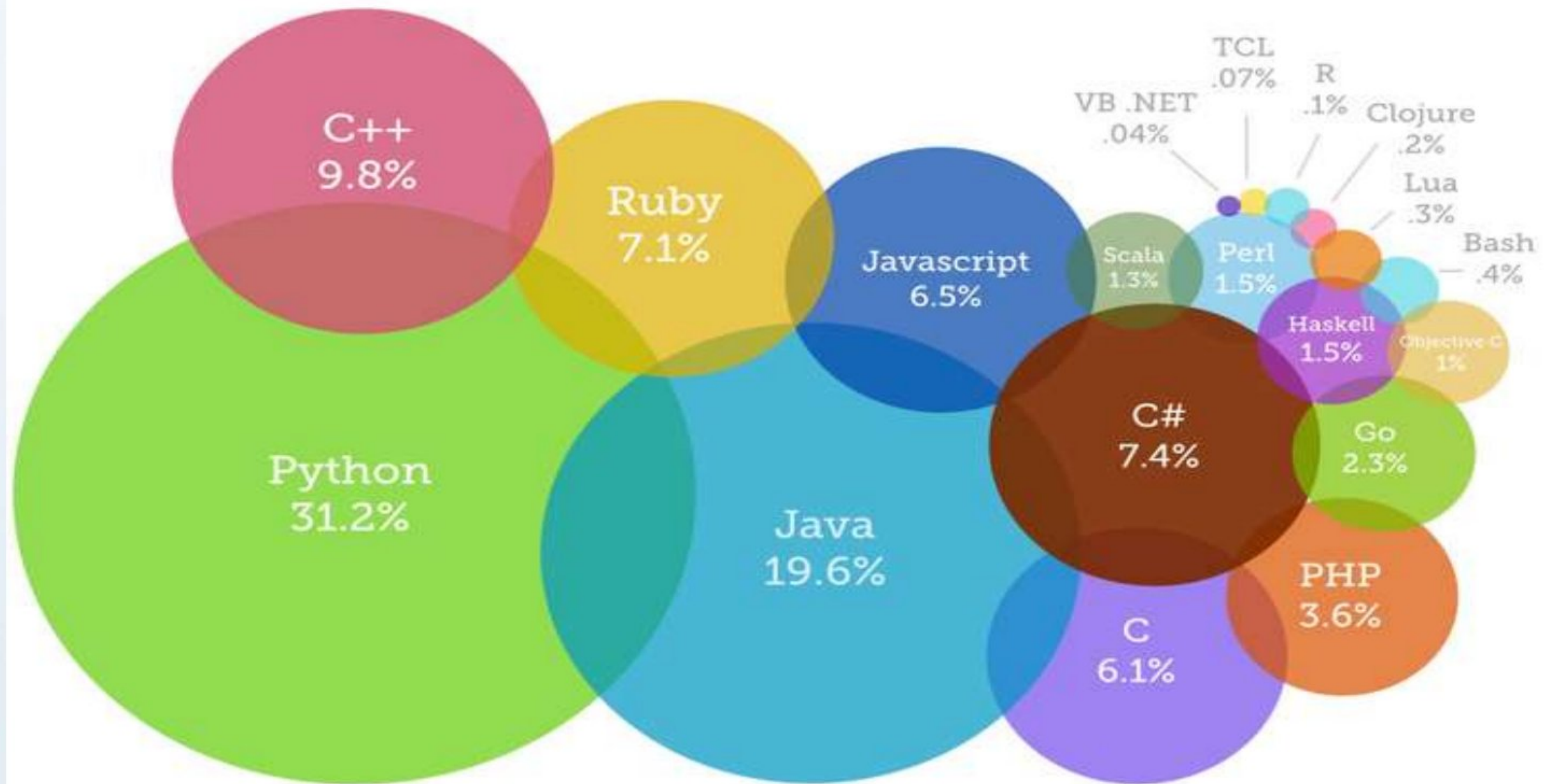
Python is not intended to work on special area such as web programming. That is why it is known as multipurpose because it can be used with web, enterprise, 3D CAD etc.

We don't need to use data types to declare variable because it is dynamically typed so we can write `a=10` to assign an integer value in an integer variable.

Python makes the development and debugging fast because there is no compilation step included in python development and edit-test-debug cycle is very fast.



## Most popular programming language of 2018





# Compiler Vs Interpreter

Compiler	Interpreter
Scans the entire program and translates it as a whole into machine code.	Translates program one statement at a time.
It takes large amount of time to analyze the source code but the overall execution time is comparatively faster.	It takes less amount of time to analyze the source code but the overall execution time is slower.
Generates intermediate object code which further requires linking, hence requires more memory.	No intermediate object code is generated, hence are memory efficient.
It generates the error message only after scanning the whole program. Hence debugging is comparatively hard.	Continues translating the program until the first error is met, in which case it stops. Hence debugging is easy.

- \* **Maximum scripting languages are interpreted language.**
- \* **All the compiled languages are True language.**



# Python History

- The implementation of Python was started in the December 1989 by Guido Van Rossum at CWI in Netherland.
- In February 1991, van Rossum published the code (labeled version 0.9.0) to alt.sources.
- In 1994, Python 1.0 was released with new features like: lambda, map, filter, and reduce.
- Python 2.0 added new features like: list comprehensions, garbage collection system.
- On December 3, 2008, Python 3.0 (also called "Py3K") was released. It was designed to rectify fundamental flaw of the language.
- Python is influenced by following programming languages:

**ABC language & Modula-3**



# Python Features

- Easy to Learn and Use
- Expressive Language
- Interpreted Language
- Cross-platform Language
- Free and Open Source
- Object-Oriented Language
- GUI Programming Support
- Large Standard Library



# Python Applications

- Web Applications
- Desktop GUI Applications
- Software Development
- Scientific and Numeric
- Business Applications
- Console Based Application
- Audio or Video based Applications
- 3D CAD Applications
- Applications for Images
- Enterprise Applications



# Python Versions

Python Version	Released Date
Python 1.0	January 1994
Python 1.5	December 31, 1997
Python 1.6	September 5, 2000
Python 2.0	October 16, 2000
Python 2.1	April 17, 2001
Python 2.2	December 21, 2001
Python 2.3	July 29, 2003
Python 2.4	November 30, 2004
Python 2.5	September 19, 2006
Python 2.6	October 1, 2008
Python 2.7	July 3, 2010
Python 3.0	December 3, 2008
Python 3.1	June 27, 2009
Python 3.2	February 20, 2011
Python 3.3	September 29, 2012
Python 3.4	March 16, 2014
Python 3.5	September 13, 2015
Python 3.6	December 23, 2016
Python 3.6.4	December 19, 2017



# Python 2.X vs 3.X

## Division operator

```
print 7 / 5
print -7 / 5
...
```

Output in Python 2.x  
1  
-2

Output in Python 3.x :  
1.4  
-1.4

## Error Handling

```
try:
    rajat
except NameError, err: # in Python 2.x
    print (err, 'Error Caused')

try:
    rajat
except NameError as err: # 'as' is needed in Python 3.x
    print (err, 'Error Caused')
```

## Print Function

```
Python 3.6.4 (v3.6.4:d48eceb, Dec 19 2017, 06:04:45) [MSC v.1900 32 bit (Intel)]
on win32
Type "copyright", "credits" or "license()" for more information.
>>> print "Welcome Friends" #in 2.x version of python
SyntaxError: Missing parentheses in call to 'print'. Did you mean print("Welcome
Friends" #in 2.x version of python)?
>>> print ("Welcome Friends") #in 3.x version of python

Welcome Friends
```

## Xrange

```
for x in xrange(1, 5):
    print(x),

for x in range(1, 5):
    print(x),
...
```

Output in Python 2.x  
1 2 3 4 1 2 3 4

Output in Python 3.x  
NameError: name 'xrange' is not defined



# Python Identifiers

A Python identifier is a name used to identify a variable, function, class, module, or other object. An identifier starts with a letter A to Z or a to z, or an underscore (\_) followed by zero or more letters, underscores and digits (0 to 9). Python does not allow punctuation characters such as @, \$, and % within identifiers. Python is a case sensitive programming language. Thus, **Car** and **car** are two different identifiers in Python.

Here are naming conventions for Python identifiers:

- Class names start with an uppercase letter. All other identifiers start with a lowercase letter.
- Starting an identifier with a single leading underscore indicates that the identifier is private.
- Starting an identifier with two leading underscores indicates a strongly private identifier.
- If the identifier also ends with two trailing underscores, the identifier is a language-defined special name.



# Keywords in Python

## Program:

```
variable.py - C:\Users\RAJAT TYAGI\AppData\Local\Programs\Python\Pyth
File Edit Format Run Options Window Help
import keyword
print(keyword.kwlist)
```

## Output:

```
Python 3.6.4 Shell
File Edit Shell Debug Options Window Help
Python 3.6.4 (v3.6.4:d48eceb, Dec 19 2017, 06:04:45) [MSC v.19
00 32 bit (Intel)] on win32
Type "copyright", "credits" or "license()" for more informatio
n.
>>>
RESTART: C:\Users\RAJAT TYAGI\AppData\Local\Programs\Python\Py
ython36-32\variable.py
['False', 'None', 'True', 'and', 'as', 'assert', 'break', 'cla
ss', 'continue', 'def', 'del', 'elif', 'else', 'except', 'fina
lly', 'for', 'from', 'global', 'if', 'import', 'in', 'is', 'la
mbda', 'nonlocal', 'not', 'or', 'pass', 'raise', 'return', 'tr
y', 'while', 'with', 'yield']
>>>
```

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



# Lines and 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 rigidly enforced.

```
variable.py - C:\Users\RAJAT TYAGI\AppData\Local\Pro
File Edit Format Run Options Window Help
if True:
    print('True')
else:
    print('False')
```

```
File Edit Shell Debug Options Window Help
Python 3.6.4 (v3.6.4:d48eceb, Dec 19 2017, 06:04:45) [MSC v.1900 32 bit (Intel)]
on win32
Type "copyright", "credits" or "license()" for more information.
>>>
RESTART: C:\Users\RAJAT TYAGI\AppData\Local\Programs\Python\Python36-32\variabl
e.py
True
```

## Multi-line statements

Statements in Python typically end with a new line. Python does, however, allow the use of the line continuation character (\) to denote that the line should continue. For example:

```
File Edit Format Run Options Window Help
a = 'hello \
users'
print(a)
b = ['Sachin', 'Himanshu',      #Statements contained within the [], {}, or () brack
    'Rajat']                  #continuation character. For example:
print(b)
```

```
File Edit Shell Debug Options Window
Python 3.6.4 (v3.6.4:d48eceb, Dec
on win32
Type "copyright", "credits" or "l
>>>
RESTART: C:\Users\RAJAT TYAGI\Ap
e.py
hello users
['Sachin', 'Himanshu', 'Rajat']
>>> |
```



# Quotation in Python

Python accepts single ('), double (") and triple (''' or """) quotes to denote string literals, as long as the same type of quote starts and ends the string.

```
variable.py - C:\Users\RAJAT TYAGI\AppData\Local\Programs\Python\Python36-32\variable....
File Edit Format Run Options Window Help
techhead='Sachin Tyagi'
cshead="Himanshu Tyagi"
trainer='''Rajat Tyagi'''
mkthead="""Chandan Singh"""
mgr="Sachin Chaudhary"
print(techhead)
print(cshead)
print(trainer)
print(mkthead)
print(mgr)
```

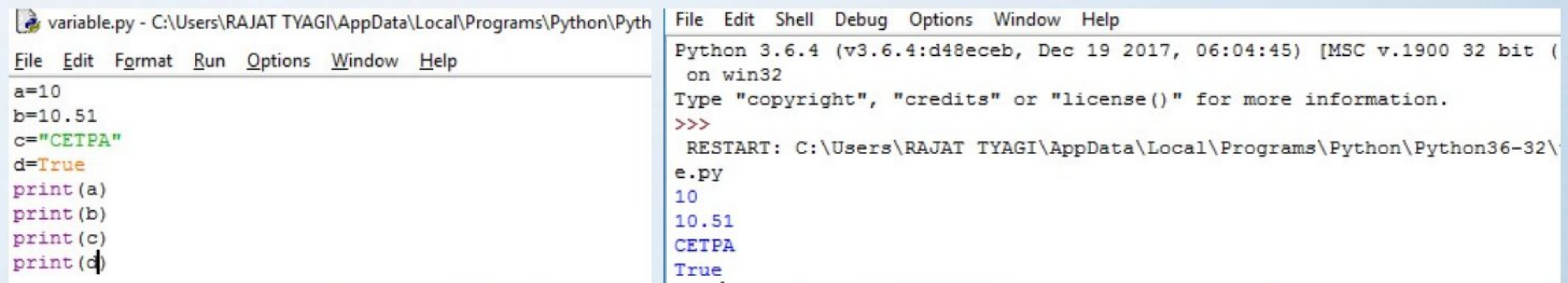
```
Python 3.6.4 Shell
File Edit Shell Debug Options Window Help
Python 3.6.4 (v3.6.4:d48eceb, Dec 19 2017, 06:04:45)
on win32
Type "copyright", "credits" or "license()" for more i
>>>
RESTART: C:\Users\RAJAT TYAGI\AppData\Local\Programs'
e.py
Sachin Tyagi
Himanshu Tyagi
Rajat Tyagi
Chandan Singh
"Sachin Chaudhary"
```



# Variables in Python

Variables are nothing but reserved memory locations to store values. This means when you create a variable, you reserve some space in memory.

Python variables do not need explicit declaration to reserve memory space. The declaration happens automatically when you assign a value to a variable. The equal sign (=) is used to assign values to variables.



The screenshot shows a Python IDE window titled 'variable.py - C:\Users\RAJAT TYAGI\AppData\Local\Programs\Python\Pyth'. The left pane contains the following code:

```
a=10
b=10.51
c="CETPA"
d=True
print(a)
print(b)
print(c)
print(d)
```

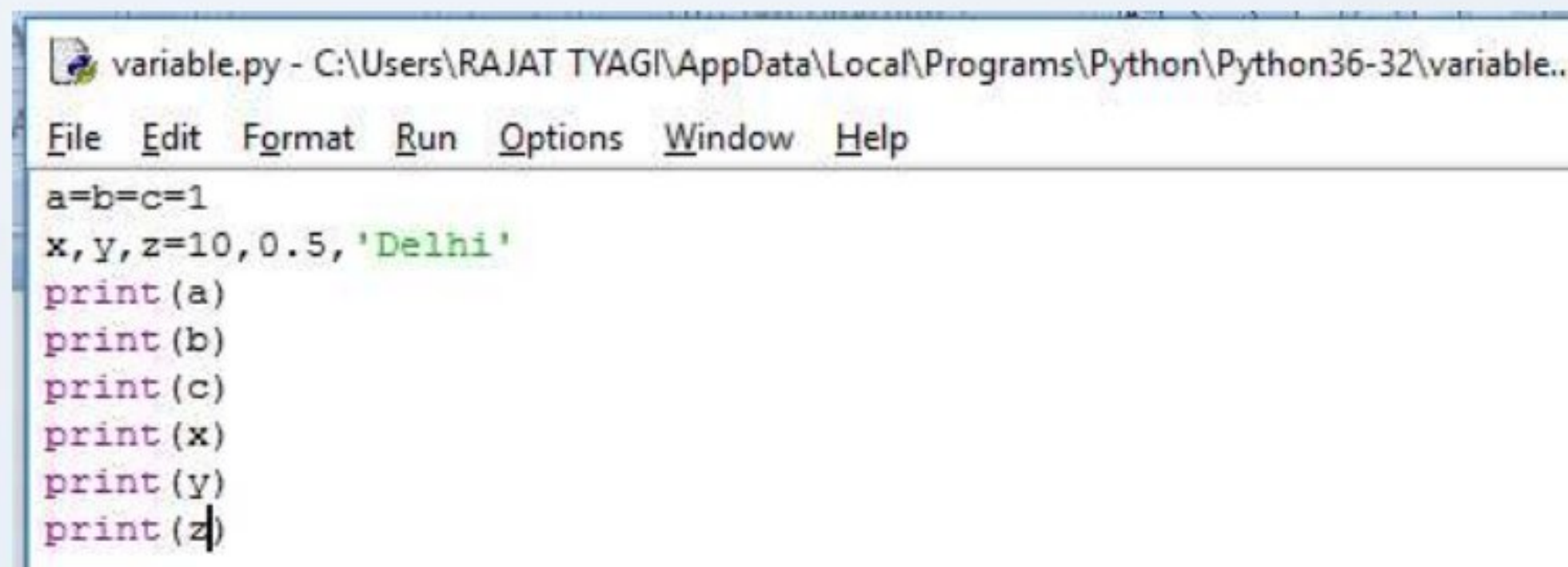
The right pane shows the output of the program:

```
Python 3.6.4 (v3.6.4:d48eceb, Dec 19 2017, 06:04:45) [MSC v.1900 32 bit (
on win32
Type "copyright", "credits" or "license()" for more information.
>>>
RESTART: C:\Users\RAJAT TYAGI\AppData\Local\Programs\Python\Python36-32\
e.py
10
10.51
CETPA
True
```

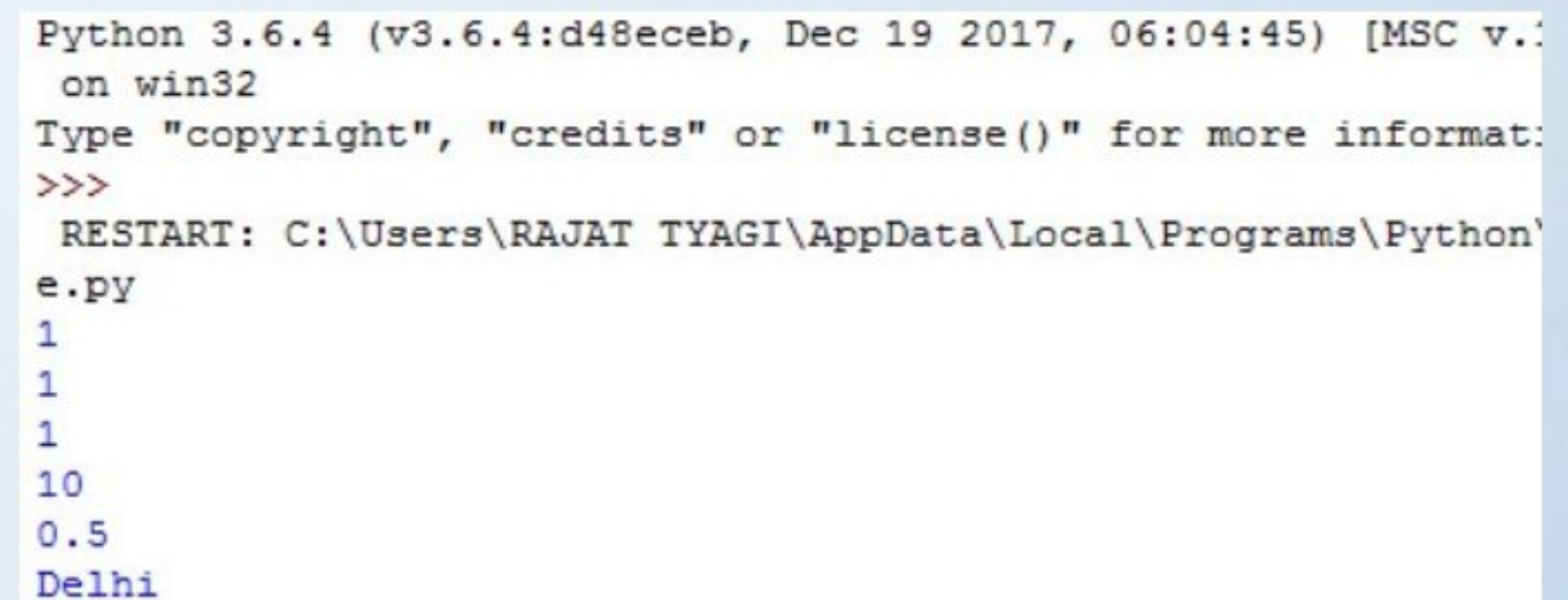


# Multiple Assignment in Python

Python allows you to assign a single value to several variables simultaneously. For example:



```
variable.py - C:\Users\RAJAT TYAGI\AppData\Local\Programs\Python\Python36-32\variable..  
File Edit Format Run Options Window Help  
a=b=c=1  
x,y,z=10,0.5,'Delhi'  
print(a)  
print(b)  
print(c)  
print(x)  
print(y)  
print(z)
```

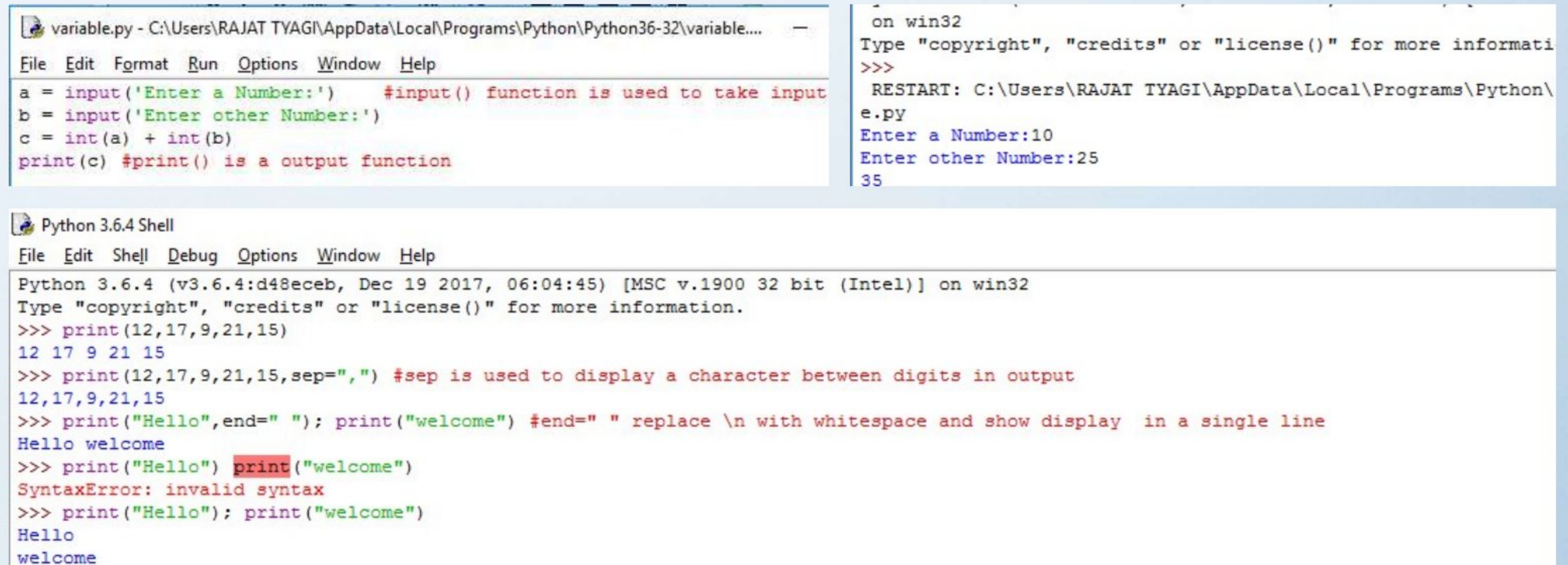


```
Python 3.6.4 (v3.6.4:d48eceb, Dec 19 2017, 06:04:45) [MSC v.1500664 on win32  
Type "copyright", "credits" or "license()" for more informati  
>>>  
RESTART: C:\Users\RAJAT TYAGI\AppData\Local\Programs\Python\Python36-32\python.exe  
e.py  
1  
1  
1  
10  
0.5  
Delhi
```



# Input and Output in Python

We use the `print()` function to output data to the standard output device (screen). To allow flexibility we might want to take the input from the user. In Python, we have the `input()` function to allow this.



The screenshot displays a Python IDE with two windows. The top window, titled 'variable.py', contains a script that takes two numbers as input and prints their sum. The bottom window, titled 'Python 3.6.4 Shell', shows the execution of this script, including user input and the resulting output.

```
variable.py - C:\Users\RAJAT TYAGI\AppData\Local\Programs\Python\Python36-32\variable....  
File Edit Format Run Options Window Help  
a = input('Enter a Number:') #input() function is used to take input  
b = input('Enter other Number:')  
c = int(a) + int(b)  
print(c) #print() is a output function
```

```
Python 3.6.4 Shell  
File Edit Shell Debug Options Window Help  
Python 3.6.4 (v3.6.4:d48eceb, Dec 19 2017, 06:04:45) [MSC v.1900 32 bit (Intel)] on win32  
Type "copyright", "credits" or "license()" for more information.  
>>> print(12,17,9,21,15)  
12 17 9 21 15  
>>> print(12,17,9,21,15,sep=",") #sep is used to display a character between digits in output  
12,17,9,21,15  
>>> print("Hello",end=" "); print("welcome") #end=" " replace \n with whitespace and show display in a single line  
Hello welcome  
>>> print("Hello") print("welcome")  
SyntaxError: invalid syntax  
>>> print("Hello"); print("welcome")  
Hello  
welcome
```



# Python Data Types

Python has various standard data types that are used to define the operations possible on them and the storage method for each of them.

Python has five standard data types:

- Numbers
- String
- Boolean
- List
- Tuple
- Set
- Dictionary