History of Python:

The name Python was selected from "Monty Python's Flying Circus" which was a British sketch comedy series created by the comedy group Monty Python and broadcast by the BBC from 1969 to 1974.

Python was created in the early 1990s by Guido van Rossum at the National Research Institute for Mathematics and Computer Science in Netherlands.

Python was created as a successor of a language called ABC (All Basic Code) and released publicly in1991. Guido remains Python's principal author, although it includes many contributions from active user community.

Between 1991 and 2001 there are several versions released, current stable release is 3.2. In 2001 the Python Software Foundation (PSF) was formed, a non-profit organization created specifically to own Python-related Intellectual Property. Zope Corporation is a sponsoring member of the PSF.

All most all Python releases are Open Source.

What is Python:

- Python is an open source, object-oriented, high-level powerful programming language.
- Developed by Guido van Rossum in the early 1990s. Named after Monty Python
- Python runs on many Unix variants, on the Mac, and on Windows 2000 and later.
- Available for download from http://www.python.org.
- Python programs are composed of modules
- Modules contain statements
- Statements contain expressions
- Expressions create and process objects

Features of Python:

Open source : Python is publicly available open source software, any one can use source code that doesn't't cost anything.

Easy-to-learn : Popular (scripting/extension) language, clear and easy syntax, no type declarations, automatic memory management, high-level data types and operations, design to read (more English like syntax) and write (shorter code compared to C, C++, and Java) fast.

High-levelLanguage: High-level language (closer to human) refers to the higher level of concept from machine language (for example assembly languages). Python is an example of a high-level language like C, C++, Perl, and Java with low-level optimization.

Portable: High level languages are portable, which means they are able to run across all major hardware and software platforms with few or no change in source code. Python is portable and can be used on Linux, Windows, Macintosh, Solaris, FreeBSD, OS/2, Amiga, AROS, AS/400 and many more.

Object-Oriented : Python is a full-featured object-oriented programming language, with features such as classes, inheritance, objects, and overloading.

Python is Interactive :Python has an interactive console where you get a Python prompt (command line) and interact with the interpreter directly to write and test your programs. This is useful for mathematical programming.

Interpreted : Python programs are interpreted, takes source code as input, and then compiles (to portable byte-code) each statement and executes it immediately. No need to compiling or linking

Extendable : Python is often referred to as a "glue" language, meaning that it is capable to work in mixed-language environment. The Python interpreter is easily extended and can add a new built-in function or modules written in C/C++/Java code.

Libraries : Databases, web services, networking, numerical packages, graphical user interfaces, 3D graphics, others.

Supports :Support from online Python community.

Need of Python programming:

Python is a general-purpose language, which means it can be used to build just about anything, which will be made easy with the right tools/libraries.

Professionally, Python is great for backend web development, data analysis, artificial intelligence, and scientific computing. Many developers have also used Python to build productivity tools, games, and desktop apps, so there are plenty of resources to help you learn how to do those as well.

Major uses of Python:

- System utilities (system admin tools, command line programs).
- Web Development.
- Graphical User Interfaces (Tkinter, gtk, Qt).
- Internet scripting.
- Embedded scripting.
- Database access and programming.
- Game programming.
- Rapid prototyping and development.
- Distributed programming

Applications of Python:

Web Development : Yahoo Maps, Yahoo Groups, Google, Zope Corporation, Ultraseek, Linux Weekly News, ElasticHosts Cloud Servers, Mojam.com, hunch, Shopzilla, Movieplayer.it, Multiplayer.it.

Games : Battlefield 2, Crystal Space, Star Trek Bridge Commander, The Temple of Elemental Evil, Vampire: The Masquerade: Bloodlines, Civilization 4, QuArK (Quake Army Knife)

Graphics : Industrial Light & Magic, Walt Disney Feature Animation, HKS, Inc. (ABAQUS/CAE), RoboFog, Caligari Corporation, Blender 3D, Jasc Software, Paint Shop Pro.
Financial : Altis Investment Management, ABN AMRO Bank, Treasury Systems, Bellco Credit Union, Journyx Timesheet and Resource Management Software.
Science : National Weather Service, Radar Remote Sensing Group, Applied Maths, Biosoft, The National Research Council of Canada, Los Alamos National Laboratory (LANL) Theoretical Physics Division, AlphaGene, Inc., LLNL, NASA, Swedish Meteorological and Hydrological Institute (SMHI), Environmental Systems Research Institute (ESRI), Objexx Engineering, Nmag Computational Micromagnetics
Electronic Design Automation : Ciranova, Productivity Design Tools, Object Domain, Pardus, Red Hat, SGI, Inc., MCI Worldcom, Nokia,
Education : University of California, Irvine, Smeal College of Business, The Pennsylvania State University, New Zealand Digital Library, IT Certification Exam preparation, SchoolTool,
Business Software : Raven Bear Systems Corporation, Thawte Consulting, Advanced Management Solutions Inc., IBM, Arakn<E9>, RealNetworks, dSPACE, Escom, The Tiny Company, Nexedi, Piensa Technologies - Bufete Consultor de Mexico, Nektra, WuBook.

Role of Interpreter of Python:

The word "interpreter" can be used in a variety of different ways when discussing Python. Sometimes interpreter refers to the Python REPL (**Read–Eval–Print Loop**), the interactive prompt you get by typing python at the command line. Sometimes people use "the Python interpreter" more or less interchangeably with "Python" to talk about executing Python code from start to finish.

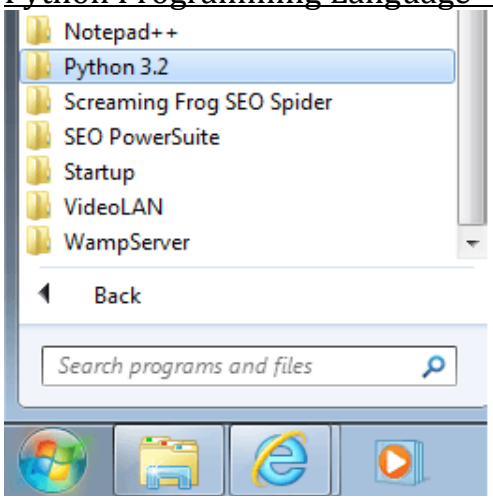
There are four steps that python takes when you hit return: lexing, parsing, compiling, and interpreting. Lexing is breaking the line of code you just typed into tokens. The parser takes those tokens and generates a structure that shows their relationship to each other (in this case, an Abstract Syntax Tree). The compiler then takes the AST and turns it into one (or more) code objects. Finally, the interpreter takes each code object executes the code it represents.

Basics of Python Programming Using the REPL(Shell):

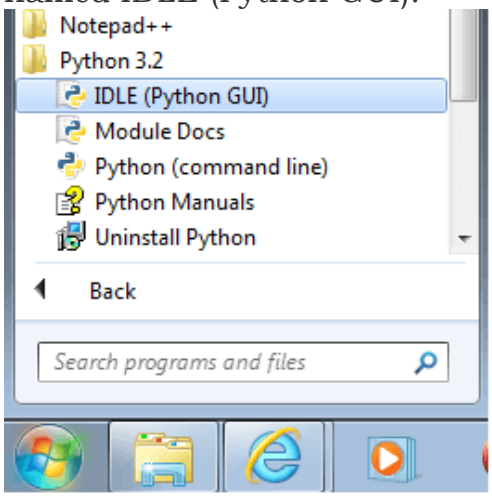
A Read-Eval-Print Loop (REPL) is a simple, interactive computer programming environment. The term 'REPL' is usually used to refer to a LISP interactive environment but can be applied to command line shells and similar environments for programming languages like Python, Ruby etc.

Python IDLE: Interactive Mode

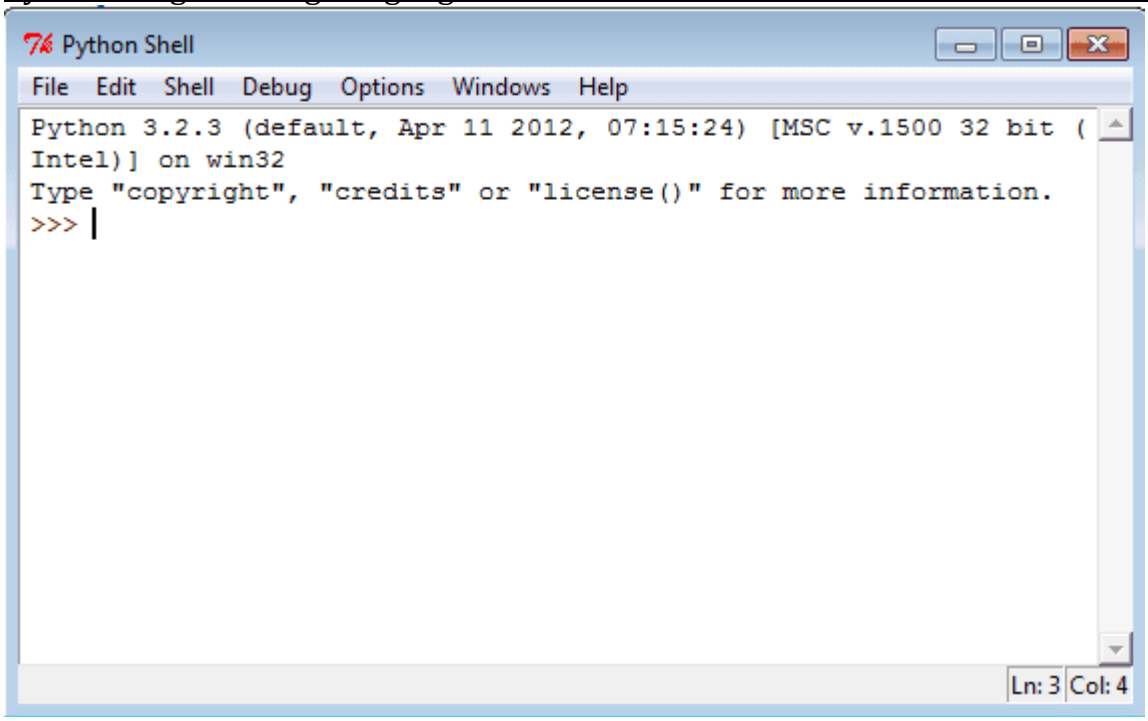
Let us assume that we've already installed Python (here we installed Python 3.2 on a standard pc with windows 7 OS). Click on start button and find Python 3.2 tab in installed programs.



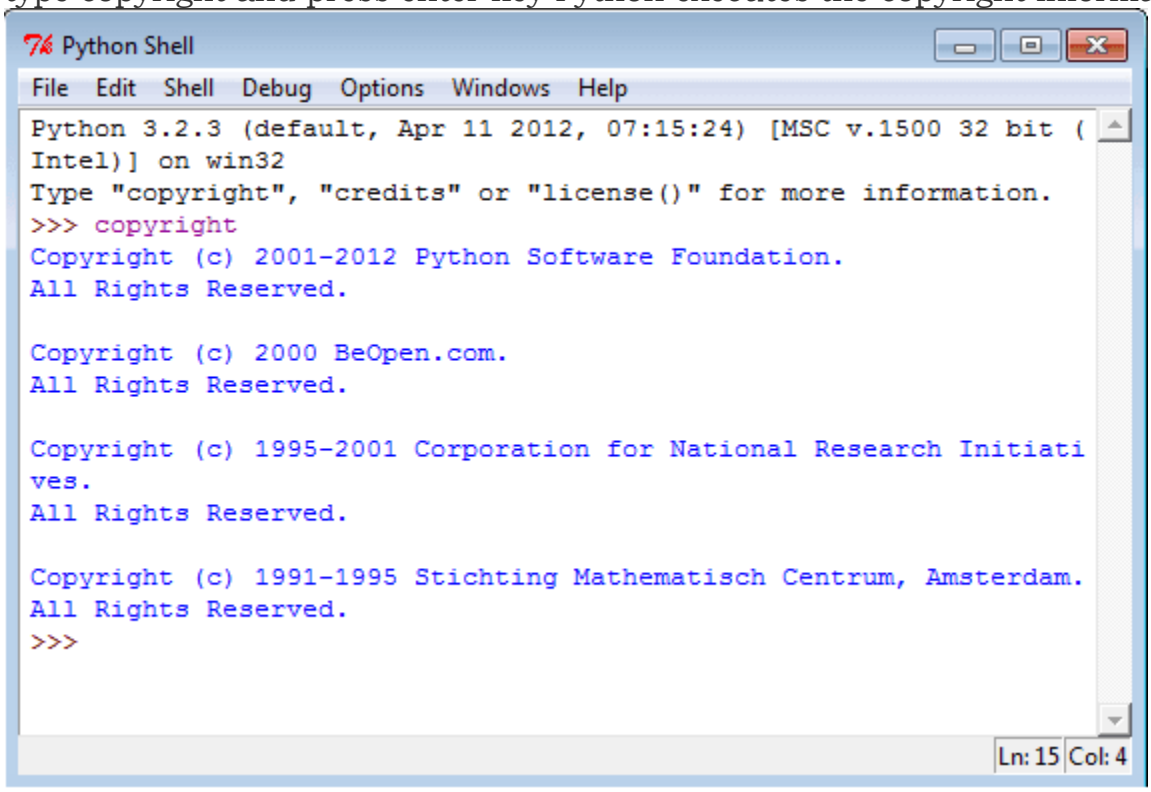
Now clicking on Python 3.2 tab you can see the Python program development tool named IDLE (Python GUI).



To start IDLE click on IDLE (Python GUI) icon, you will see a new window opens up and the cursor is waiting beside '>>>' sign which is called command prompt.



This mode is called interactive mode as you can interact with IDLE directly, you type something (single unit in a programming language) and press enter key Python will execute it, but you can not execute your entire program here. At the command prompt type copyright and press enter key Python executes the copyright information.

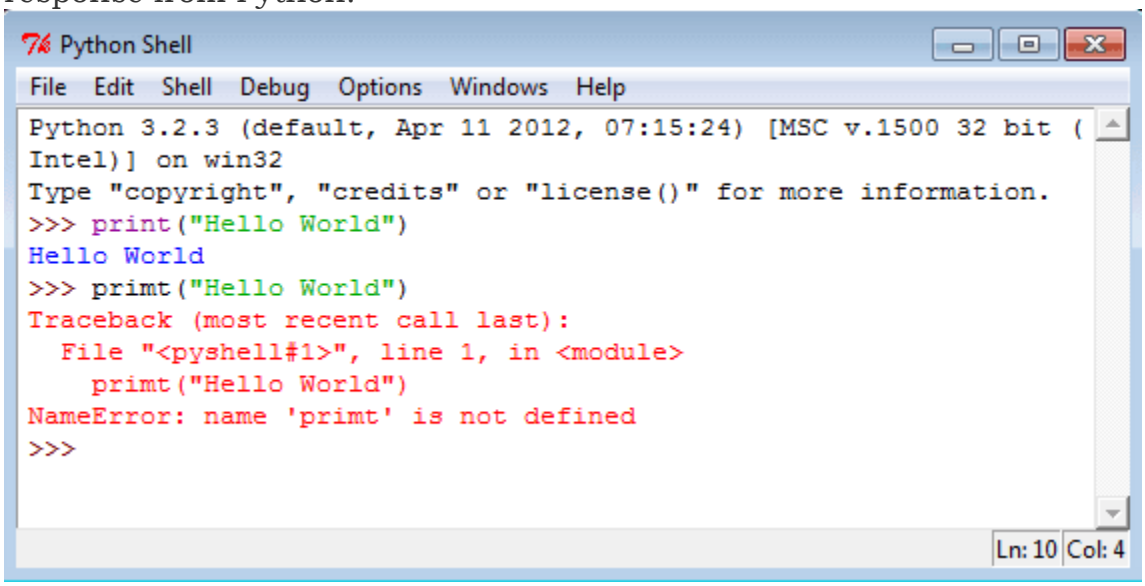


Now Python is ready to read new command. Let's execute the following commands one by one.

Command -1 : print("Hello World")

Command -2 : primt("Hello World")

The first command is correct and but the second one has a syntax error, here is the response from Python.

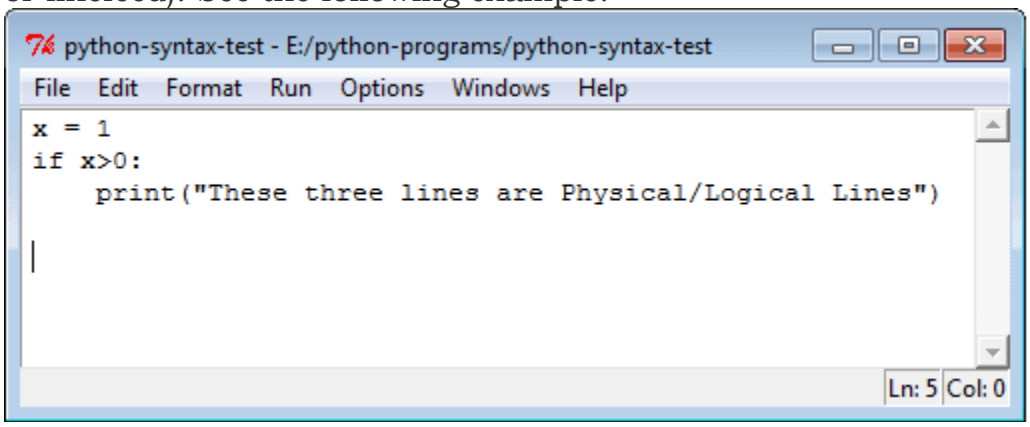


Python line structure:

A Python program is divided into a number of logical lines and every logical line is terminated by the token NEWLINE. A logical line is created from one or more physical lines.

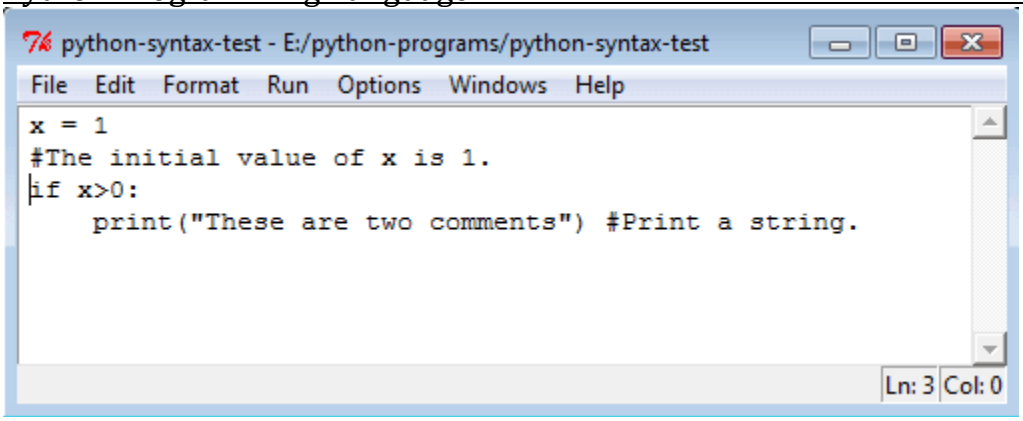
A line contains only spaces, tabs, formfeeds possibly a comment, is known as a blank line, and Python interpreter ignores it.

A physical line is a sequence of characters terminated by an end-of-line sequence (in windows it is called CR LF or return followed by a linefeed and in Unix, it is called LF or linefeed). See the following example.



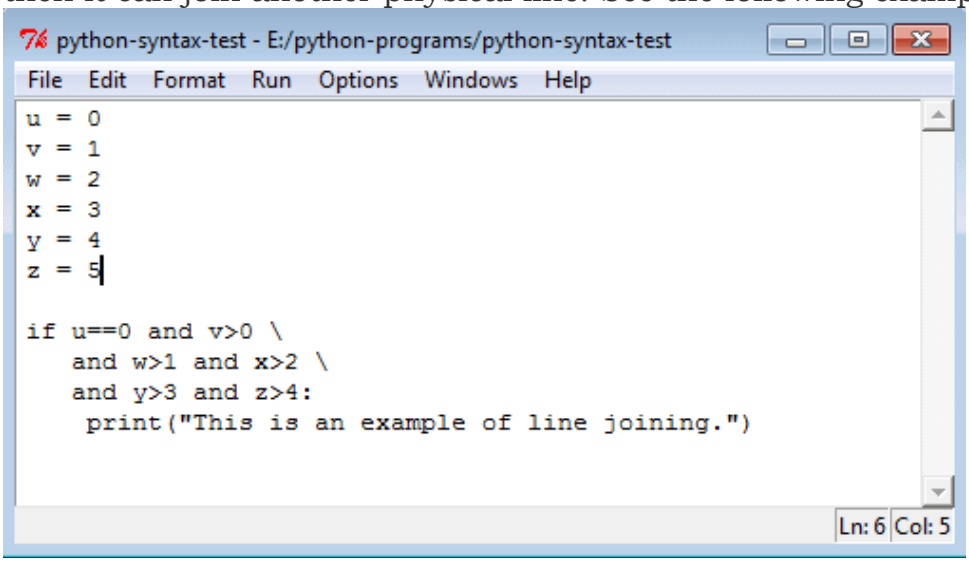
Comments in Python:

A comment begins with a hash character(#) which is not a part of the string literal and ends at the end of the physical line. All characters after the # character up to the end of the line are part of the comment and the Python interpreter ignores them. See the following example. It should be noted that Python has no multi-lines or block comments facility.



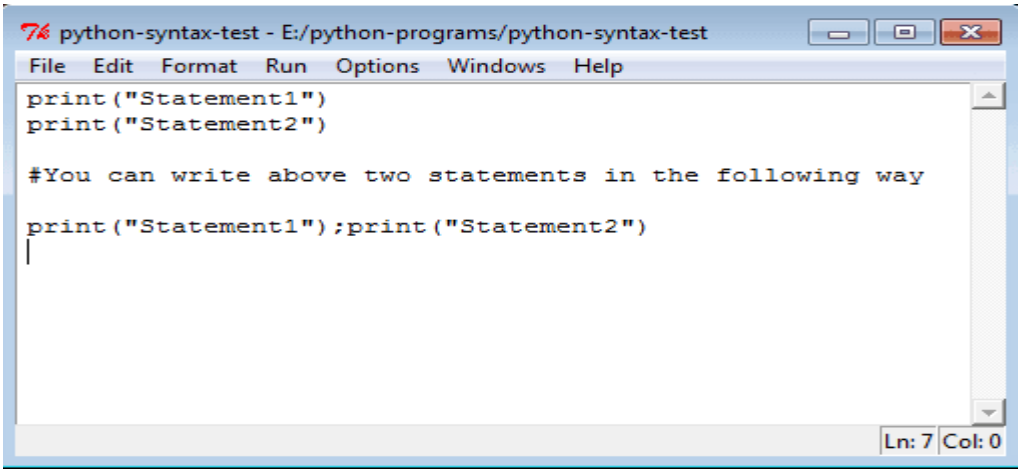
Joining two lines:

When you want to write a long code in a single line you can break the logical line in two or more physical lines using backslash character(\). Therefore when a physical line ends with a backslash characters(\) and not a part of a string literal or comment then it can join another physical line. See the following example.



Multiple statements on a single line:

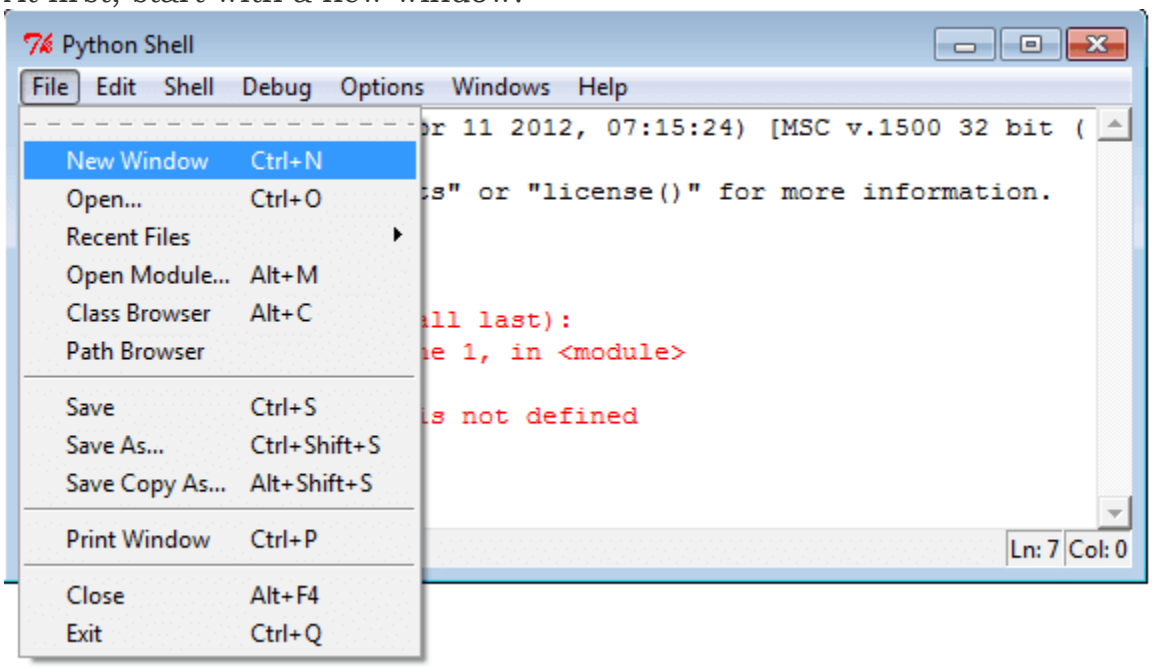
You can write two separate statements into a single line using a semicolon (;) character between two line.



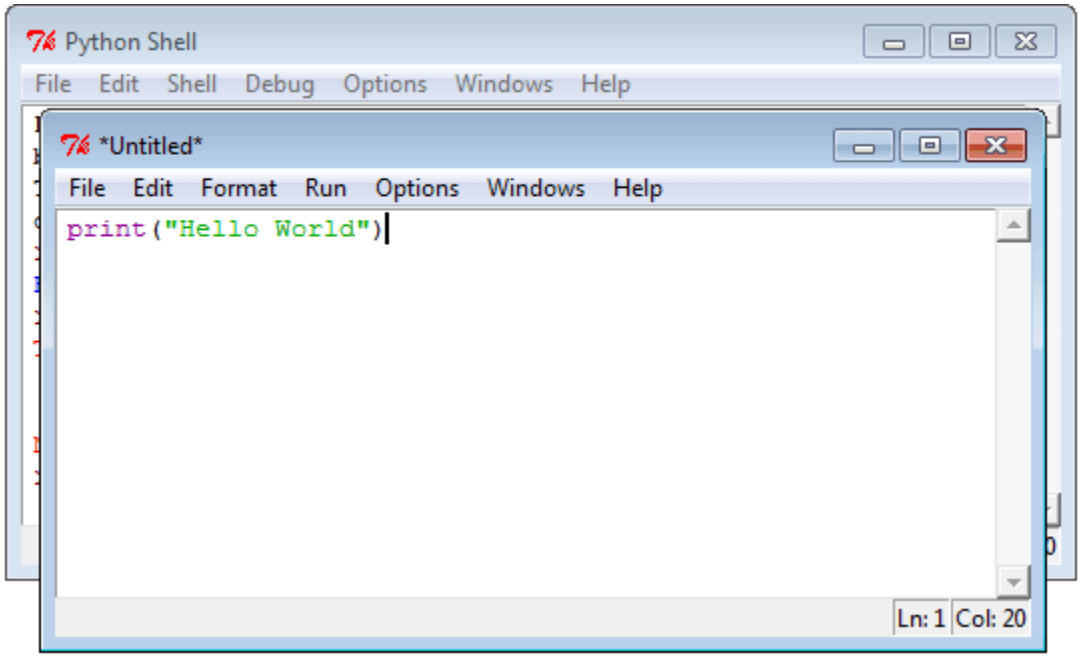
Running Python Scripts

Python IDLE: Development mode

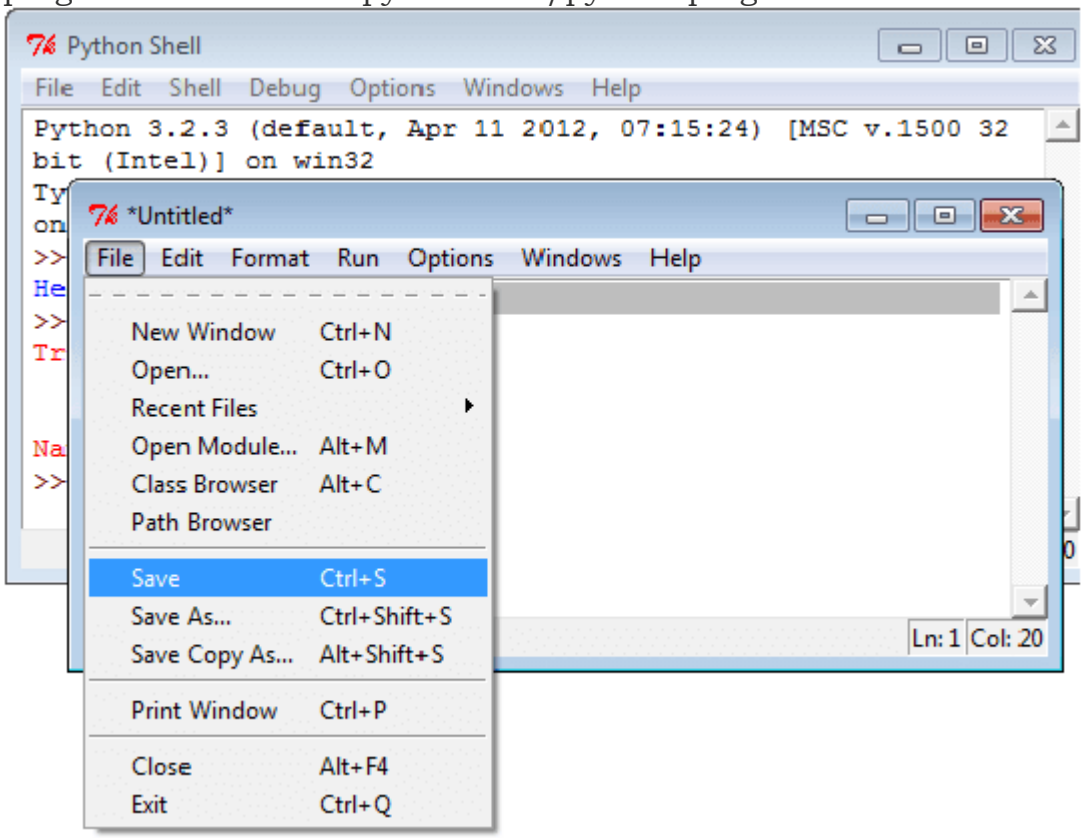
At first, start with a new window.



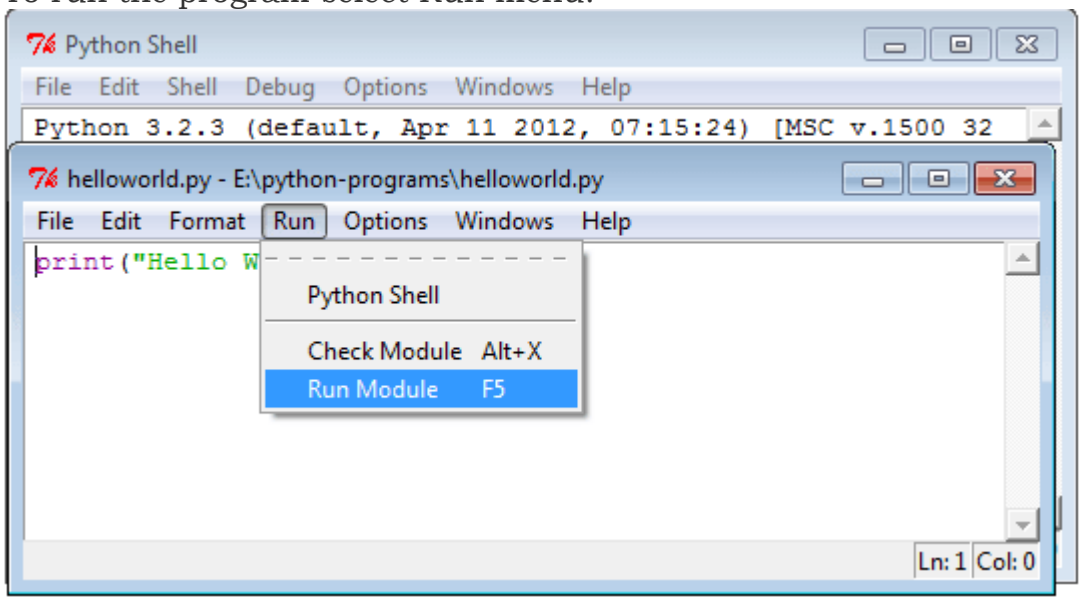
Clicking on "New window" under file menu a new window will come. Type print "Hello World" in the new window.



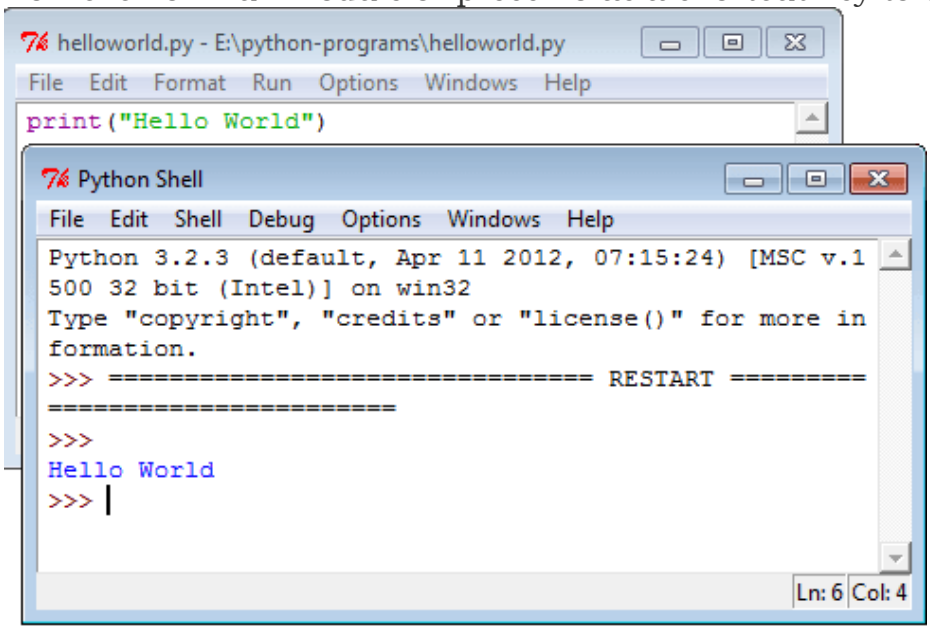
Let's save (Save command is located under the File menu) the file now. We save the program as helloworld.py under E:/python-programs folder.



To run the program select Run menu.

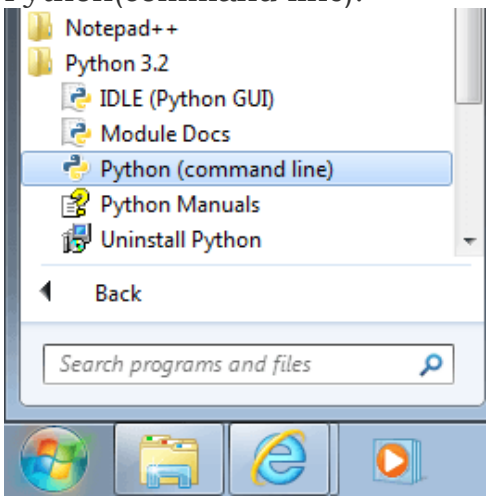


Now click on Run Module or press F5 as a shortcut key to see the result.

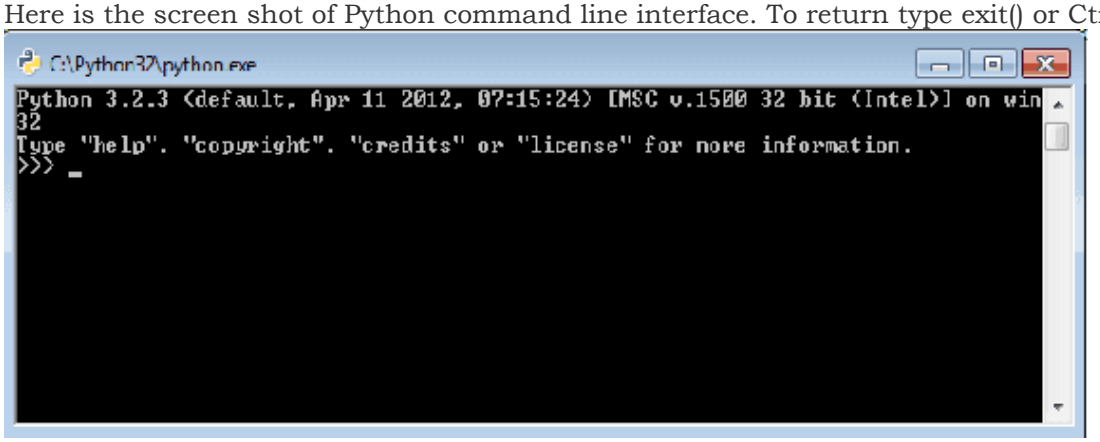


Python Command line interface:

There are some users who prefer command line intersection, rather than a GUI interface. To go Python command line, click on Python 3.2 tab then click on Python(command line).



Here is the screen shot of Python command line interface. To return type exit() or Ctrl+Z plus Enter key.



Python Variables:

- A variable is a memory location where a programmer can store a value. Example : roll_no, amount, name etc.
- Value is either string, numeric etc. Example : "Sara", 120, 25.36
- Variables are created when first assigned.
- Variables must be assigned before being referenced.
- The value stored in a variable can be accessed or updated later.
- No declaration required
- The type (string, int, float etc.) of the variable is determined by Python
- The interpreter allocates memory on the basis of the data type of a variable.

Variable name rules:

- Must begin with a letter (a - z, A - B) or underscore (_)
- Other characters can be letters, numbers or _
- Case Sensitive
- Can be any (reasonable) length
- There are some reserved words which you cannot use as a variable name because Python uses them for other things.

Python Assignment statements:

The assignment statement creates new variables and gives them values. Basic assignment statement in Python is :

Syntax

```
<variable> = <expr>
```

Where the equal sign (=) is used to assign value (right side) to a variable name (left side). See the following statements :

view plaincopy to clipboardprint?

```
>>> Item_name = "Computer" #A String
>>> Item_qty = 10 #An Integer
>>> Item_value = 1000.23 #A floating point
>>> print(Item_name)
Computer
>>> print(Item_qty)
10
>>> print(Item_value)
1000.23
>>>
```

One thing is important, assignment statement read right to left only.

Example :

a = 12 is correct, but 12 = a does not make sense to Python, which creates a syntax error. Check it in Python Shell.

view plaincopy to clipboardprint?

```
>>> a = 12
>>> 12 = a
SyntaxError: can't assign to literal
>>>
```

Multiple Assignment:

The basic assignment statement works for a single variable and a single expression. You can also assign a single value to more than one variables simultaneously.

Syntax

```
var1=var2=var3...varn= = <expr>
```

Example :

```
x = y = z = 1
```

Now check the individual value in Python Shell.

view plaincopy to clipboardprint?

```
>>> x = y = z = 1
>>> print(x)
1
>>> print(y)
1
>>> print(z)
1
>>>
```

Here is an another assignment statement where the variables assign many values at the same time.

Syntax

```
<var>, <var>, ..., <var> = <expr>, <expr>, ..., <expr>
```

Example :

```
x, y, z = 1, 2, "abcd"
```

In the above example x, y and z simultaneously get the new values 1, 2 and "abcd".

view plaincopy to clipboardprint?

```
>>> x,y,z = 1,2,"abcd"
>>> print(x)
1
>>> print(y)
2
>>> print(z)
abcd
```

You can reuse variable names by simply assigning a new value to them :

view plaincopy to clipboardprint?

```
>>> x = 100
>>> print(x)
100
>>> x = "Python"
>>> print(x)
Python
>>>
```

Swap variables:

Python swap values in a single line and this applies to all objects in python.

Syntax

```
var1, var2 = var2, var1
```

Example :

view plaincopy to clipboardprint?

```
>>> x = 10
>>> y = 20
>>> print(x)
10
>>> print(y)
20
>>> x, y = y, x
>>> print(x)
20
        >>> print(y)
        10
        >>>
```

Local and global variables in python:

In Python, variables that are only referenced inside a function are implicitly global. If a variable is assigned a value anywhere within the function's body, it's assumed to be a local unless explicitly declared as global.

Example :

```
var1 = "Python"
def func1():
    var1 = "PHP"
    print("In side func1() var1 = ",var1)

def func2():
    print("In side func2() var1 = ",var1)
func1()
func2()
```

Output :

```
In side func1() var1 =  PHP
In side func2() var1 =  Python
```

You can use a global variable in other functions by declaring it as global keyword :

Example :

```
def func1():
    global var1
    var1 = "PHP"
    print("In side func1() var1 = ",var1)

def func2():
    print("In side func2() var1 = ",var1)
func1()
func2()
```

Output :

```
In side func1() var1 =  PHP
In side func2() var1 =  PHP
```

Python Reserved words/ Key words:

The following identifiers are used as reserved words of the language, and cannot be used as ordinary identifiers.

False	class	finally	is	return
None	continue	for	lambda	try
True	def	from	nonlocal	while
And	del	global	not	with
As	el	if	or	yield
Assert	else	import	pass	
Break	except	in	raise	

Input and Output functions:

Python provides numerous built-in functions that are readily available to us at the python prompt.

Some of the functions like input() and print() are widely used for standard input and output operations respectively. Let us see the output section first.

Python Output Using print() function

We use the print() function to output data to the standard output device (screen).

```
print('This sentence is output to the screen')
# Output: This sentence is output to the screen
a = 5
print('The value of a is', a)
# Output: The value of a is 5
```

In the second print() statement, we can notice that a space was added between the string and the value of variable a.This is by default, but we can change it.

The actual syntax of the print() function is

```
print(*objects, sep=' ', end='\n', file=sys.stdout, flush=False)
```

Here, objects is the value(s) to be printed.

The sep separator is used between the values. It defaults into a space character.

After all values are printed, end is printed. It defaults into a new line.

The file is the object where the values are printed and its default value is sys.stdout(screen). Here are an example to illustrate this.

```
print(1,2,3,4)
# Output: 1 2 3 4
print(1,2,3,4,sep='*')
# Output: 1*2*3*4
print(1,2,3,4,sep='#',end='&')
# Output: 1#2#3#4&
```

Output formatting:

Sometimes we would like to format our output to make it look attractive. This can be done by using the str.format() method. This method is visible to any string object.

```
>>> x = 5; y = 10
>>> print('The value of x is {} and y is {}'.format(x,y))
The value of x is 5 and y is 10
```

Here the curly braces {} are used as placeholders. We can specify the order in which it is printed by using numbers (tuple index)

```
print('I love {0} and {1}'.format('bread','butter'))
# Output: I love bread and butter
print('I love {1} and {0}'.format('bread','butter'))
# Output: I love butter and bread
```

We can even use keyword arguments to format the string.

```
>>> print('Hello {name}, {greeting}'.format(greeting = 'Goodmorning', name =
'John'))

Hello John, Goodmorning
```

We can even format strings like the old `sprintf()` style used in C programming language. We use the `%` operator to accomplish this.

```
>>> x = 12.3456789

>>> print('The value of x is %3.2f' %x)

The value of x is 12.35

>>> print('The value of x is %3.4f' %x)

The value of x is 12.3457
```

Python Input:

Up till now, our programs were static. The value of variables were defined or hard coded into the source code.

To allow flexibility we might want to take the input from the user. In Python, we have the `input()` function to allow this. The syntax for `input()` is

```
input([prompt])
```

where `prompt` is the string we wish to display on the screen. It is optional.

```
>>> num = input('Enter a number: ')
Enter a number: 10
>>> num
'10'
```

Here, we can see that the entered value `10` is a string, not a number. To convert this into a number we can use `int()` or `float()` functions.

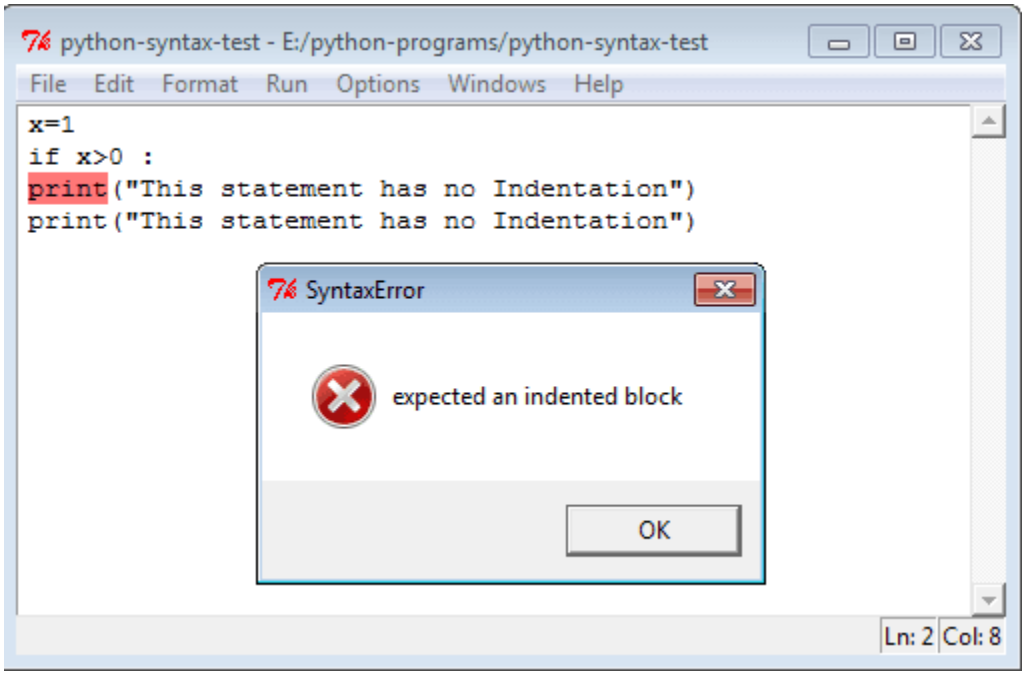
```
>>> int('10')
10
>>> float('10')
10.0
```

This same operation can be performed using the `eval()` function. But it takes it further. It can evaluate even expressions, provided the input is a string

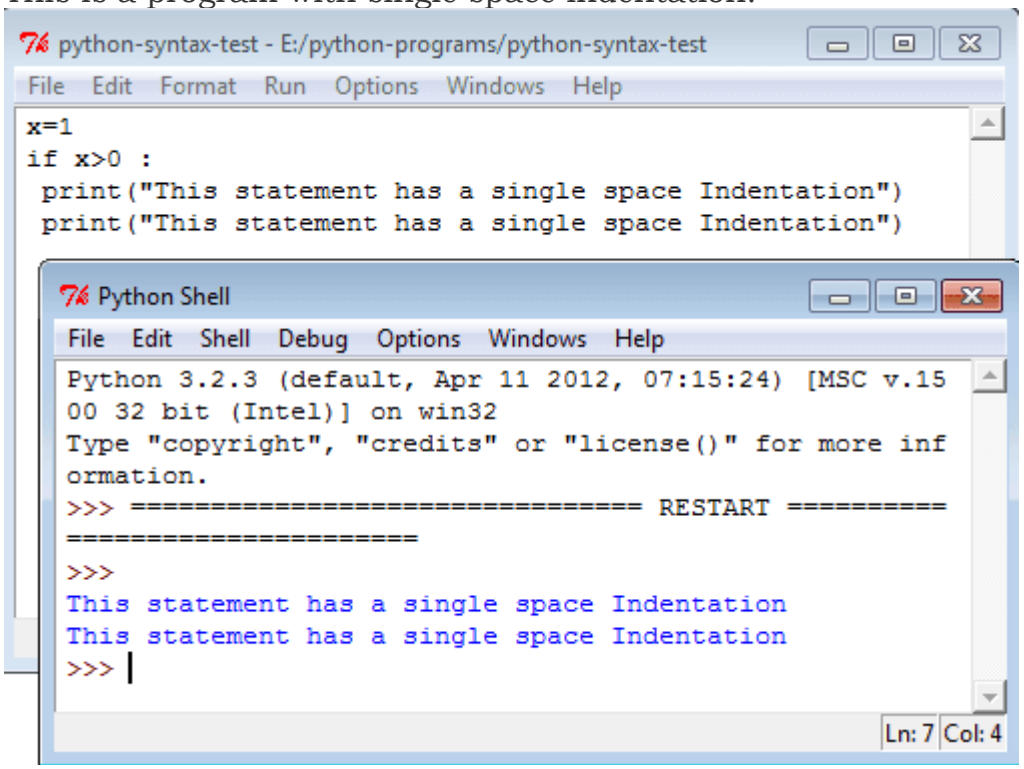
```
>>> int('2+3')
Traceback (most recent call last):
  File "<string>", line 301, in runcode
  File "<interactive input>", line 1, in <module>
ValueError: invalid literal for int() with base 10: '2+3'
>>> eval('2+3')
5
```

Indentation:

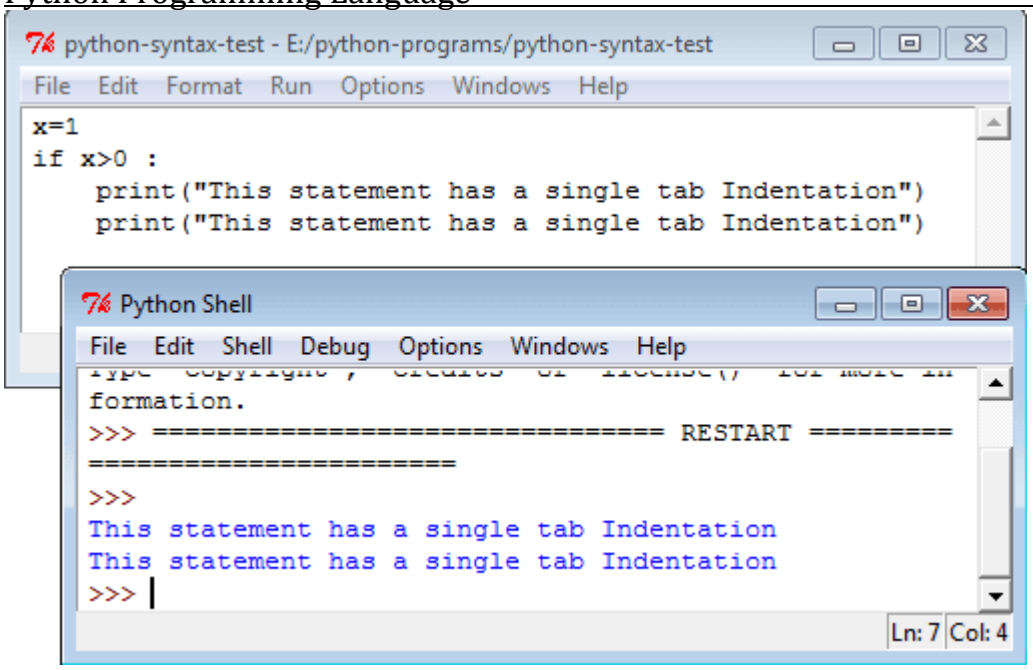
Python uses whitespace (spaces and tabs) to define program blocks whereas other languages like C, C++ use braces ({}) to indicate blocks of codes for class, functions or flow control. The number of whitespaces (spaces and tabs) in the indentation is not fixed, but all statements within the block must be the indented same amount. In the following program, the block statements have no indentation.



This is a program with single space indentation.



This is a program with single tab indentation.



Here is an another program with an indentation of a single space + a single tab.