**Basic rules for Python variables:**

• A variable name must start with a letter or the underscore character   
• A variable name cannot start with a number   
• A variable name can only contain alpha-numeric characters (A-z, 0-9) and underscores   
• Variable names are case-sensitive, e.g., amount, Amount and AMOUNT are three different variables.  
• A variable name cannot be any of the [Python keywords](https://www.w3schools.com/python/python_ref_keywords.asp).

**Remember that variable names are case-sensitive**

**Camel Case**

Each word, except the first, starts with a capital letter:

myVariableName = “John”

**Pascal Case**

Each word starts with a capital letter:

MyVariableName = "John"

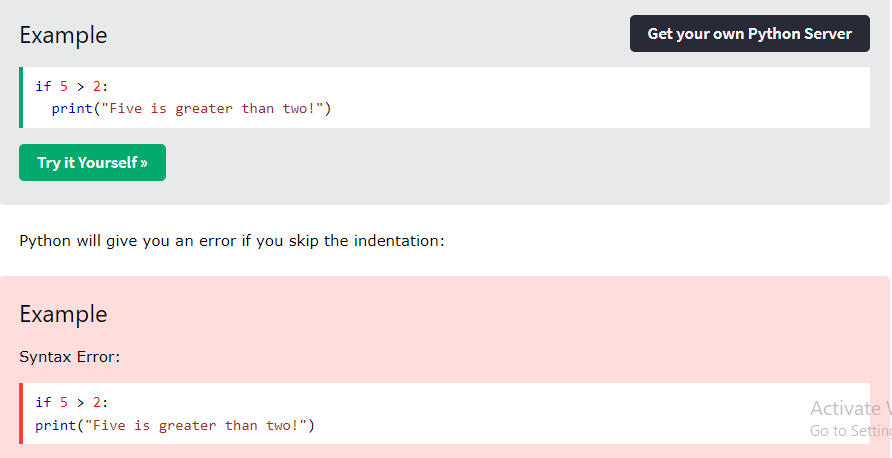
**Snake Case**

Each word is separated by an underscore character:

my\_variable\_name = "John"

**Python Indentation**

Indentation refers to the spaces at the beginning of a code line.  
Where in other programming languages the indentation in code is for readability only, the indentation in Python is very important.  
Python uses indentation to indicate a block of code.



**----------------------------------------------------------------------------------------**

**Casting**

## If you want to specify the data type of a variable, this can be done with casting

x = int(1.4)  
y= float("3")  
z = str(2)  
  
print(x)  
print(y)  
print(z)

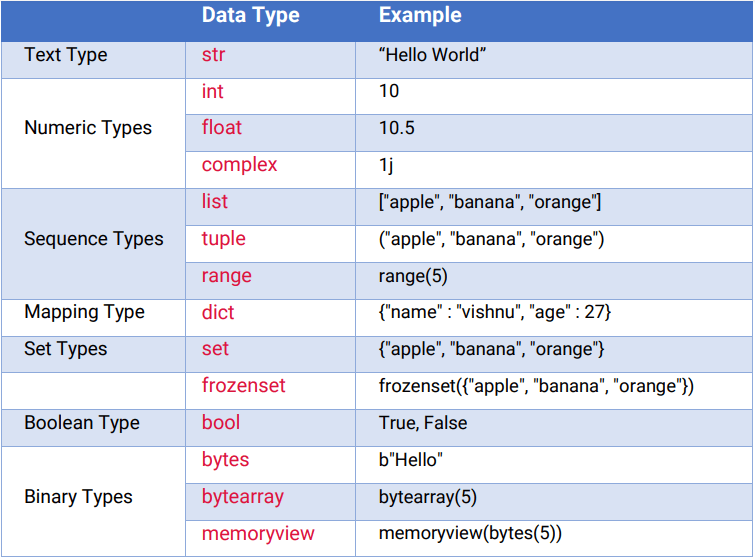
1, 3.0, 2

**----------------------------------------------------------------------------------------**

**Get the Type**

**x = 1   
y = 2.8   
z = 3 + 2j**

**print(type(x))   
print(type(y))   
print(type(z))  
  
<class 'int'>**  
**<class 'float'>**  
**<class 'complex'>**



**----------------------------------------------------------------------------------------**

**Python Variables - Assign Multiple Values**

**x, y, z = "Orange", "Banana", "Cherry"  
print(x)  
print(y)  
print(z)**

**Orange  
Banana  
Cherry**

**x = y = z = "Orange"**

**print(x)  
print(y)  
print(z)**

**Orange  
Orange  
Orange**

## Unpack a Collection

If you have a collection of values in a list, tuple etc. Python allows you to extract the values into variables. This is called *unpacking*.

fruits = ["apple", "banana", "cherry"]  
x, y, z = fruits  
print(x)  
print(y)  
print(z)

**apple  
banana  
cherry**

x = "Python"  
y = "is"  
z = "awesome"  
print(x, y, z)

**Python is awesome**

**x = "Python "  
y = "is "  
z = "awesome"  
print**(x + y + z)

**Python is awesome**

x = 5  
y = 10  
print(x + y)  
  
OUTPUT :15  
**-------------------------------------------------------**  
x = 5  
y = "John"  
print(x + y)

**TypeError: unsupported operand type(s) for +: 'int' and 'str**

## Global Variables

x = "awesome"  
def myfunc():  
  print("Python is " + x)  
myfunc()

**Python is awesome**

x = "awesome"  
def myfunc():  
  x = "fantastic"  
  print("Python is " + x)  
myfunc()  
print("Python is " + x)

**Python is fantastic   
Python is awesome**

**def myfunc():  
 global x  
 x = "fantastic"  
myfunc()  
print("Python is " + x)**

**Python is fantastic**

### x = "awesome" def myfunc(): global x x = "fantastic" myfunc( ) print("Python is " + x)

**Python is fantastic**The variable  **x** is initially set to **"awesome".**Inside **the myfunc() function, the global keyword is used to indicate that the function will modify the global variable x** instead of creating a new local variable.  
When **myfunc()** is called, it changes the value of **x to "fantastic".**  
Finally, the print statement concatenates **"Python is "** with the updated global value of x, which is now **"fantastic"**

x = "awesome"  
  
def myfunc():  
 global x  
 print("Python is " + x) **# Print "awesome" before changing it**

x = "fantastic"  
  
myfunc()  
  
print("Python is " + x)

Python is awesome

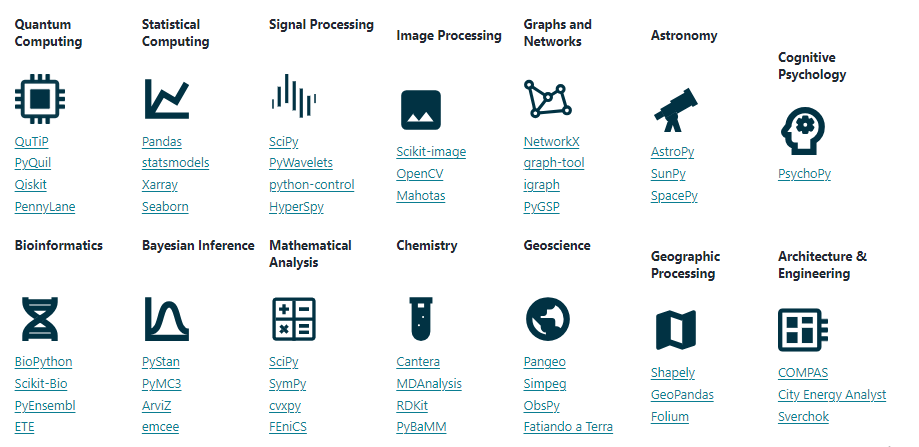
Python is fantastic

**String Input**

**print("Enter your name: ")  
x = input( )  
print("Hello, " + x)**

import random  
print(random.randrange(1, 10))

**NumPy** - NumPy is the fundamental package for scientific computing with Python



**• SciPy - SciPy is** a free and open-source Python library used for scientific computing and technical computing. SciPy contains modules for optimization, linear algebra, integration, interpolation, special functions, FFT, signal and image processing, ODE solvers and other tasks common in science and engineering

**Matplotlib**

- Matplotlib is a Python 2D plotting library  
 **example**: <https://matplotlib.org/stable/plot_types/index.html>