Python numbers:

a = 5  
print(a, "is of type", type(a))  
  
a = 2.0  
print(a, "is of type", type(a))  
  
a = 1+2j  
print(a, "is complex number?", isinstance(1+2j,complex))

**python list** : [List](https://www.programiz.com/python-programming/list) is an ordered sequence of items. It is one of the most used datatype in Python and is very flexible. All the items in a list do not need to be of the same type.

Declaring a list is pretty straight forward. Items separated by commas are enclosed within brackets [ ].

slicing operator [ ]

a = [5,10,15,20,25,30,35,40]  
  
# a[2] = 15  
print("a[2] = ", a[2])  
  
# a[0:3] = [5, 10, 15]  
print("a[0:3] = ", a[0:3])  
  
# a[5:] = [30, 35, 40]  
print("a[5:] = ", a[5:])

## Python Tuple

[Tuple](https://www.programiz.com/python-programming/tuple) is an ordered sequence of items same as a list. The only difference is that tuples are immutable. Tuples once created cannot be modified.

Tuples are used to write-protect data and are usually faster than lists as they cannot change dynamically.It is defined within parentheses () where items are separated by commas.

t = (5,'program', 1+3j)  
  
# t[1] = 'program'  
print("t[1] = ", t[1])  
  
# t[0:3] = (5, 'program', (1+3j))  
print("t[0:3] = ", t[0:3])

## Python Strings

sequence of Unicode characters. We can use single quotes or double quotes to represent strings. Multi-line strings can be denoted using triple quotes, ''' or """.

s = "This is a string"  
print(s)  
s = '''A multiline  
string'''  
print(s)

## Python Set

[Set](https://www.programiz.com/python-programming/set) is an unordered collection of unique items. Set is defined by values separated by comma inside braces { }. Items in a set are not ordered.

a = {5,2,3,1,4}  
  
# printing set variable  
print("a = ", a)  
  
# data type of variable a  
print(type(a))

## Python Dictionary

[Dictionary](https://www.programiz.com/python-programming/dictionary) is an unordered collection of key-value pairs.

It is generally used when we have a huge amount of data. Dictionaries are optimized for retrieving data. We must know the key to retrieve the value.

In Python, dictionaries are defined within braces {} with each item being a pair in the form key:value. Key and value can be of any type.

d = {1:'value','key':2}  
print(type(d))  
  
print("d[1] = ", d[1])  
  
print("d['key'] = ", d['key'])  
  
# Generates error  
print("d[2] = ", d[2])

## Type Conversion

**Key Points to Remember**

1. Type Conversion is the conversion of object from one data type to another data type.
2. Implicit Type Conversion is automatically performed by the Python interpreter.
3. Python avoids the loss of data in Implicit Type Conversion.
4. Explicit Type Conversion is also called Type Casting, the data types of objects are converted using predefined functions by the user.
5. In Type Casting, loss of data may occur as we enforce the object to a specific data type.

### Arithmetic operators in Python

x = 15  
y = 4  
  
# Output: x + y = 19  
print('x + y =',x+y)  
  
# Output: x - y = 11  
print('x - y =',x-y)  
  
# Output: x \* y = 60  
print('x \* y =',x\*y)  
  
# Output: x / y = 3.75  
print('x / y =',x/y)  
  
# Output: x // y = 3  
print('x // y =',x//y)  
  
# Output: x \*\* y = 50625  
print('x \*\* y =',x\*\*y)

### Comparison operators in Python

x = 10  
y = 12  
  
# Output: x > y is False  
print('x > y is',x>y)  
  
# Output: x < y is True  
print('x < y is',x<y)  
  
# Output: x == y is False  
print('x == y is',x==y)  
  
# Output: x != y is True  
print('x != y is',x!=y)  
  
# Output: x >= y is False  
print('x >= y is',x>=y)  
  
# Output: x <= y is True  
print('x <= y is',x<=y)

### Logical Operators in Python

x = True  
y = False  
  
print('x and y is',x and y)  
  
print('x or y is',x or y)  
  
print('not x is',not x)

### Identity operators in Python

x1 = 5  
y1 = 5  
x2 = 'Hello'  
y2 = 'Hello'  
x3 = [1,2,3]  
y3 = [1,2,3]  
  
# Output: False  
print(x1 is not y1)  
  
# Output: True  
print(x2 is y2)  
  
# Output: False  
print(x3 is y3)

### Membership operators in Python

x = 'Hello world'  
y = {1:'a',2:'b'}  
  
# Output: True  
print('H' in x)  
  
# Output: True  
print('hello' not in x)  
  
# Output: True  
print(1 in y)  
  
# Output: False  
print('a' in y)