**Iterations In Python**

An iterator is an object that contains a countable number of values.

An iterator is an object that can be iterated upon, meaning that you can traverse through all the values.

Technically, in Python, an iterator is an object which implements the iterator protocol, which consist of the methods \_\_iter\_\_()and\_\_next\_\_().

## **Iterator vs Iterable**

Lists, tuples, dictionaries, and sets are all iterable objects. They are iterable containers which you can get an iterator from.

All these objects have a iter() method which is used to get an iterator:

**Example:**

mytuple = ("apple","banana","cherry")  
myit =iter(mytuple)  
print(next(myit))  
print(next(myit))  
print(next(myit))

**Example:**

mystr ="banana"  
myit =iter(mystr)  
print(next(myit))  
print(next(myit))  
print(next(myit))  
print(next(myit))  
print(next(myit))  
print(next(myit))

## Looping Through an Iterator

We can also use a for loop to iterate through an iterable object:

mytuple = ("apple","banana","cherry")  
  
for x in mytuple:  
 print(x