Notes on Chapter 8 - Classes and Object Oriented Programming

Swarup Tripathy *

March 2022

A curated list of important points for my reference.

1. Objects are the core things that Python programs manipulate. Every object has a type that defines the kinds of things that programs can do with that object.
2. An **Abstract Data type** is a set of objects and the operations on those objects.
3. The two powerful mechanisms for managing the complexity of programming are
   - Decomposition → Creates the structure of the program
   - Abstraction → Suppresses the detail
4. One implements data abstractions using **classes**.

: is a slice syntax for every element in the array.

5. When a function definition occurs within a class definition, the defined function is called as **method** and is associated with the class. These methods are sometimes referred to as **method attributes** of the class.
6. Class supports 2 kinds of operations
   - **Instantiation** is used to create instances of the class.
     For ex., the statement s = IntSet() creates a new object of type IntSet. This object is called an Instance of IntSet.
   - **Attribute References** use dot notations to access attributes associated with the class. For ex., s.member refers to the method member associated with the instance s of type IntSet.
7. Whenever a class is instantiated, a call is made to the __init__method defined in that class.

*John V Guttag

```
s=IntSet()
s.insert(3)
print(s.member(3))
```

creates a new instance of IntSet, inserts the integer 3 into that IntSet, and then prints true.

8. When data attributes are associated with a class we call them **Class variables**. When they are associated with an instance we call them **instance variables**.

9. All instances of user-defined classes are hashable, and therefore can be used as dictionary keys.

10. What actually **Hashable** means in python? Ref: Geeks for Geeks

    - hashable is a feature of Python objects that tells if the object has a hash value or not.
    - If the object has a hash value then it can be used as a key for a dictionary or as an element in a set.
    - An object is hashable if it has a hash value that does not change during its entire lifetime.
    - Python has a built-in hash method (__hash__()) that can be compared to other objects.
    - if the hashable objects are equal then they have the same hash value.
    - All immutable built-in objects in Python are hashable like tuples while the mutable containers like lists and dictionaries are not hashable. An example below

```
t1 = (1, 5, 6)
t2 = (1, 5, 6)
# show the id of object
print(id(t1))
print(id(t2))
######## output ########
140040984150664
140040984150880
```