OOP-Python

(Object Oriented Programming)

Beginner to Advance

Python- What Is OOP

Procedural programming is about writing procedures or

methods that perform operations on the data,

while object-oriented programming is about creating objects

that contain both data and methods

Python- Classes & Object

class is a template definition of the method s and variable s in a particular kind of object

object is a specific instance of a class; it contains real values instead of variables

Example

class	objects
Fruit	Apple
	Banana
	Mango

Python-Create Class

To create a class, use the class keyword

Example

Create a class named MyClass, with a property named x:

```
class MyClass:
  x = 5
```

Python-Create Object

Now we can use the class named myClass to create objects:
Example

Example

Create an object named p1, and print the value of x:

```
p1 = MyClass()
print(p1.x)
```

The-init-()function

The examples above are classes and objects in their simplest form, and are not really useful in real life applications.

To understand the meaning of classes we have to understand the built-in __init__() function.

All classes have a function called __init__(), which is always executed when the class is being initiated.

Use the __init__() function to assign values to object properties, or other operations that are necessary to do when the object is being created

Example

Create a class named Person, use the __init__() function to assign values for name and age:

```
class Person:
  def __init__(self, name, age):
    self.name = name
    self.age = age

p1 = Person("John", 36)

print(p1.name)
print(p1.age)
```

The **__init__()** function is called automatically every time the class is being used to create a new object.

Python-Object Method

Objects can also contain methods. Methods in objects are functions that belong to the object.

Let us create a method in the Person class:

Example

Insert a function that prints a greeting, and execute it on the p1 object:

```
class Person:
  def __init__(self, name, age):
    self.name = name
    self.age = age
  def myfunc(self):
    print("Hello my name is " + self.name)
p1 = Person("John", 36)
p1.myfunc()
```

Python-Inheritance

Inheritance allows us to define a class that inherits all the methods and properties from another class.
Parent class is the class being inherited from, also called base class.

Child class is the class that inherits from another class, also called derived class.

Example

Create a class named Person,
with firstname and lastname properties, and
a printname method:

```
class Person:
  def __init__(self, fname, lname):
    self.firstname = fname
    self.lastname = lname

  def printname(self):
    print(self.firstname, self.lastname)
```

#Use the Person class to create an object, and then execute the printname method:

```
x = Person("John", "Doe")
x.printname()
```

Python-Iterators

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

Return an iterator from a tuple, and print each value:

```
mytuple = ("apple", "banana", "cherry")
myit = iter(mytuple)

print(next(myit))
print(next(myit))
print(next(myit))
```

Python Scopes

A variable is only available from inside the region it is created. This is called **scope**

Local Scope

A variable created inside a function belongs to the local scope of that function, and can only be used inside that function.

Global Scope

A variable created in the main body of the Python code is a global variable and belongs to the global scope.
Global variables are available from within any scope, global and local

Python Modules

Consider a module to be the same as a code library.
A file containing a set of functions you want to include in your application.

Python Dates

A date in Python is not a data type of its own, but we can import a module named datetime to work with dates as date objects.

Python Jason

JSON is a syntax for storing and exchanging data.
JSON is text, written with JavaScript object notation