Day 4

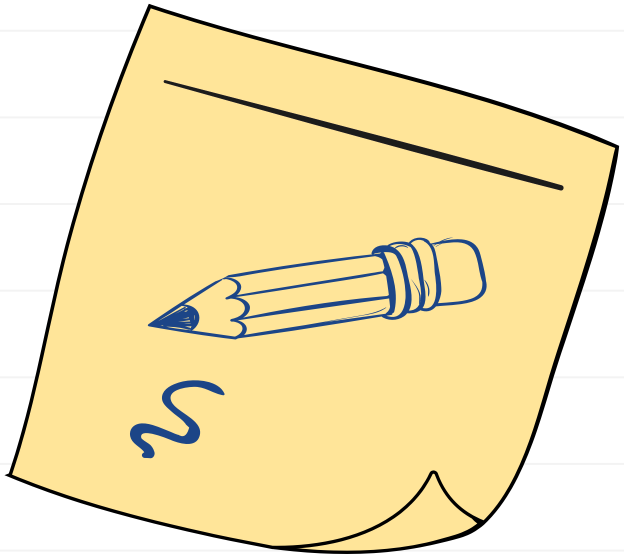
Python BootCamp

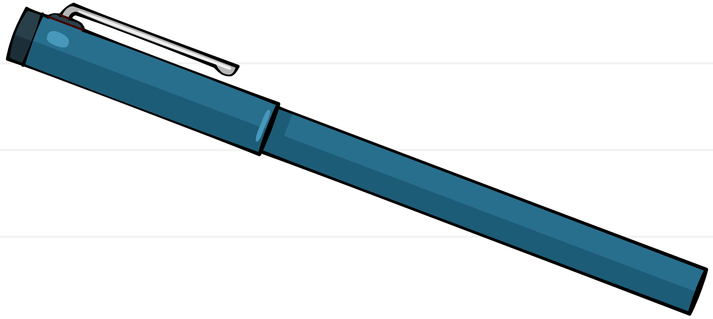
Object Oriented Programming

OOPS !!

Class :

A Class is like a "blueprint" for creating objects.





Creating a Class and an Object :

Class Definition :

```
class ABC :
    # Class Body
    name = "Python"
```

Object Definition :

```
object = ABC ( )
```

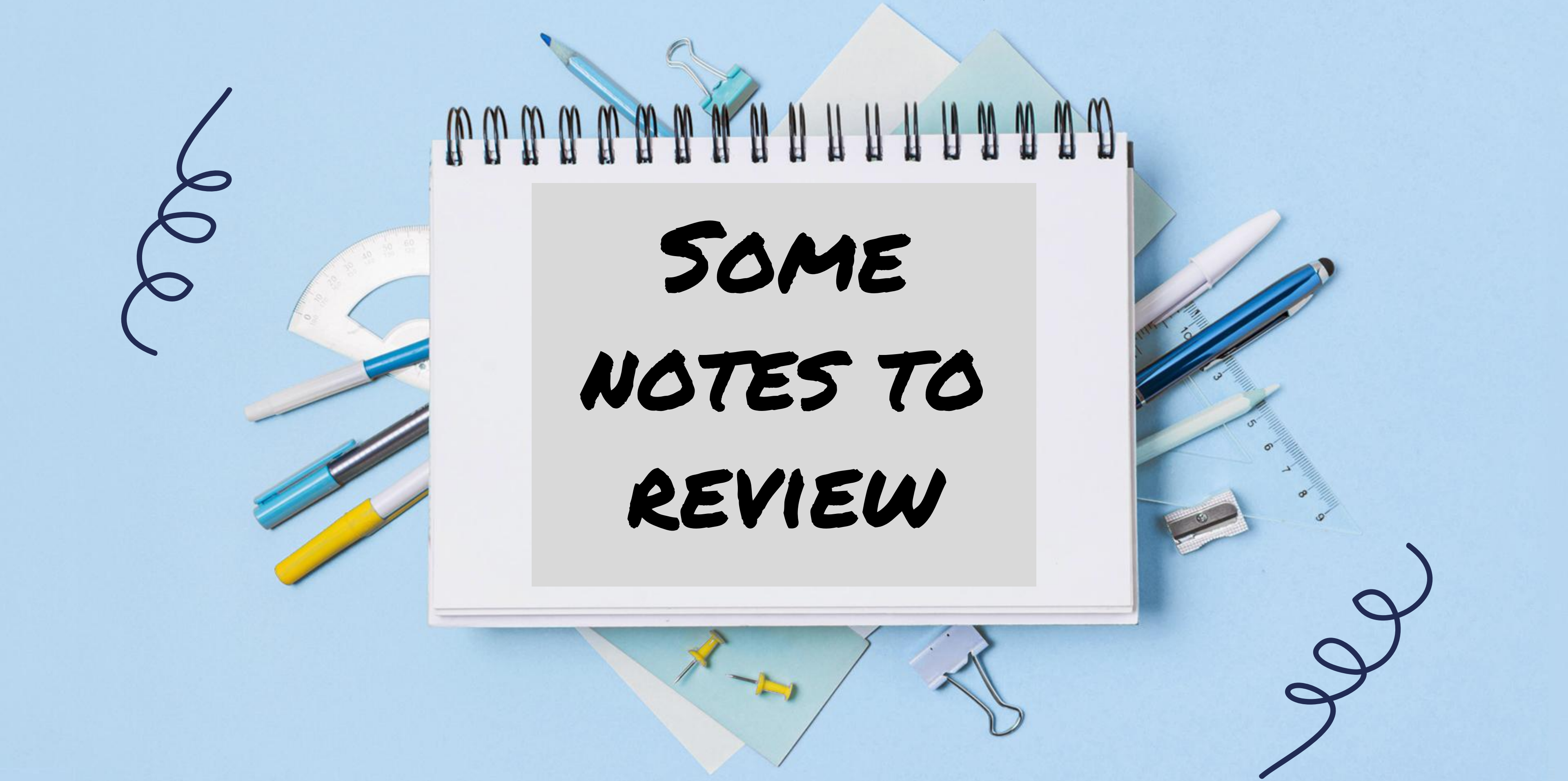
More about Classes :

- Accessing attributes
- __init__ function and self keyword
- Methods in class and data manipulation
- Deleting attributes and objects
- Pass Statement
- Inheritance - Parent and Child classes
- __init__ function in Child class
- super( ) function

More about Classes :

- Encapsulation
- Abstract and Concrete Class

Let's do some coding



Some notes to review

Accessing attributes

A class can have attributes that will be the properties of its objects. We can access the respective object attributes with the dot operator.

__init__ function and self keyword

We use init function to initialise the class with the given arguments .
The self parameter is a reference to the current instance of the class, and is used to access variables that belongs to the class.

Deleting attributes and objects

We can use the del keyword to delete the attributes of an object or the whole object too.

Pass Statement

The pass statement is use to execute a class without a body and avoiding the error.

Inheritance - Parent and Child classes

Inheritance is a way of creating a new class that uses the attributes and methods of an existing class without modifying it. The class that we are deriving from is called Base Class or Parent Class. The class that is inheriting the Parent Class is called Child Class.

__init function in Child class

The init function in child class will now initialise the class and not the init function of parent class.
If we want to initialise the parent class attributes , we might do that by calling Parent_class_name.__init__ inside init of child class.

super[ ] function

Now , when we want to initialise the parent class attributes from base class init function , we can use super().__init__ function to do that too .Super will refer to the parent class

Encapsulation

Using OOPs in Python, we can restrict access to methods and attributes. This prevents data from direct modification which is called encapsulation. In Python, we denote private attributes using underscore as the prefix i.e single _ or double __. The private, protected , public concepts in inheritance is same in python.

Abstract Class

Abstract classes are incomplete because they have abstract methods which have no body. Abstract class works as a template that we can use to build it up in other classes. An abstract class is not a concrete class hence cannot be instantiated.

Concrete Class

A concrete class is a class that has an implementation for all of its methods that were inherited from abstract or implemented via interfaces. It also does not define any abstract methods of its own.