**Object Oriented Programming**

Object-oriented programming is a programming paradigm that provides a means of structuring programs so that properties and behaviors are bundled into individual objects.

Object-oriented programming is an approach for modeling concrete, real-world things, like cars, as well as relations between things, like companies and employees, students and teachers, and so on.

OOP models real-world entities as software objects with some associated data and can perform certain functions.

**What is a Class?**

**A class is a collection of instance variables and related methods that define a particular object type.**

Class name contains no parentheses or any kind of argument

Note 1 - Unlike other programming languages, the file name does need not to match the class name

Note 2 - In Python, built-in classes are named in lowercase, but user-defined classes are named in Camel or Snake case, with the first letter capitalized.

All class definitions start with the class keyword, followed by the class’s name and a colon. Any code that is indented below the class definition is considered part of the class’s body.

# CREATING A CLASS

class Book: pass

pass is often used as a placeholder indicating where the code will eventually go. It allows you to run this code without Python throwing an error.

**\_\_init\_\_ Method**

The \_\_init\_\_ special method, also known as a Constructor, is used to initialize the Book class with attributes such as title, quantity, author, and price.

**Arguments of the init method should be the basic requirement of the class init runs automatically whenever an instance of class is created.**

You can give \_\_init\_\_() any number of parameters, but the first parameter will always be a variable called self.

When a new class instance is created, the instance is automatically passed to the self-parameter in .\_\_init\_\_() so that new attributes can be defined on the object.

class Book:

    def \_\_init\_\_(self, title, quantity, author, price):

        self.title = title

        self.quantity = quantity

        self.author = author

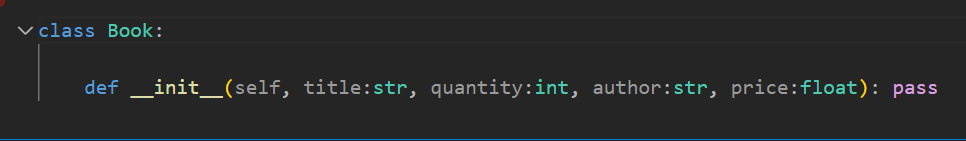
        self.price = price

These above declarations help to access these variables not in just init function but in the whole class

**Defining the datatypes of the arguments**

In a function, you can define the datatype of argument by arg: datatype

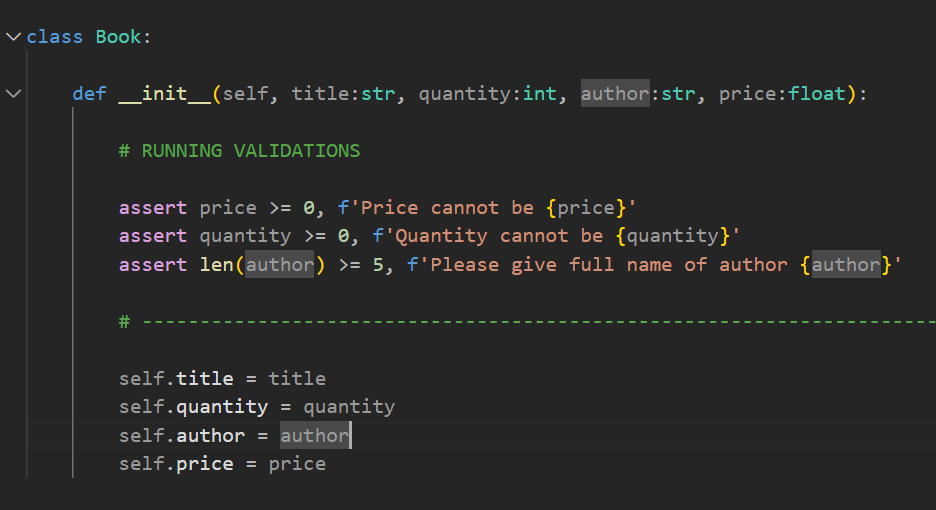
Note 3 - All the default arguments must be after non-default arguments in a function.



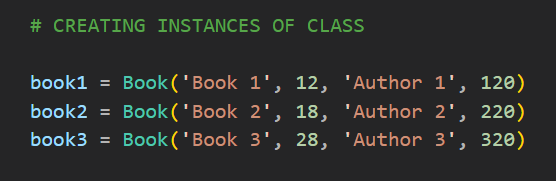
**Running Validations**

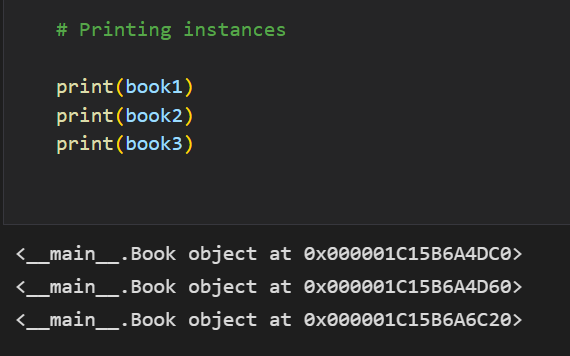
Usually, the first thing we do when the \_\_init\_\_ method is called is run validations, which means whether the arguments fulfil the required condition.

Validations are run using assert - if False show error, opposite of if statement.



Creation of instances





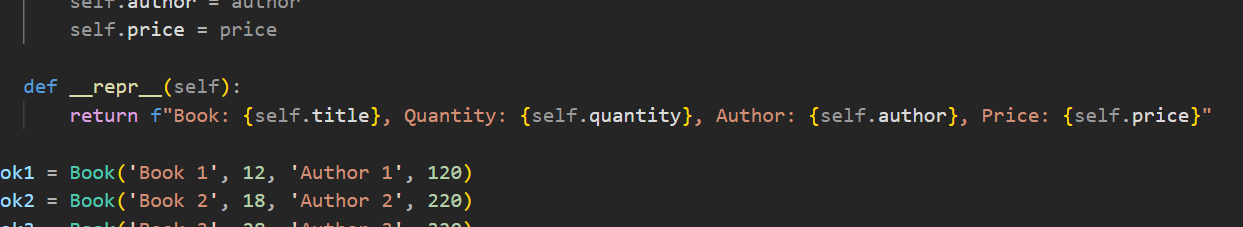
**REPR Method**

The class and memory location of the objects are printed when they are printed.

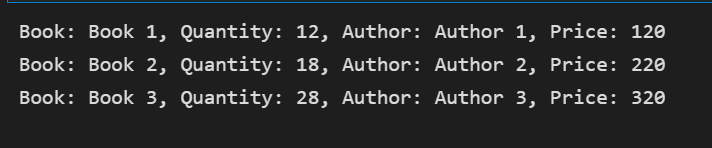
We can't expect them to provide specific information on the qualities, such as the title, author name, and so on.

But we can use a specific method called \_\_repr\_\_ to do this.

print(Item.all) # This will return the object address of instances which is not so helpful, so we can use a magic method called repr stands for representation of objects.

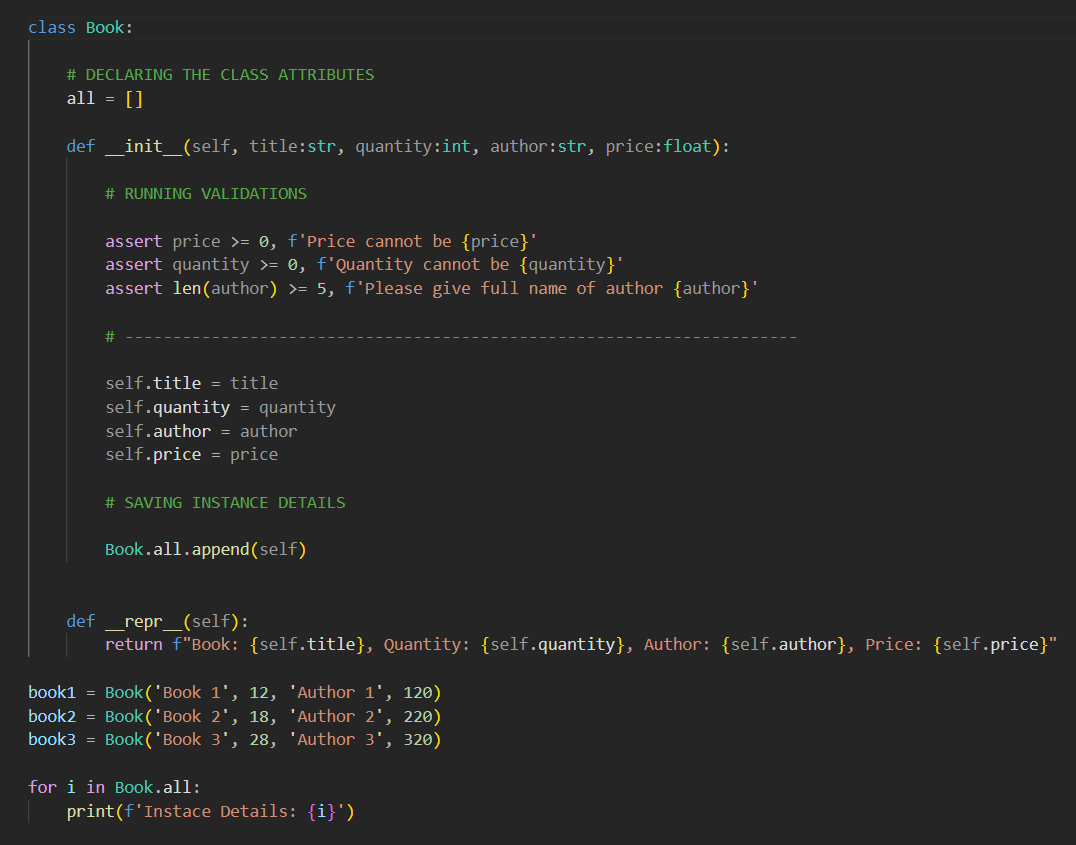


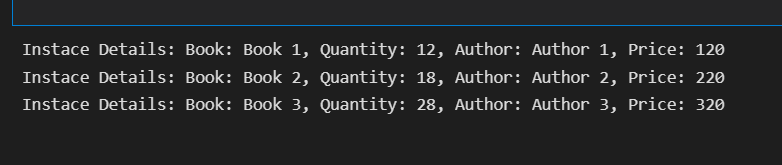
Printing the classes after using repr method



**Storing the instance details into a list**

Variables that are declared outside of any kind of function but in the class are global, they can even be used outside the class by **ClassName.variableName**





**INSTANCE ATTRIBUTES VS CLASS ATTRIBUTES**

Attributes created in .\_\_init\_\_() are called instance attributes.

An instance attribute’s value is specific to a particular instance of the class.

All Dog objects have a name and an age, but the values for the name and age attributes will vary depending on the Dog instance.

On the other hand, class attributes are attributes that have the same value for all class instances.

You can define a class attribute by assigning a value to a variable name outside of .\_\_init\_\_().

Class attributes are defined directly beneath the first line of the class name and are indented by four spaces.

They must always be assigned an initial value.

When an instance of the class is created, class attributes are automatically created and assigned to their initial values.