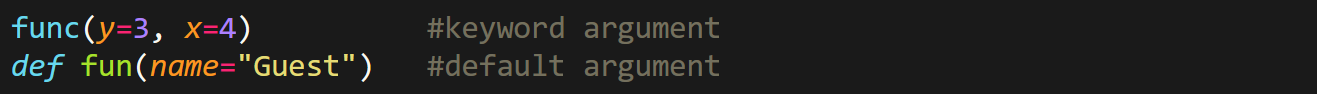
**OBJECT ORIENTED PROGRAMMING**

**Function**

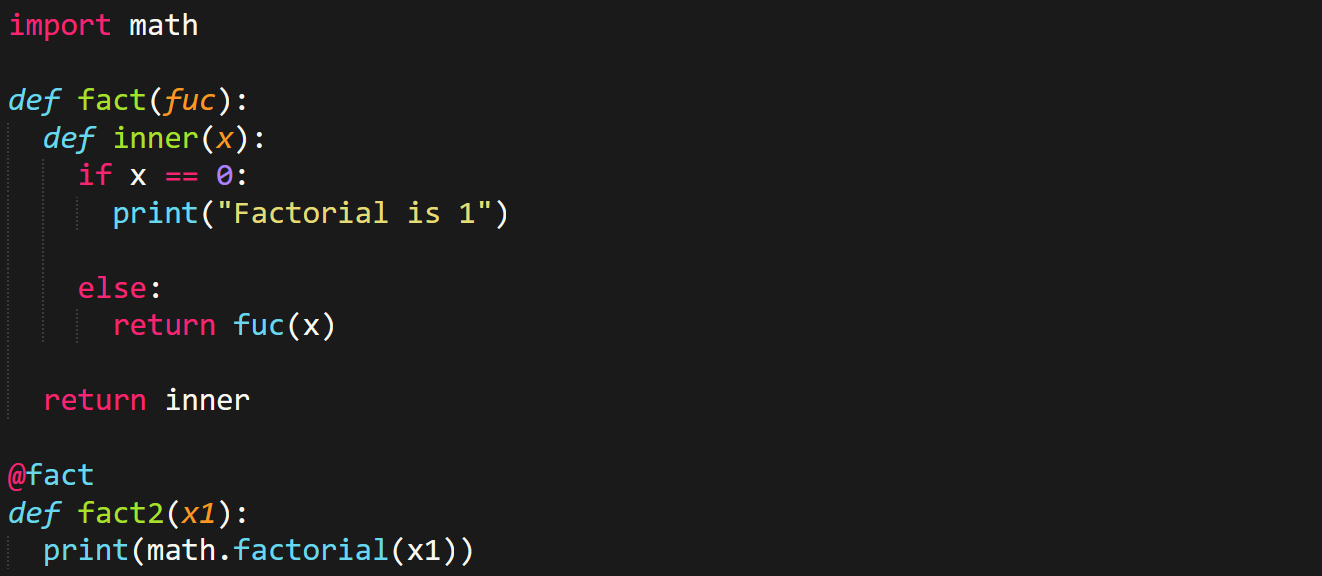
* Traditional argument passing is called **positional argument**.



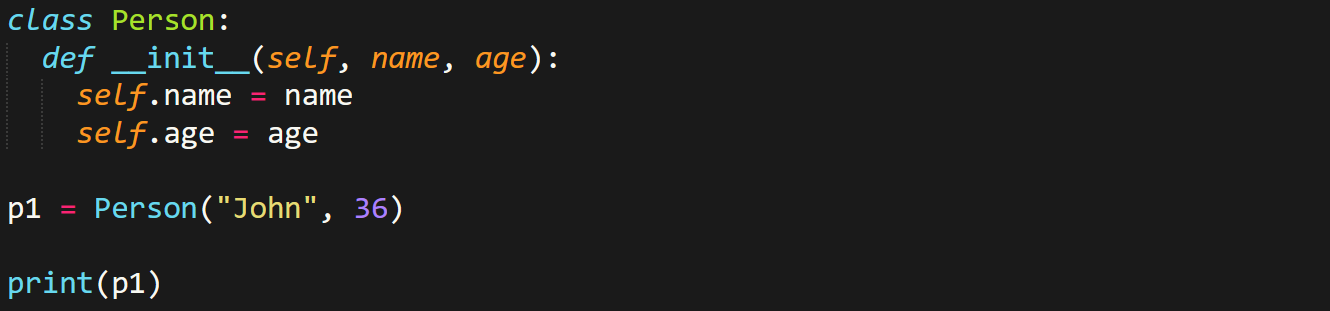
**Lambda Function**

******

**Function Decorator**

******

**Classes & Objects**



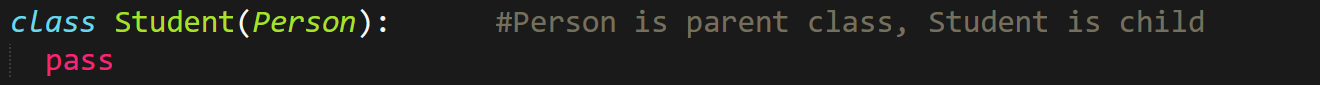
* Methods in it are as **usual functions** in Python.
* The ***\_\_init\_\_*** method is **constructor**.
* We can use any word other than ***self***, the first argument is considered ***self*** always.
* Even regular methods must contain ***self***.

Deleting object properties, or even whole ***class***:-



* Keyword ***pass*** inside block of class **doesn’t** read the class.
* ***pass*** is like ***break*** for ***class***.

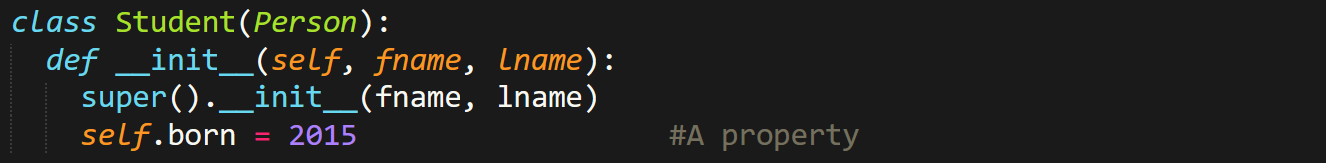
**Inheritance**



* Even the constructor is **inherited**.
* Methods of parent ***class*** can be accessed with ***super()***.



**Properties & Polymorphism**

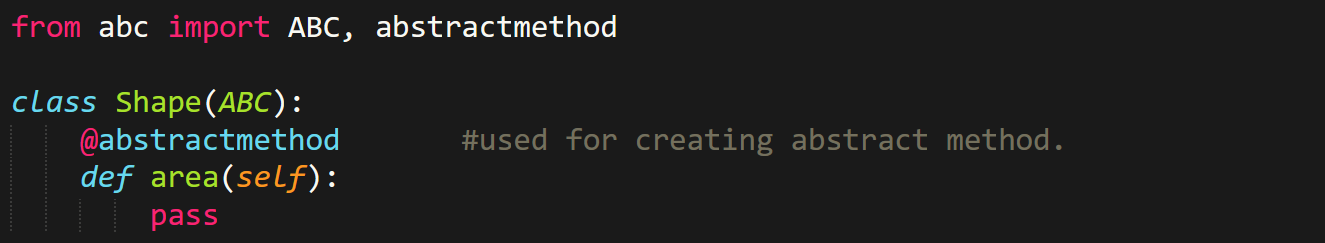


* For **overriding** (in **polymorphism**), simply just change the **body** of method in the other ***class***.

**Encapsulation**

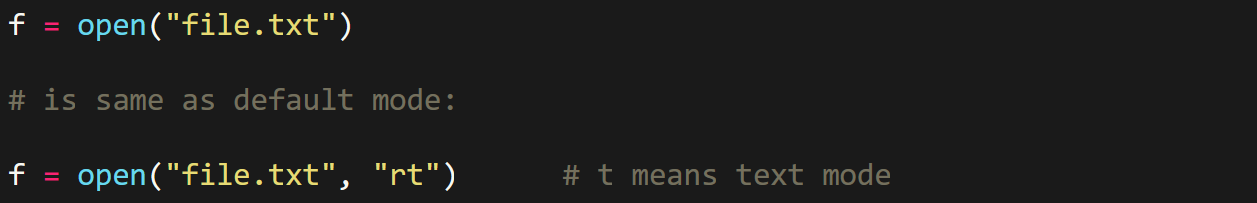
* **No** built-in mechanism for defining a **private** **variable**.
* But as per the naming convention, a **private variable** must start with ***\_***.
* For ***getter*** and ***setter*** methods, just access the variable or method from **within** the **public method**.

**Abstraction**



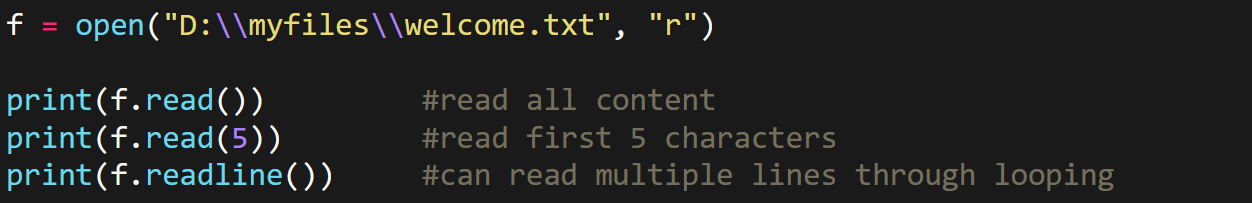
**File Handling**

Opening:-



* File can be created with **“x”**.
* But **returns error** if file already exists.

Reading:-



Writing & closing:-



Removing file:-

