Object oriented programming with Python:

|  |  |
| --- | --- |
| Class | A paradigm of programming or structure  Consists of methods/ functions and data specific to that class.  So, don’t have to repeat the same codes. Just create a new instance/ objects |
| Method | Like functions but in a class |
| Constructors | \_\_init\_\_  Runs automatically when an instance of class is created |
| Attribute |  |
| Object | An instance of class |
| Class variable | Variables that belong to the class.  A variable that is shared by all instances of a class. Class variables are defined within a class but outside any of the class's methods |
| Instance variable | Variable that belongs to individual object  When we create an object, we can pass them through.  For example:  apple = Fruit(‘apple’,2) |
| Class methods | Add decorator @classmethod  A method bound to the class not the objects, can be used to access or modify a class variable  Automatically pass in ‘cls’ as first argument |
| Static methods | Doesn’t require to pass in ‘self’(instance) or ‘cls’ (class) as first argument |
| Regular methods | Automatically pass in the instance/ object as first argument (‘self’) |
| Special methods | (Dunder/ Magic) method. Used to change behavior of built-in method. These methods allow us to emulate built-in types or implement operator overloading  Example: \_\_init\_\_, \_\_str\_\_, \_\_repr\_\_ |
| Property decorator | Allows us to define Class methods that we can access like attributes. This allows us to implement getters, setters, and deleters. |