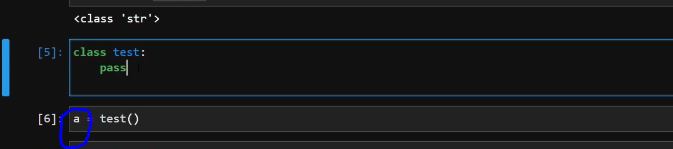
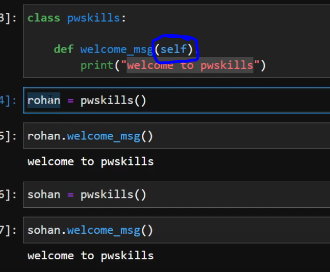
**OOPS**:

* When we create class and assign that to any variable called as OBJECT/Instance of the Class
  + **\_\_init\_\_** is the constructor, which creates inbuilt method inside Class, which takes outside data and keeps that gloabla parameter. This is the one way to get external parameter inside class.
  + Another way is by **@class\_method decorator .** This takes external input data like \_\_init\_\_
* Function created inside Class called as “**Method**”/Instance Method/Class method if created with **@class\_method decorator / Static Method**
  + Static method and Class method can directly call as class.method, no need to create object. In case of instance method, we should create object
  + Static method definition no need self.



* Any function created inside class should have value as self as pointer/reference inside(). This tells that function belongs to that class

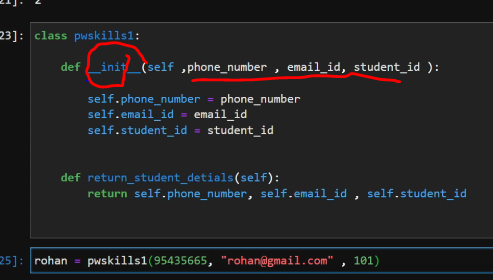


* By creating object to any class, we can call all the function inside class

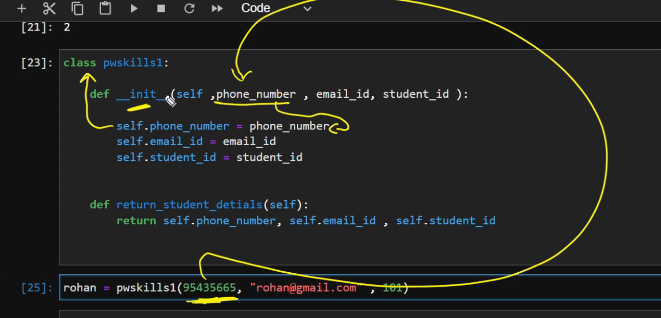
1. **Constructor**:

Passing value inside any class and get the output is done by **Constructor**.

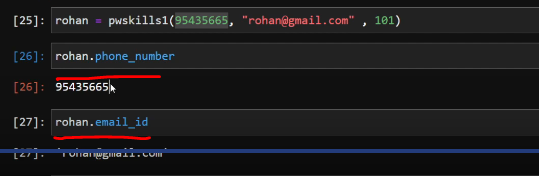
* + **\_\_init\_\_** is the constructor, which creates inbuilt method inside Class, which takes outside data and keeps that as global value inside method

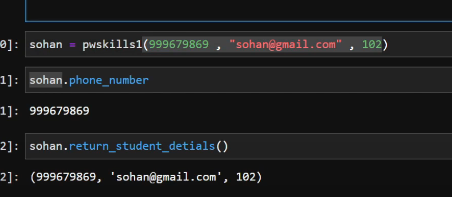


* To access this value inside Class , we need to use **self.phone\_numbe**r etc

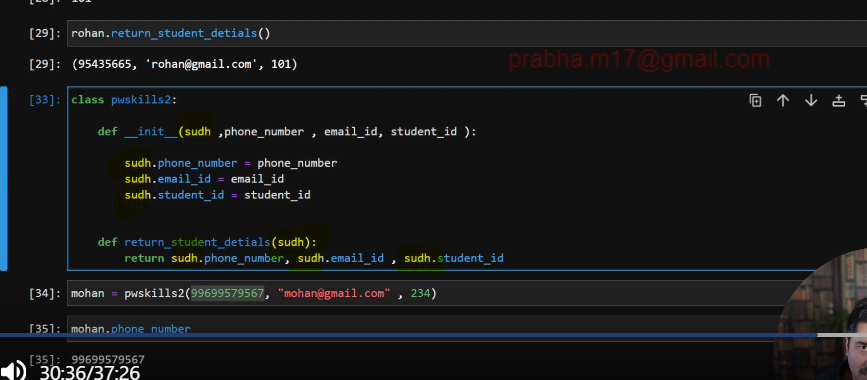


* To access this value outside Class. Calling this value is done by creating

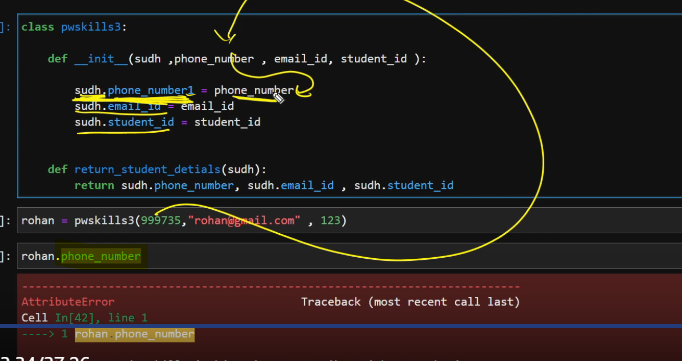




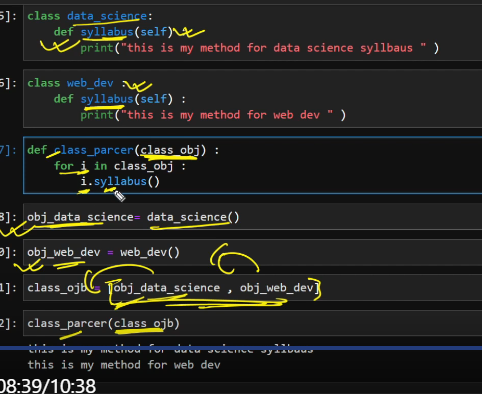
* Using self is not mandatory keyword, we can use other word also.



* We should not call with assign variable, always call with variable associated with self.



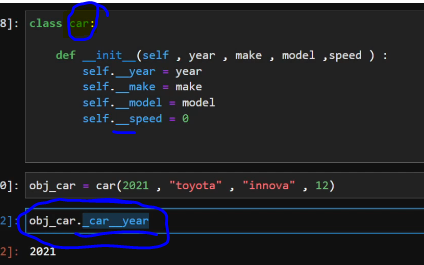
1. **Polymorphism:**
   * Func behaviour will change based on different circumstances



1. **Encapsulation and Private variable:**

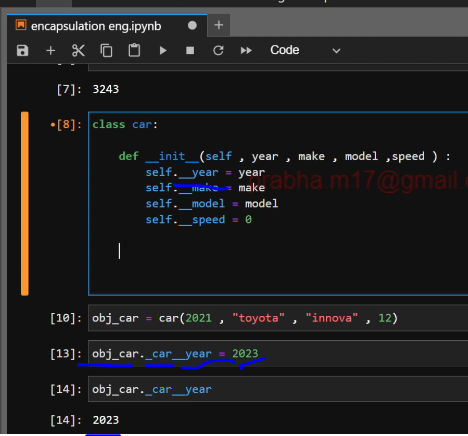
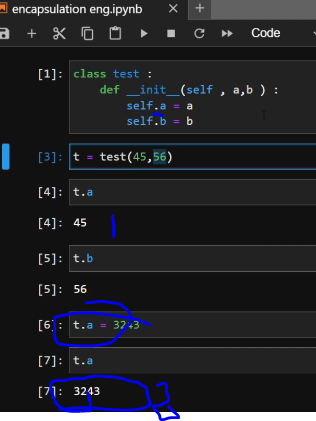
This allows to restrict any update or modification or some access

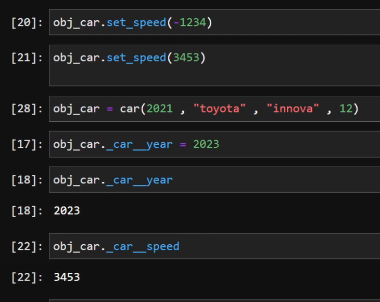
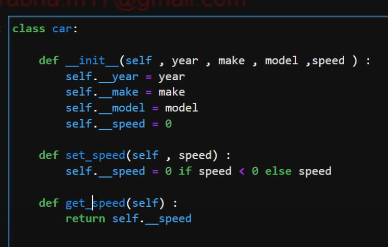
If any variable name starts with **\_\_** means , it’s a private variable. We cant see that variable in outside Class . To access that , we have to call classname **with \_** and then private variable name

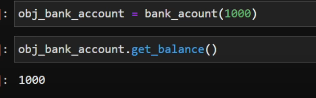
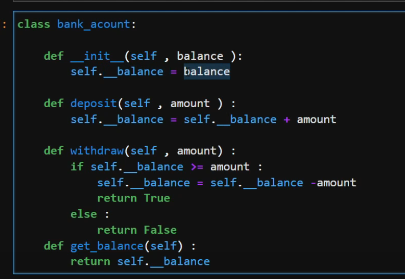


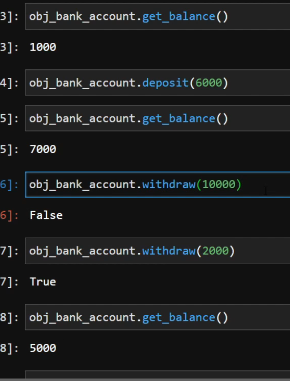
Why this encapsulation came? Earlier normal class variable anyone could able to assign new value. Now this way, we need to give name with class and private varaible, then only can access and update.

Earlier: Now:



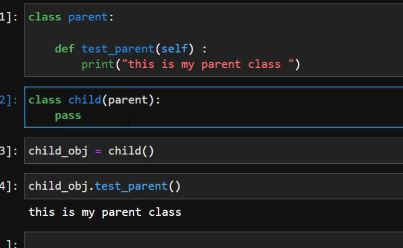




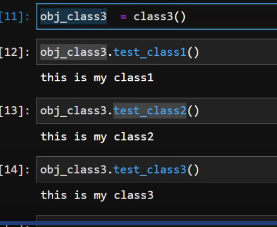
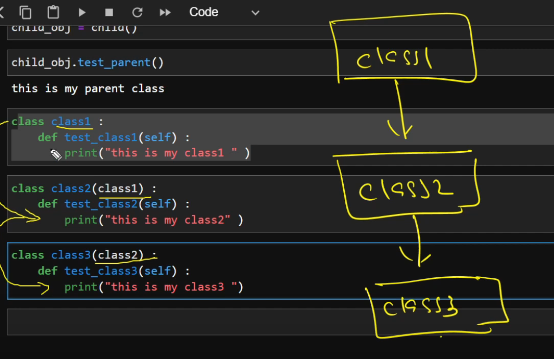


1. **Inheritance:**

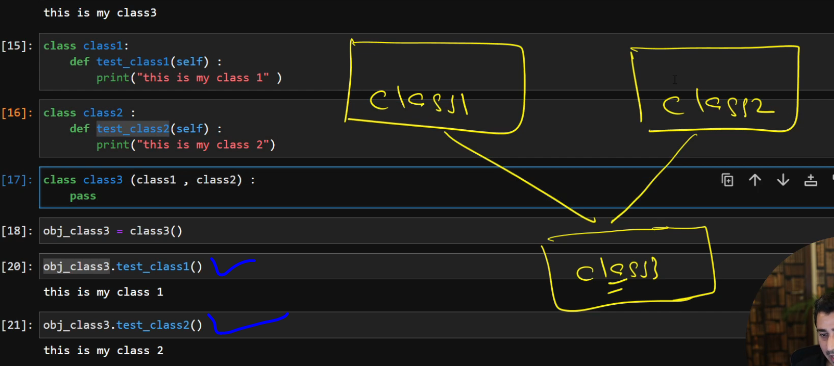
Here child class inherent properties of Parent class



**Multi Label Inheritance:**



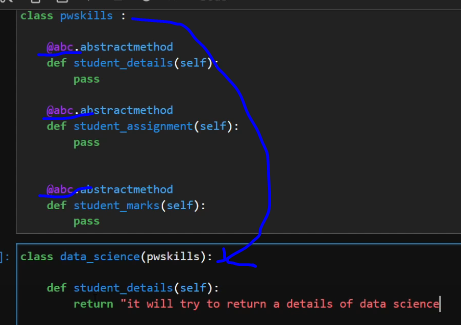
**Multiple Inheritance:**

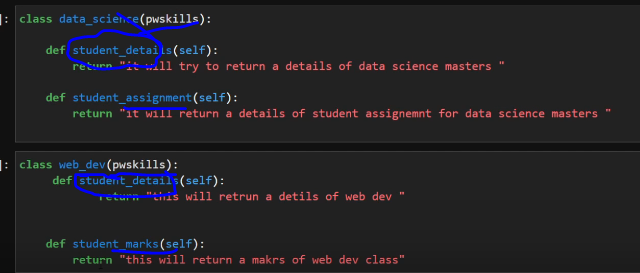


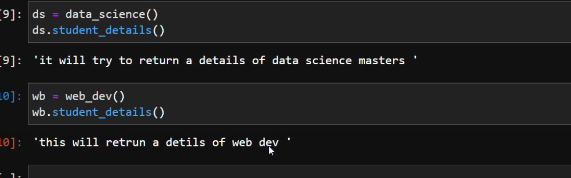
1. **Abstraction**

Its like a just blueprint. By defining any function with **@** before def /Method/Function will make that function as abstract function

Whatever method created as abstracted method can be called inside child class, main class behaves as a skeleton. From that we can define similar method inside child

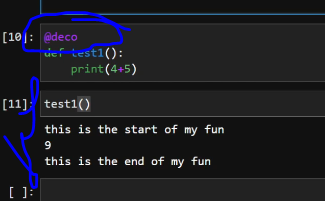
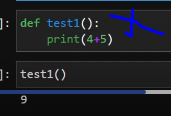
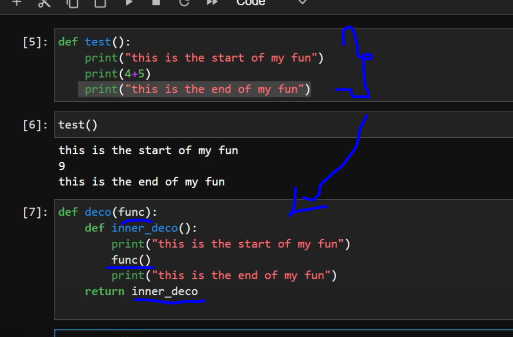






1. **Decorators**

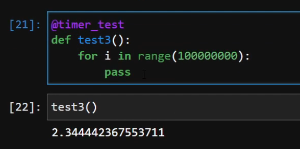
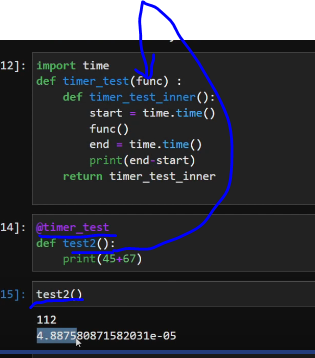
Some task/print we need to do each and very time whenever we call any function, that we can achieve by Decorator.



By calling that decoration function wit**h @ (@deco)** by its name before other function, it will do the task.

Here test1 function will be passed inside decoration function deco and it executed in same order and returned as per deco function

**Ex**: Adding starting time and end time of any function with total time taken can be achieved by this decor way

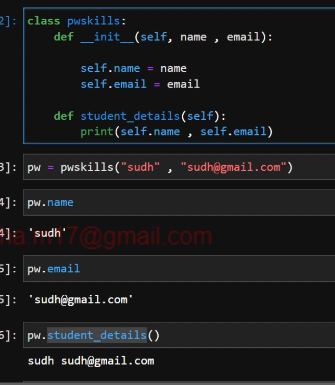


1. **Class Method:**

Inside Class, method/function which are created with parameter cls not **self** are called as **Class method.**

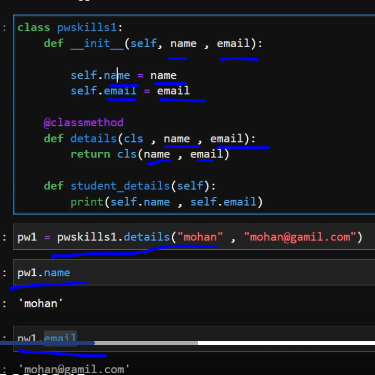
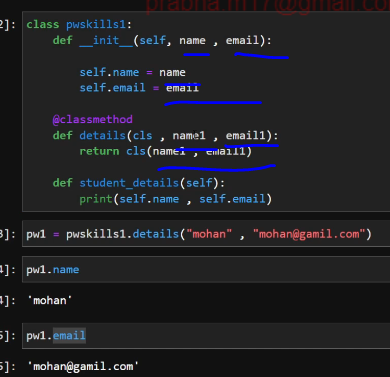
While creating any clss object, we can create by using this Class method also.

Normally \_\_init\_\_ method used to pass value to class.



But there is another alternative method to achieve this by using **cls** not **self** inside function with word **@classmethod**

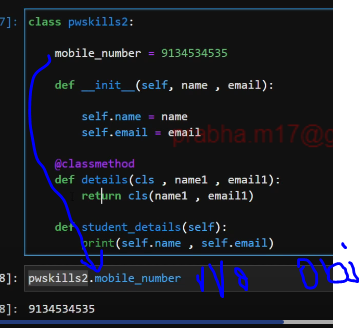
This way it assigns external value to \_init\_ functions’s variable also and to @classmethods functions variable also . if variable name not same as \_\_init\_\_variable, still it assigns.

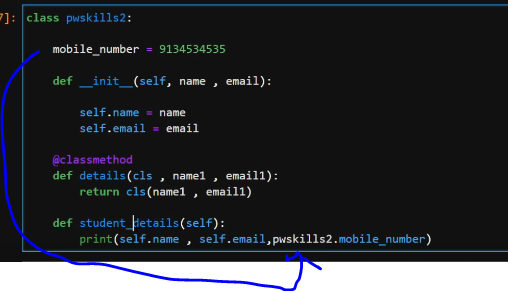
 

Defined mobile no as variable inside class, how we can access within other methods of that class or outside class.

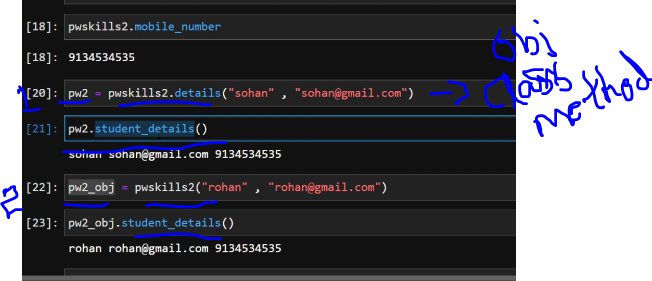
To call outside class, no need to create object from class, directly we can call as **class.variable**

To call inside class’s other method, we need to call that variable as **class.variable**

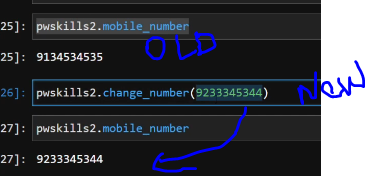
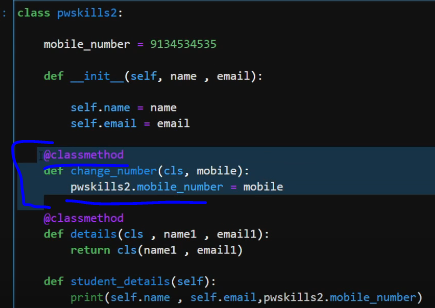




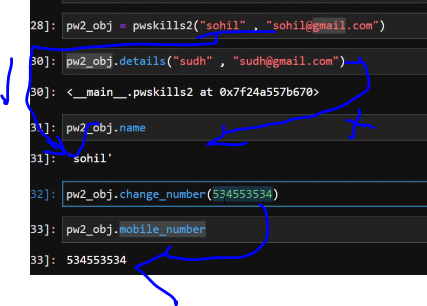
While creating any clss object, we can create by using this Class method also.



By using Class method we can change the class variable value also. Here mobile no



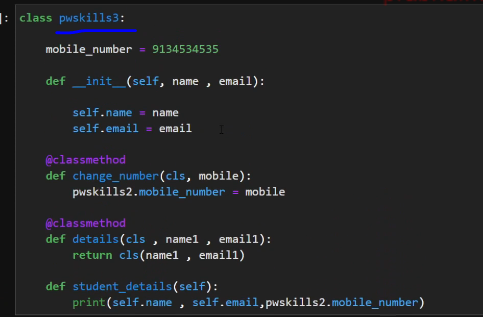
Below while creating object we passed name as **sohil**, while calling class method from that object we passed name as **sudh**. But when we execute for value, it shows old value **sohil** only



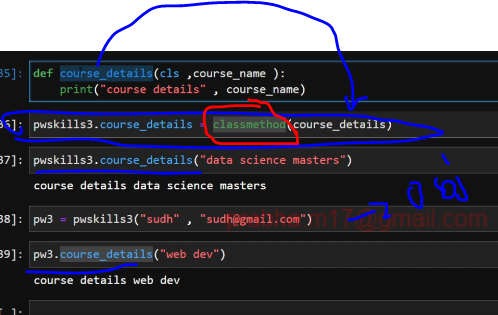
1. **How to link external function to any class:**

Some time we created Class earlier, but still we want to add some extra function to that calss as **class method,** that we can link or add by using c**lass\_method()** function

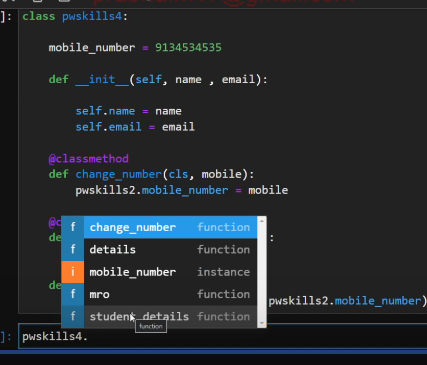
**Class and External Func:**



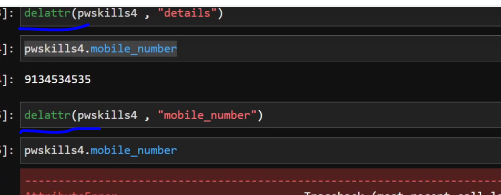
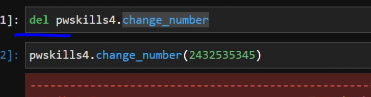
**Link External Func to Class and access it:**

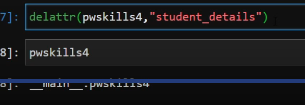


1. **How to delete Class method /Instance method from class:**



**Way1: Way 2:**



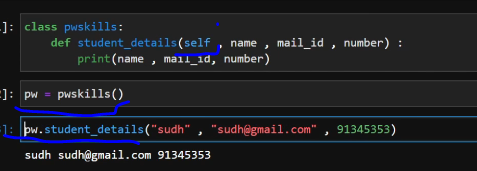


1. **Static Method:**

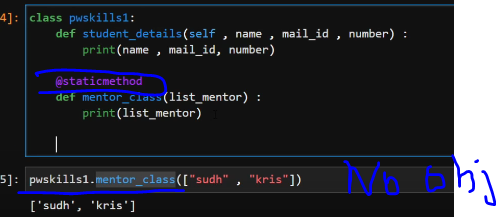
Some calss function/method , should keep static value whenever we create multiple object from that class.

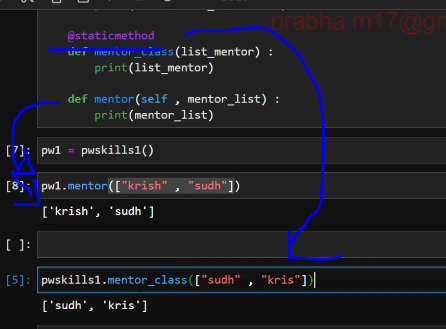
* + Static method and Class method can directly call as **class.method,** no need to create object. In case of instance method, we should create object
  + Static method definition no need self.
  + This used to reduce memory

**Normal instance Method:**

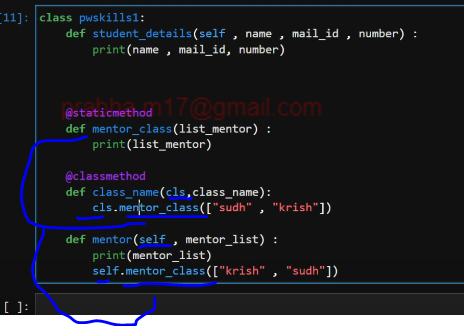


**Static method:**

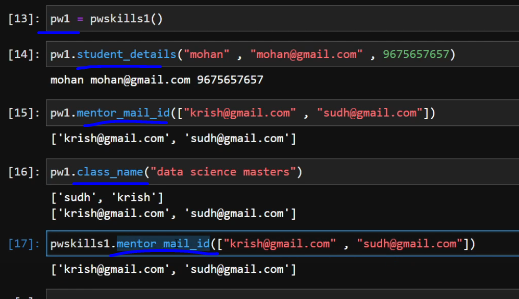
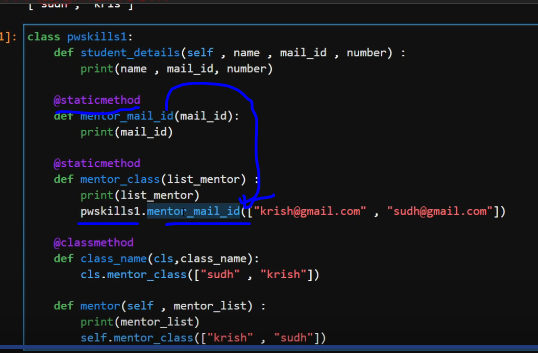




Static method’s /function can be used callled inside other instance method/ Class method also

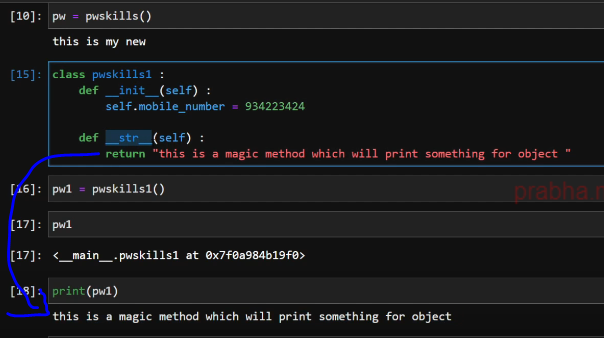
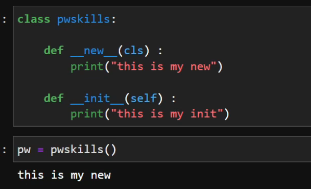
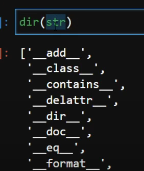
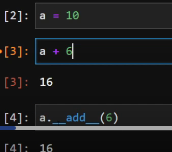


Static method’s /function can be used callled inside other static method also

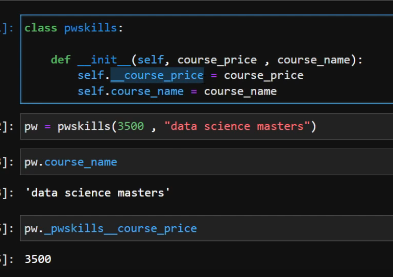


1. **Special (Magic Dunder) Methods**

Dir(int) and dir(str) gives all magic methods/functions

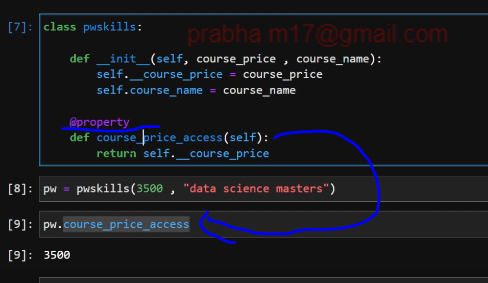


1. **Property Decorators Getters, Setters, And Deletes**

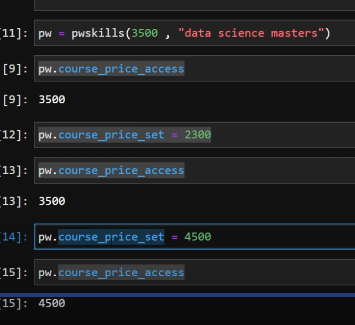
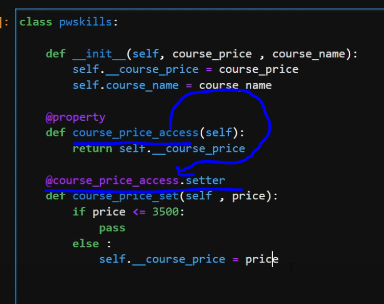
(

Normally private variable cant accessed by public, except the creator who knows how to call as **\_class.private** variable

To make this private to accessible to public id done by **property decorator - @property.** This wont allow others to modify this private value, only they can see



To allow **private variable** modification- update by outsiders is done via **decorator setter**



To allow **private variable** value delete by outsiders is done via **decorator deleter**

