***Object Oriented Programming – Python***

*So, here we will be learning about Object Oriented Programming (OOPs) in Python and know how to deal with those. We have a lot of Concepts to Cover like,*

*Classes, Objects, Encapsulation, Polymorphism, Abstraction and a lot more …. So, let’s deep dive into it.*

***Object***

Now, Objects are a way of defining things we work with. So, Everything we work with in Python or another language or maybe even in real world everything is an Object.

***Class***

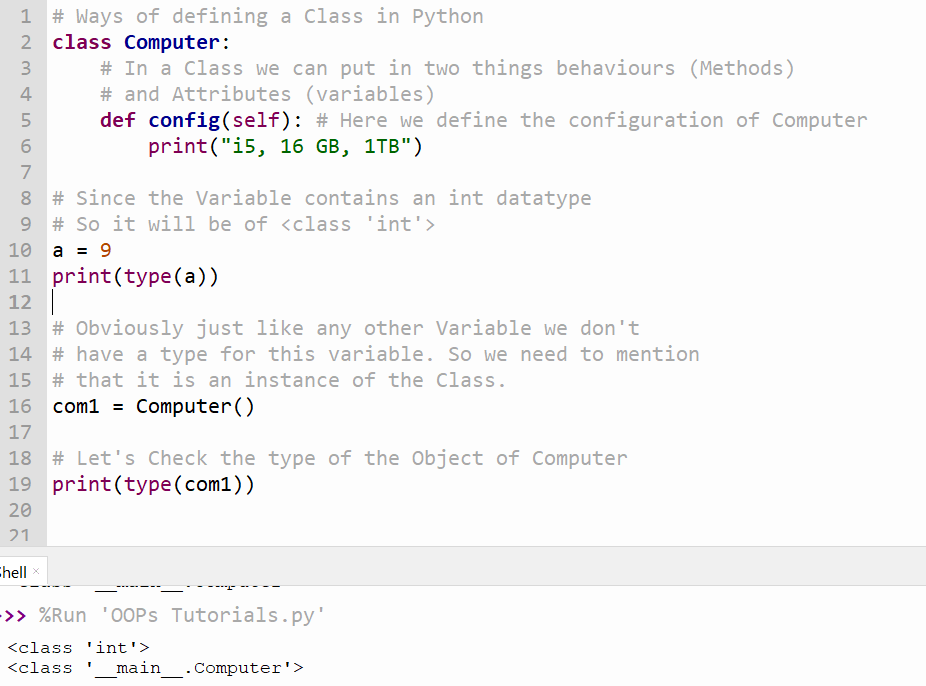
Classes are a way of categorizing objects into different forms. In simple words Classes are a blueprint of what we do and Objects define that. Let’s take an Analogy if we have a Samsung or iPhone obviously of any particular model and that’s obviously not the only model you have. There are millions of thousands of units of that phone so all of those are Objects and they are of the Class Phone. We can also say that Objects are an instance of a particular Class.

*Now let’s begin with Programming.*

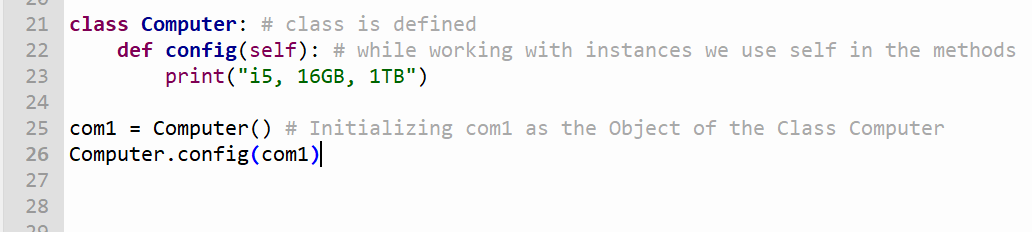
Now to define a class in Python we use the keyword Class. Let’s say we want to work with computer, now obviously we don’t have any computer data type so we create a class of our own for that and make an instance or object of that class which will help us retrieve information from that class.

*Note: Functions inside a Class are called Methods.*

Let’s See how we play around with the Code.



Now, let’s see how we call a method from a Class. We simply first create an instance or Object of the class and then we mention the Class and finally pass the Object into the Called method. Let’s See how that works.

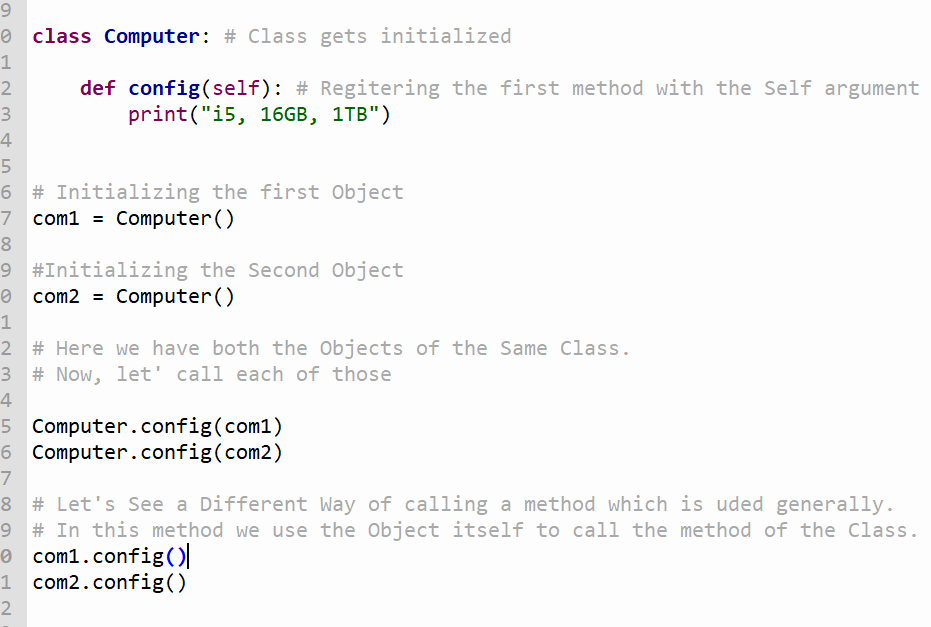


The Output as we Predict is,



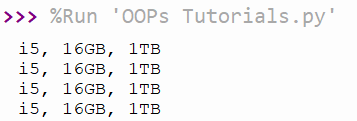
Technically when we work with classes then inside the methods we pass self as the argument of that particular method.

Now, let’s see if we call the Class with a different object and with other way. How to do it?

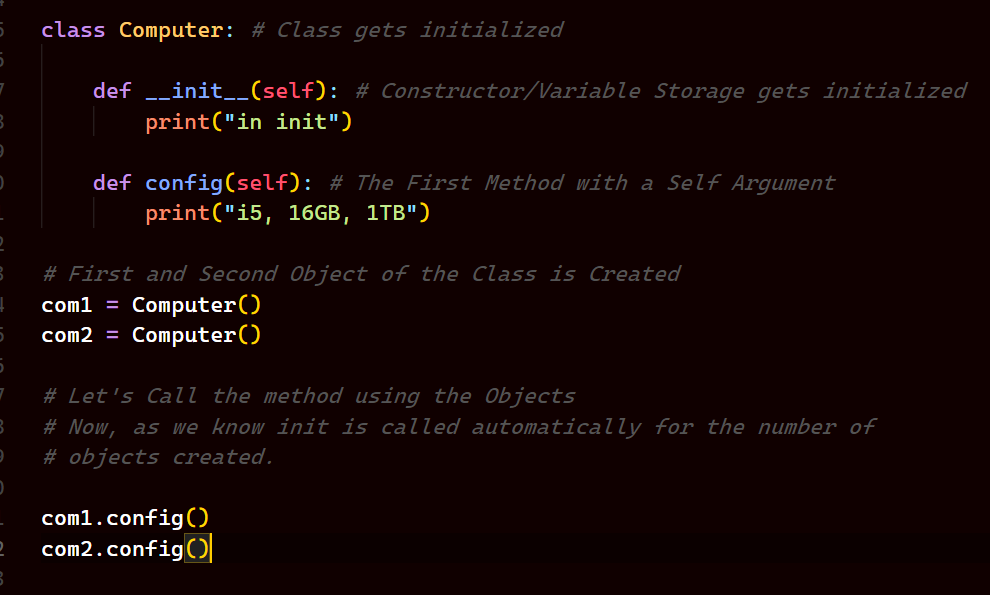


*Here we call the method by the Object itself. Behind the Scene what is happening is as we have a self argument for the method we have in a class. So, the Object we use before the ‘ . ‘ gets replaced by Self.*

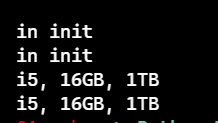
*The Output Comes out to be,*

**

*Now, let’s talk of Variables with Methods in a Class. To do with that we have a question, where to define the variables? For that we have special method with underscores. The method is “ \_\_init\_\_”. It is use to initialise Variables it can be treated as a Constructor also. Also, the idea behind “\_\_init\_\_” is that it will called automatically every time we have a Object/instance of the Class. Let’s see how we do that. We will also se how to pass variables there.*

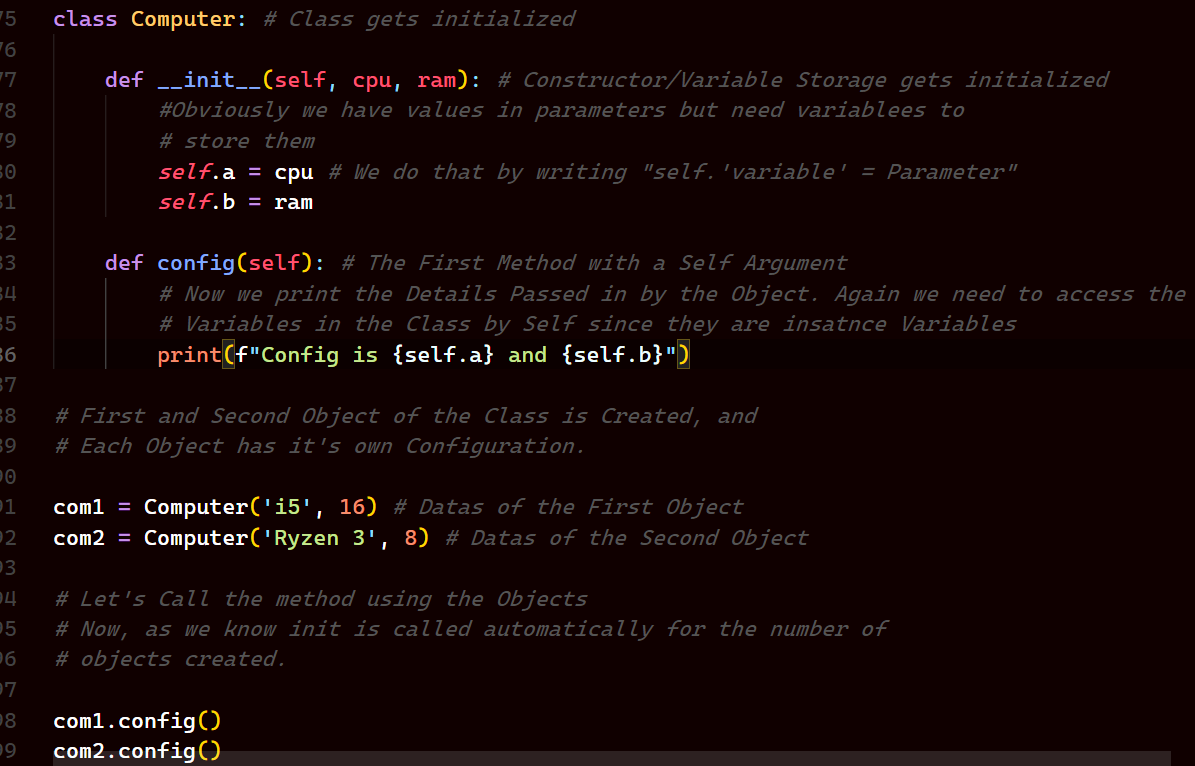
**

*Now, the Output will be let’s see,*

**

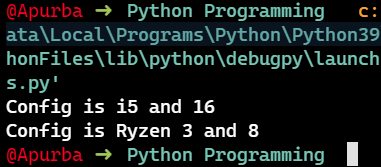
*Now, we have seen how “\_\_init\_\_” works. Let’s See how to pass the Variables and how that works.*

*So, as we know that “\_\_init\_\_” is used to store variables of values passed on to the Class. Now, while doing that what we do is we call the method of the class with the help of the Object and while we create the Object we pass the Values in the “\_\_init\_\_”. But underlying, while we call the method with the help of the Object what happens is that actually n+1 number of arguments are passed where n is the number of values you have given in the Object and one extra is of the Self in the Constructor which takes the Object as a Value. Let’s see the Code in action!!*

**

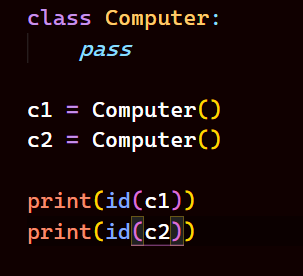
*Now, every Object has it’s own data so the Output will be different for different Objects passing it’s data into the Constructor.*

***Output***

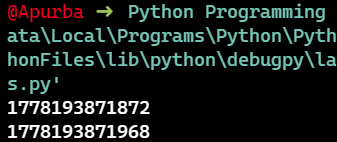
**

*Now, let’s explore the* ***Constructor*** *and* ***Self*** *more.*

*In your Computer there is a Concept of Heap Memory. Which stores the Data. So, every time you create an Object it gets allocated a Different Memory Location. So, the Point being every time you create an Object it will get allocated to different Spaces.*

**

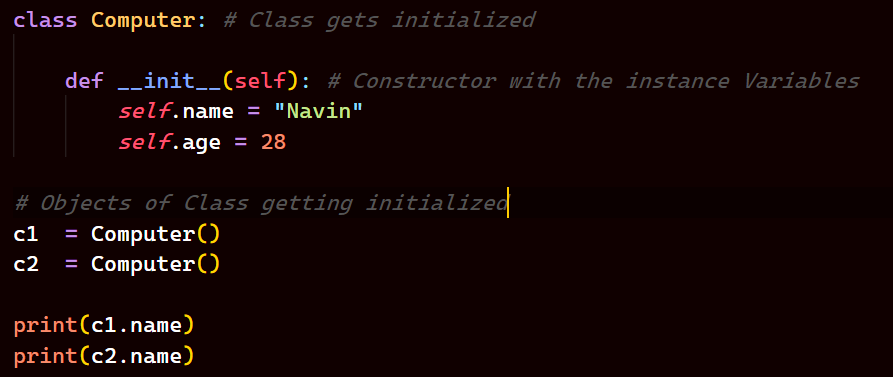
***Output***

**

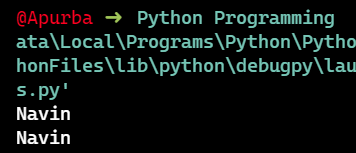
*Now, the Question is how much space or memory do these uses?, Who decides this?*

1. *Size of an Object is dependent on number of Variables and Size of each Variables.*

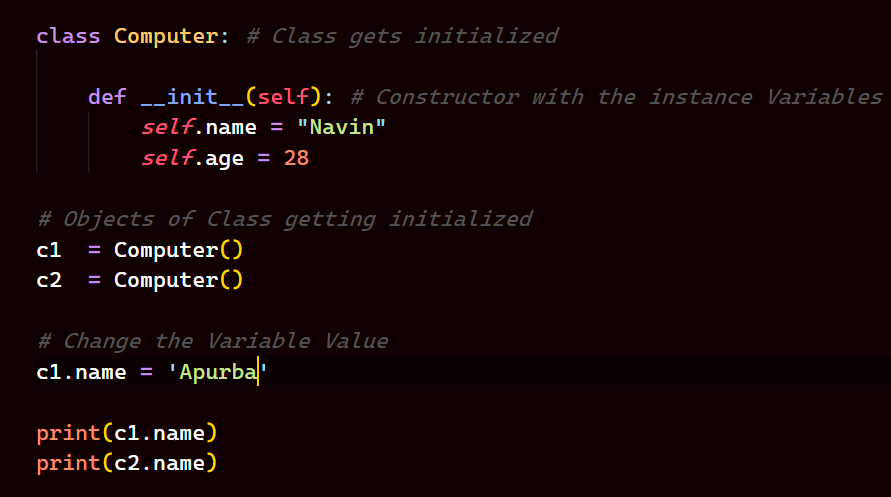
*And the Constructor decided the amount of memory to be allocated. Let’s see an Example*

**

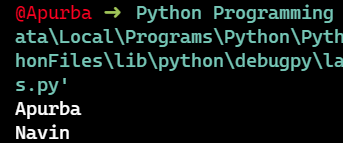
***Output***

**

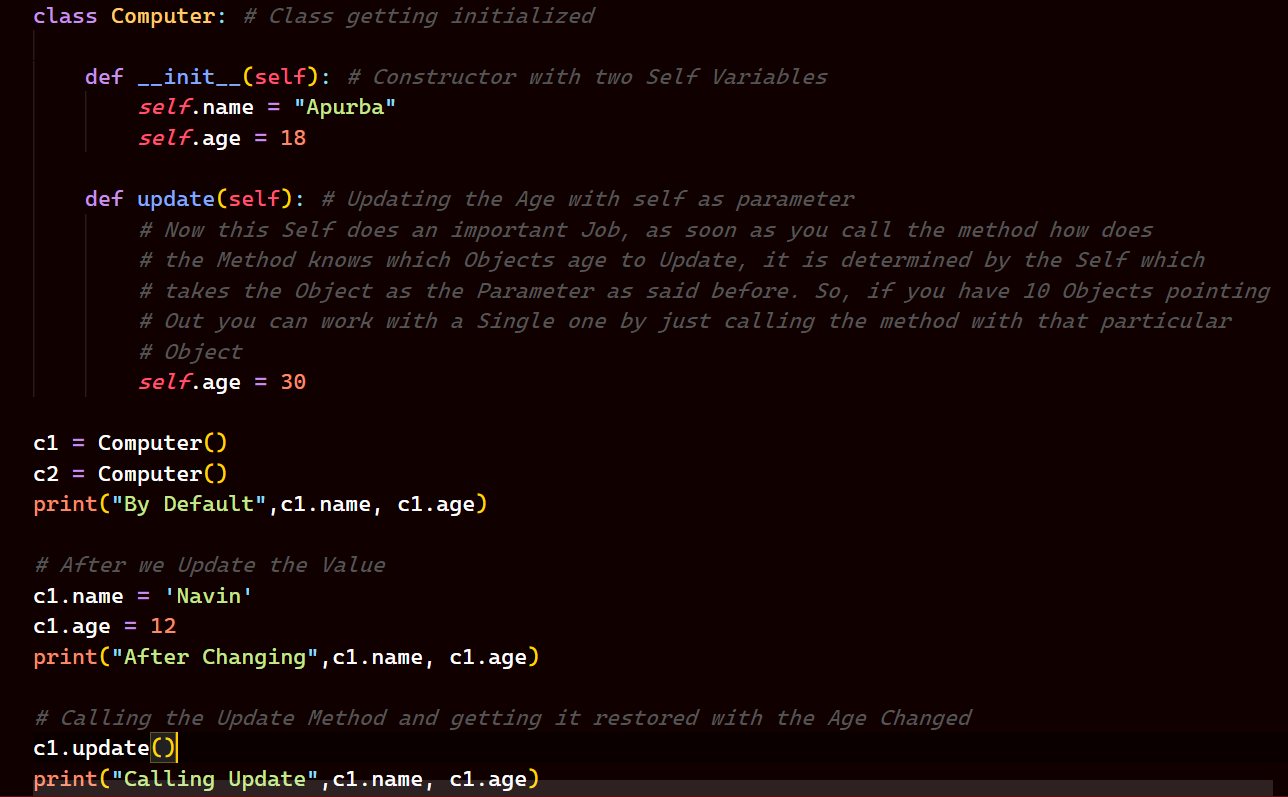
*Now, if we wanna change the vale of the variable outside the class we do that by,*

**

***Output***

**

*Now, let’s see why this* ***Self*** *is necessary. Let’s make another method in the Class and it will update the age.*

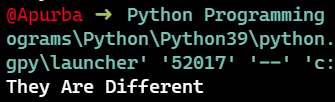
**

*As you can read above the Comments in the Program has explained it all.*

*Now, what if I want to compare the age of two Objects, how to do that? Since c1 and c2 contains the Memory Address, how to compare a particular property of that Objects. We do that by declaring a method which we will have to define. Let’s See how to do it.*



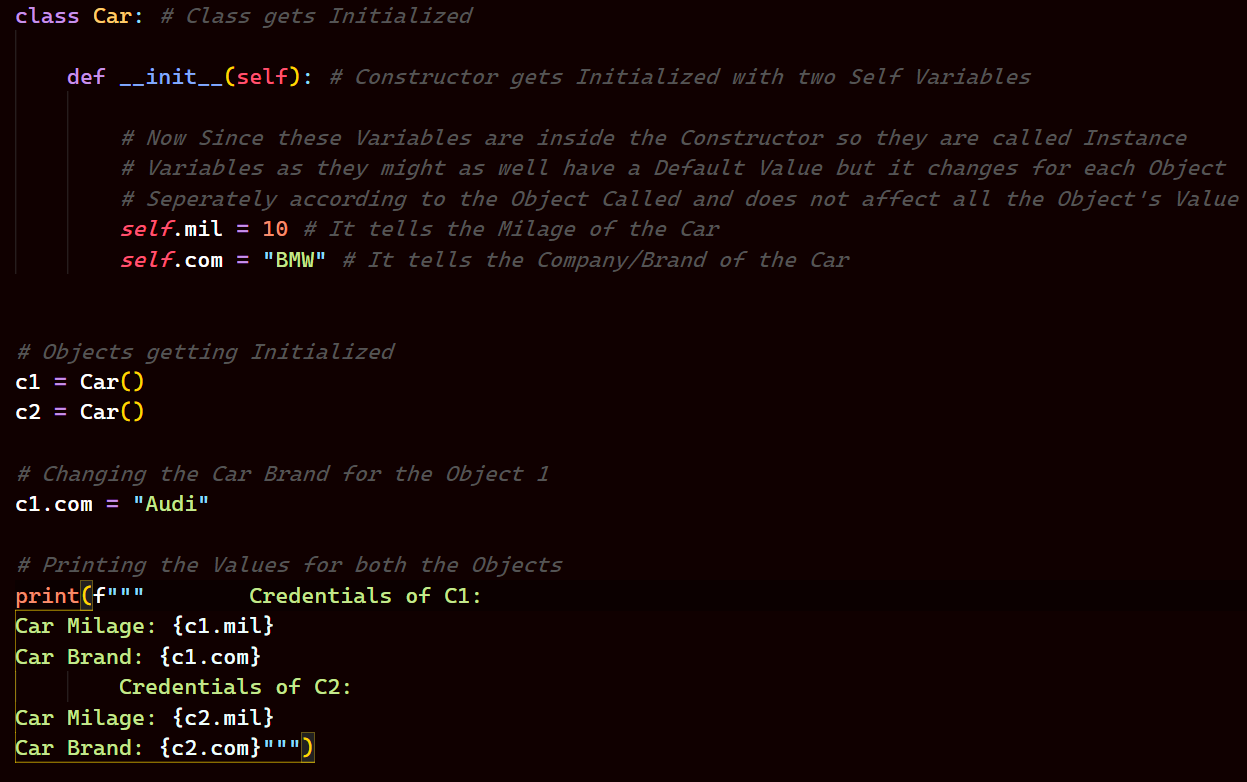
***Output***

******

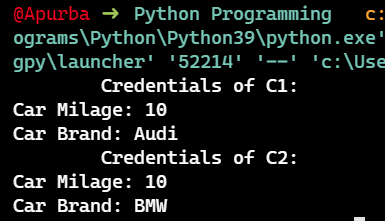
*So, we can Compare two Objects by defining our Own Methods but yes it will take two Parameters one is* ***Self*** *which gets replaced by the* ***Object who is Calling the method*** *and the* ***Other*** *is a* ***formal argument*** *which gets* ***replaced by the Object*** *which is passed.*

***Let’s look more at Variables***

*So, when we talk about variables in a Class we have two types of Variables, One is* ***Instance Variable*** *and Other is* ***Static(Class) Variable.*** *Now, to define a Variable in a Class we do that in a “\_\_init\_\_” method and those are Called Instance Variable, as by default they have some values but they can be changed with each Object as we Know Variables inside a Special Method can be Changed as they are different for each Object. So, changing them for one Object does not affect the Others. Let’s look at an Example,*

**

***Output***

******

*But what if we wanted a Variable that’s same for every Object. That’s when we use Class Variable.*

***Concept:***

*See, when we work with Variables we have two types of NameSpace in the Program.*

*What is a NameSpace?*

* *So, a NameSpace is an area where you create and store Object/Variable.*

*The two types of NameSpaces are:*

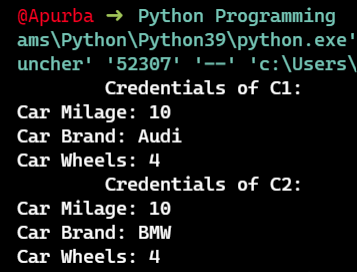
1. *Class NameSpace*
2. *Object/Instance NameSpace*

*Now, let’s say we have 4 wheels till now and that’s same for all the Object as it’s defined not in Special Method but the Class itself and we will check that by printing the Values for all the Objects Seperately. So, if we want to suddenly change the number of Wheels to 5 and want to change it for all. Let’s see how to do it and then we will check if it’s updated.*

*Let’s See a Class Variable of same Value for all Different Objects.*

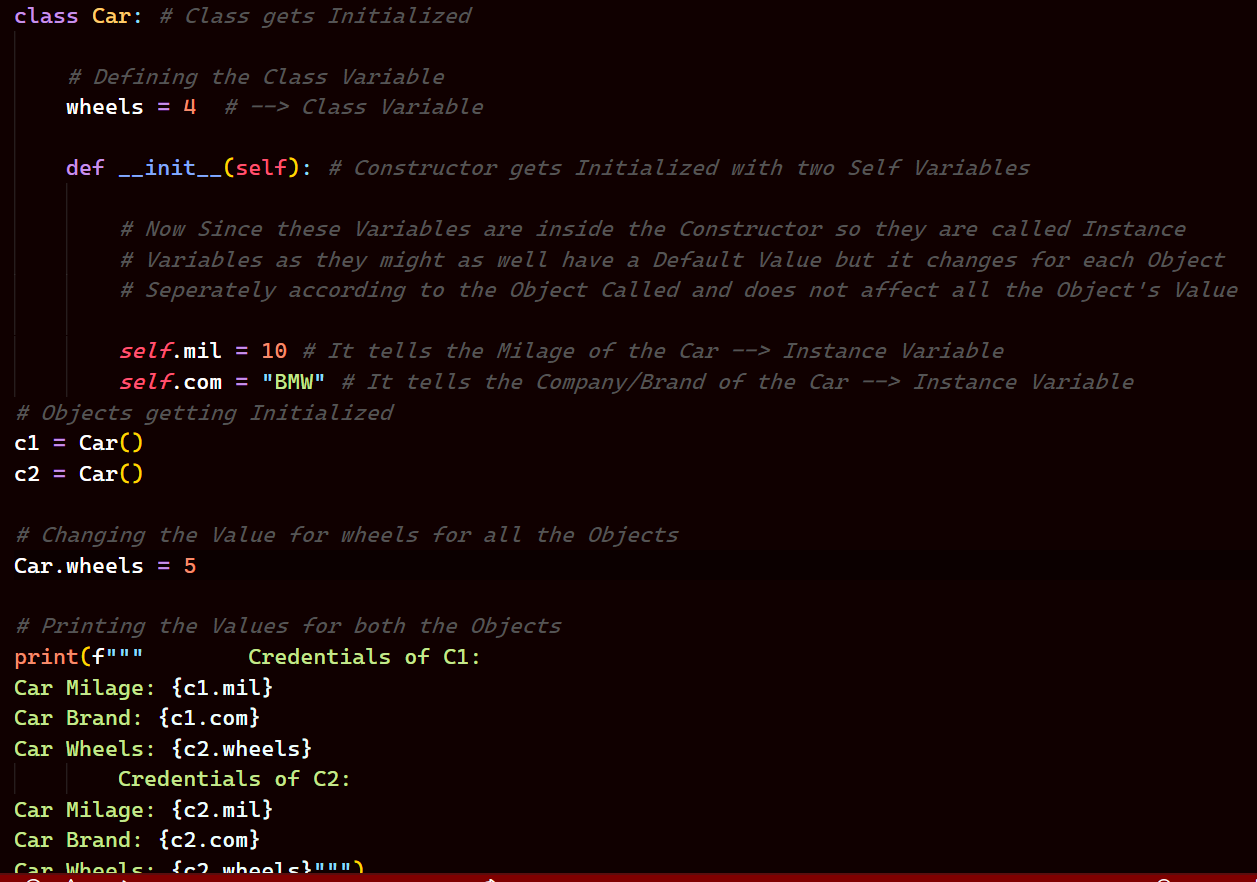
**

***Output:***

**

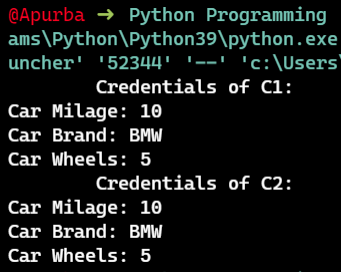
*Note that in both the Objects the Value of the number of Wheels are Same.*

*Now, what if we want to change the Value of wheels for all the Objects, for that we need to change the Class Variable and then let’s see if it changes really.*

**

*Note that the value of wheels is changed to 5 from 4 and we do that by defining the <Class name>.<variable name> = value.*

*Let’s See the Output and Verify if the Value of Wheels has changed for all the Objects.*

**

*Note that the Value of wheels has changed for both the Objects as we change the value of the Class Variable.*

***Note: Class Variables are also Called Static Variables.***

***Types of Methods***

*Now, when we talk of methods we have three types of Methods. They are:*

1. *Instance Method*
2. *Class Method*
3. *Static Method*

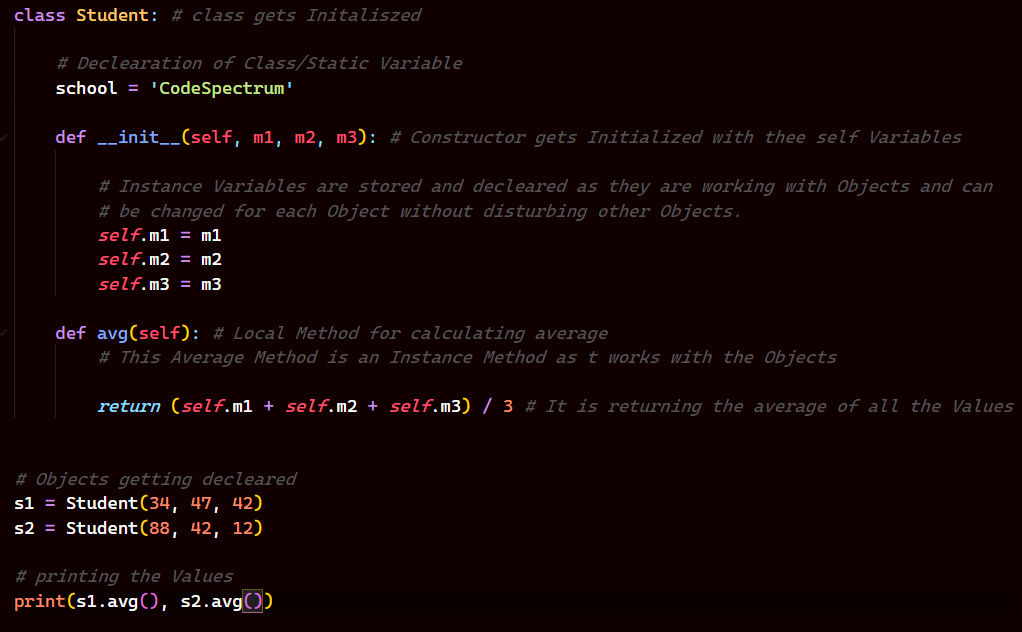
*Note: In Variables Static and Class were same but in methods that’s not the Case.*

*Let’s create a Class Student which accepts Marks of three Subjects. And we will calculate average marks. Let’s Play with that and understand the different types of Methods.*

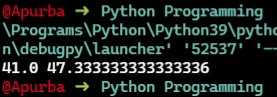
*Note: In Instance Method we have two type of Methods.*

1. *Accessors*
2. *Mutators*

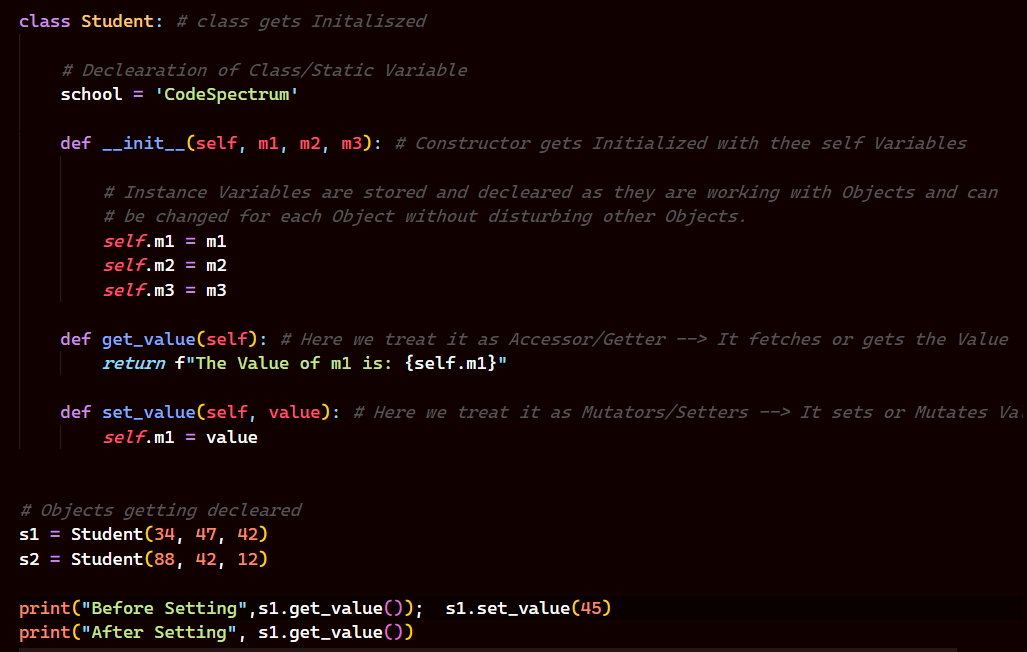
*Ultimately, we are working with Instance Variables. So, if you want to fetch a value we use Accessors and on the Other hand if we wanna change the value we use Mutators.*

**

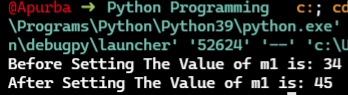
***Output:***

**

*Now, let’s see how do we work with* ***Accessors*** *and* ***Mutators***

**

***Output:***

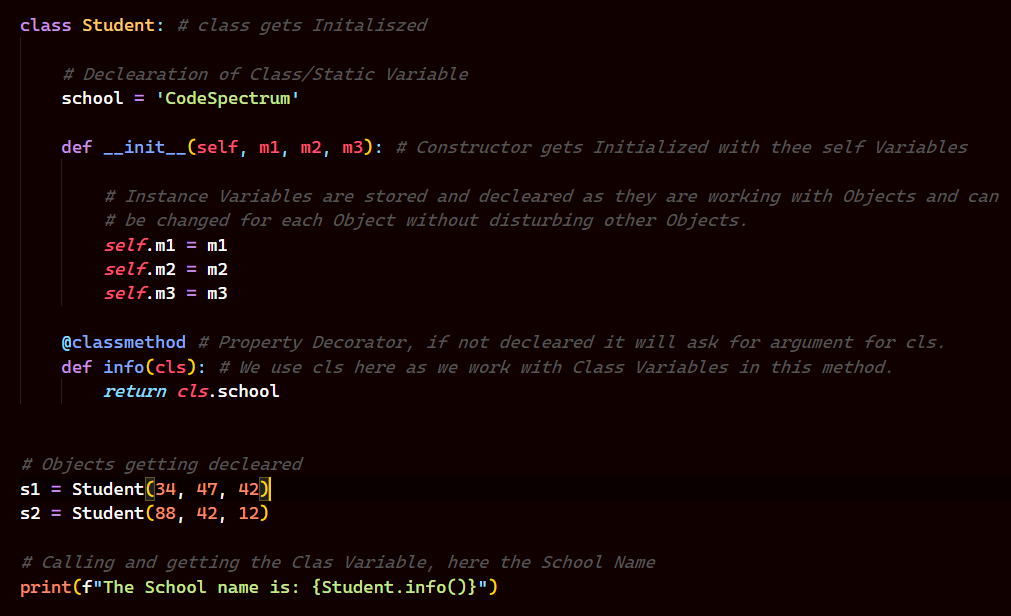
**

*Now, let’s see how do you work with Classmethod.*

*Let’s Say you wanna Know the info of a Student and it will print the name of the School, and since the Method Works with Class variables so changing the name of the School will affect all the Objects of the Class.*

*Note, here you are working with a Class Variable so you need to specify a Decorator before the method which you will declare i.e* ***@classmethod*** *also since you work with Class Variables here so you will not use self as argument in the Method rather you will use* ***cls.***

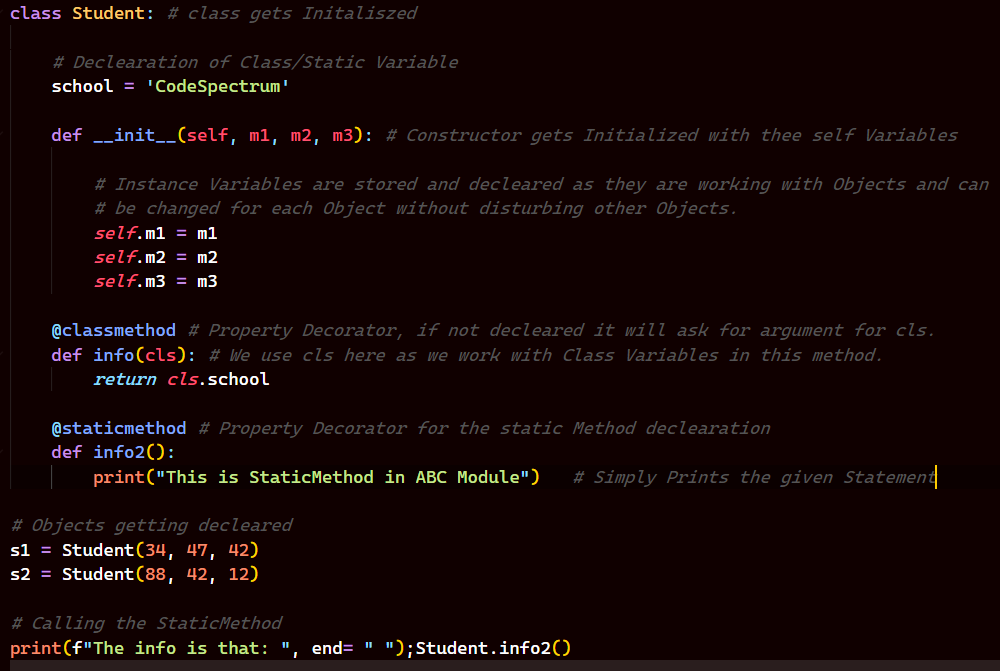
*Let’s see how this works.*

**

***Output:***

**

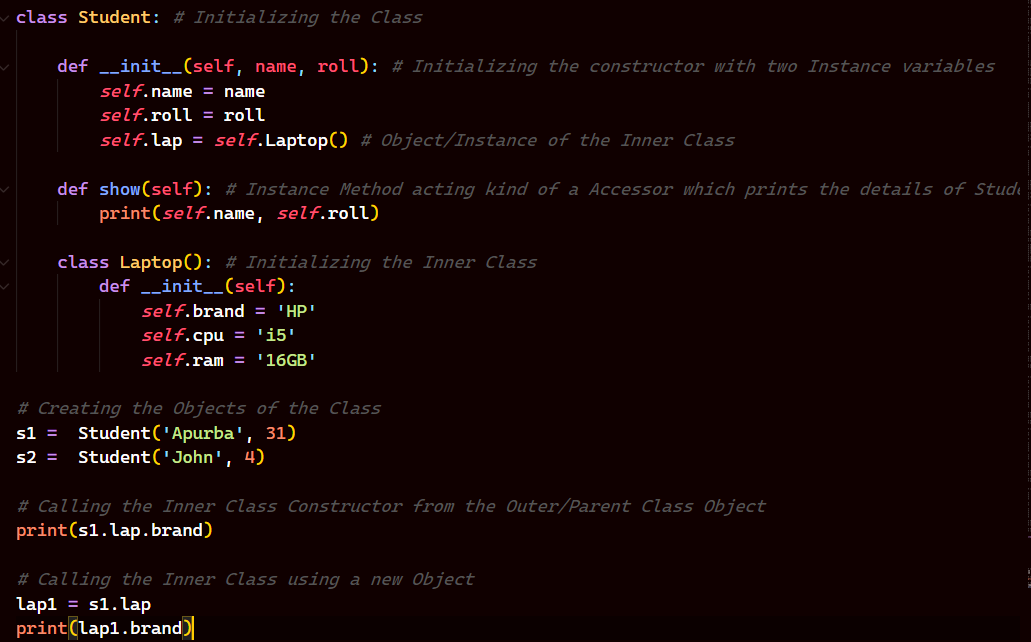
*Now, let’s see how Static Methods work. So, let’s say you have a Situation where you don’t have anything to do with Class Variables or the Instance Variables, for example factorial of a Number. Now, that what has no relation with both the variables that is when we use StaticMethods and it might be useful when you wanna work with different classes as well. Let’s see a Code regarding that.*

**

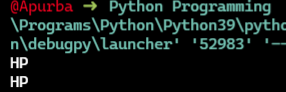
***Inner Class***

*Now, we know that we can have Functions inside a Function, a Class can have a variable and Methods inside a Class. But why do we need Class inside a class? Let’s see why*

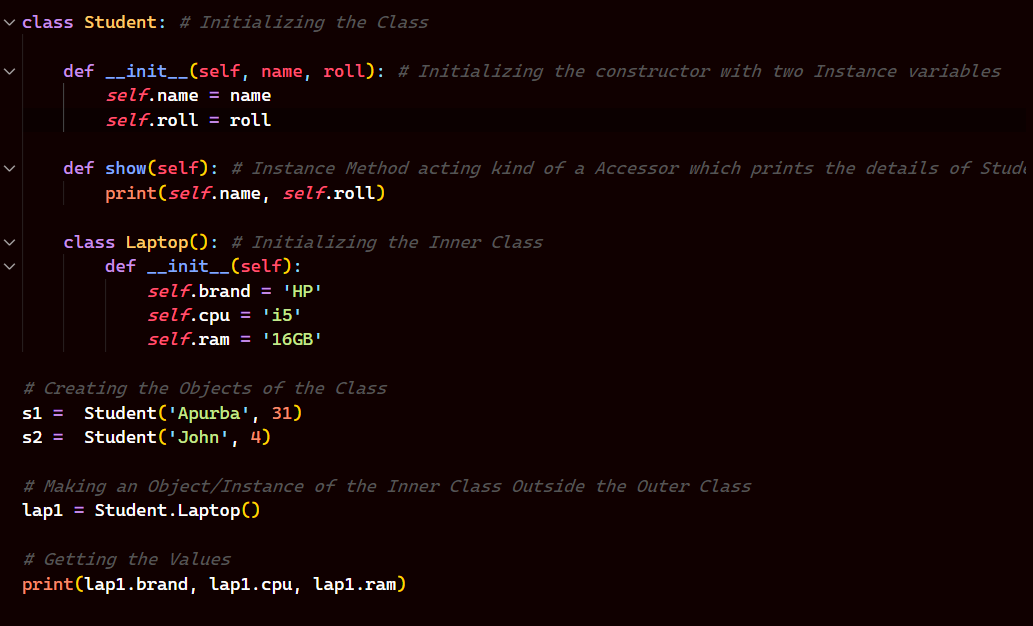
*Now, imagine a Scenario where we have a School and then we have some Students and their Credentials are stored inside a Class and the each of them has Laptops as well and we want to know that and not only that but also the Configuration. Now, what we could do is we could ask the user to pass the Details which will be hefty tasks as we will have to create different variables for different Configurations. Now, since every Student has a Laptop so why not create a class Laptop inside a Class Student and then store the configuration there in the constructor. It would be handy. Let’s see how we do it.*

**

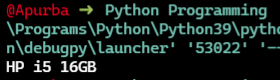
*Output:*

**

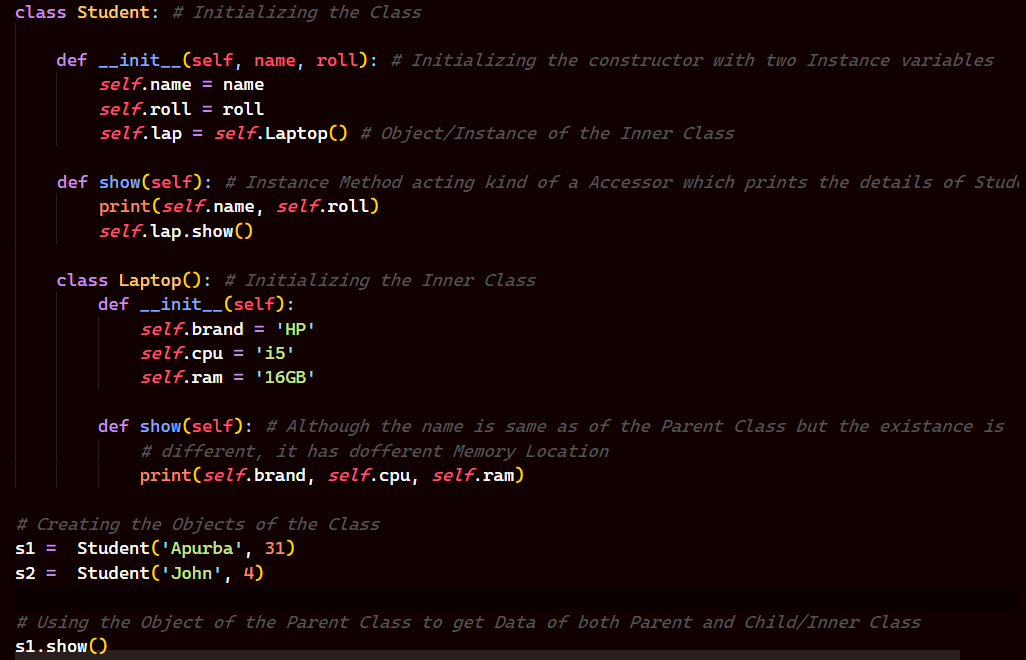
*Now, what if we want to Create an Object of the Inner Class outside the Outer Class, we could do that as well. Let’s see how to do it,*

**

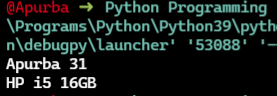
***Output:***

**

*A better Example is shown below of the Same,*

**

***Output:***

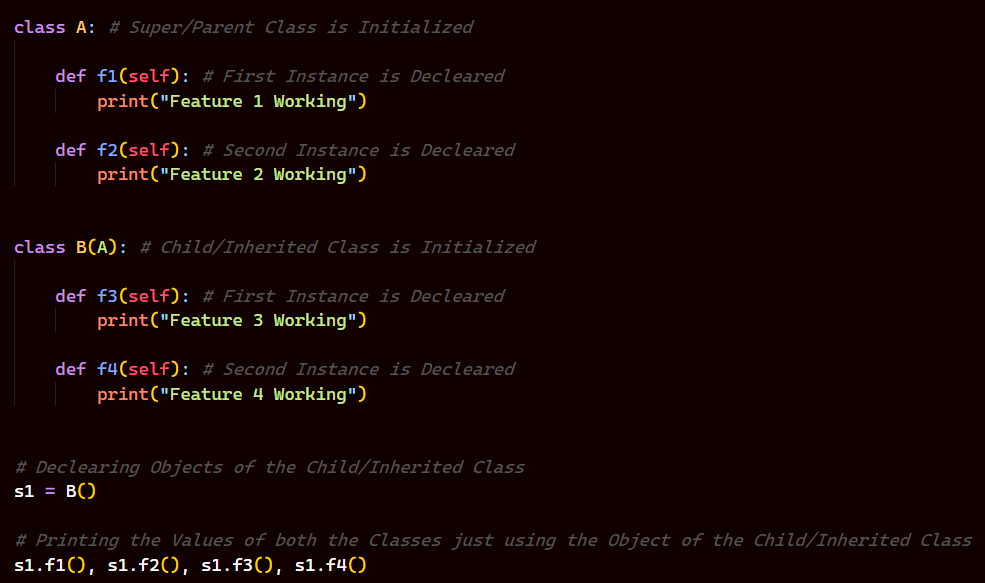
**

***Inheritance***

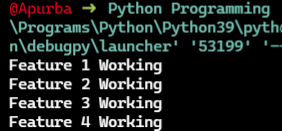
*So, the Concept of inheritance comes when we inherit some features or Properties from a Outer Class into some other Class and we do that to save ourselves some line of Code. Let’s See how we do that.*

*Now, let’s say we have a Class A which has two methods namely f1 and f2 and we can call both the methods by using Objects of the Class.*

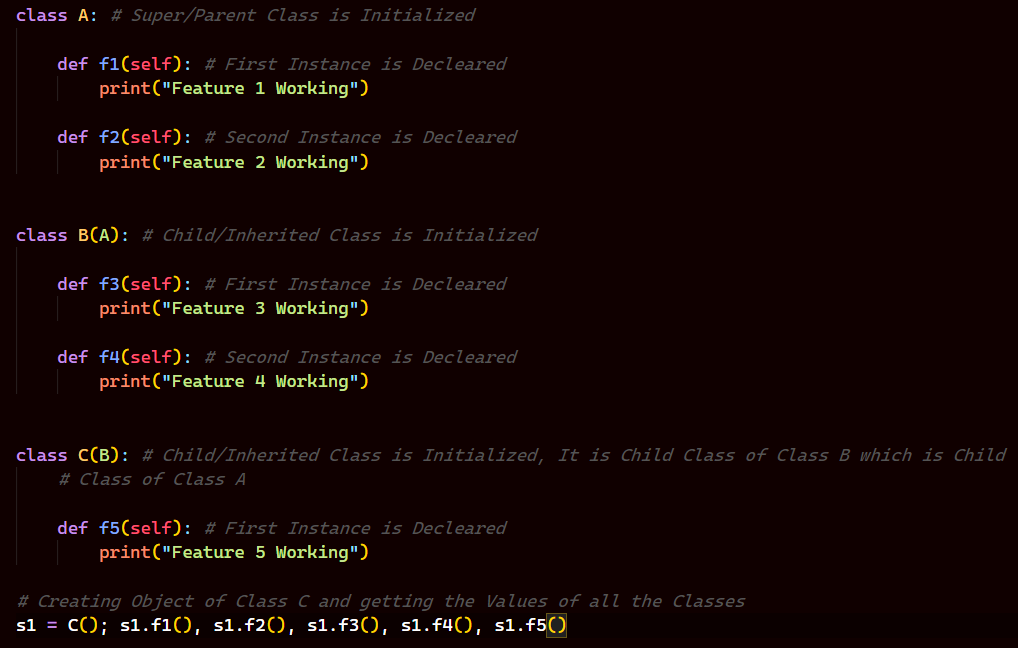
*Now lets say we are working on a big Project and we have a Separate Class called Class B and it has it’s individual methods and the situation is we want some more methods and co-incidentally that’s present in Class A, so the way I can use that in Class B is inheriting the methods of Class A. Let’s see how we do that.*

**

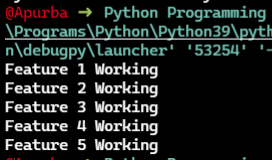
***Output:***

**

*Now, the above Code we just saw is working on* ***Single Level Inheritance Model****, where we are Inheriting features of Parent Class from Child Class, but what if we have multiple Child Classes inheriting features from the Parent Classes. Let’s see how that works. This is Called* ***Multilevel Inheritance****.*

**

***Output:***

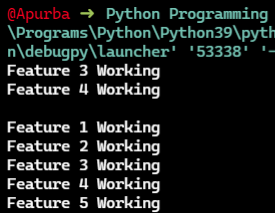
**

*Now, there are multiple way in which we can inherit features/Methods from classes to classes. But let’s see one more.*

*Here, we have three classes namely Class A, Class B, Class C and the Class C is inheriting features from Class A and Class B. So, let’s see the Code for that. It is Called* ***Multiple Inheritance****.*

**

***Output:***

**

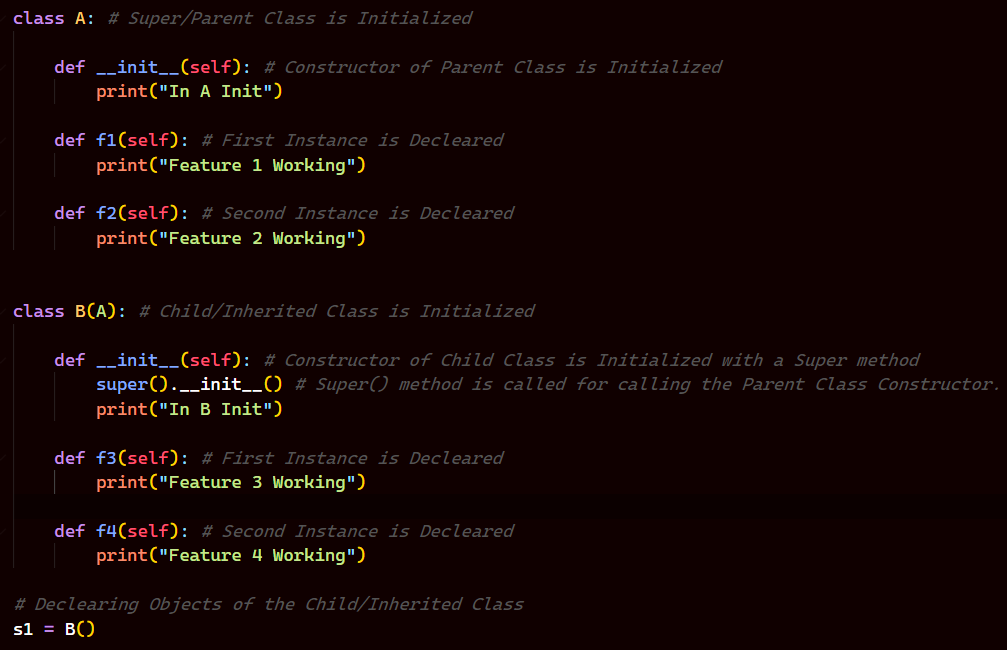
*As you see since Class B is not inheriting features from Class A so it cannot access those Methods but since Class C is inheriting features from both Class A and Class B so it has access of all the methods.*

***Constructor in Inheritance***

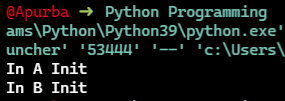
*Now, here we will be learning about* ***Constructor in Inheritance*** *and* ***MRO*** *(****Method Resolution Order****).*

***Note: Subclass can access all the features of Super/Parent Class but that’s not Vice-versa.***

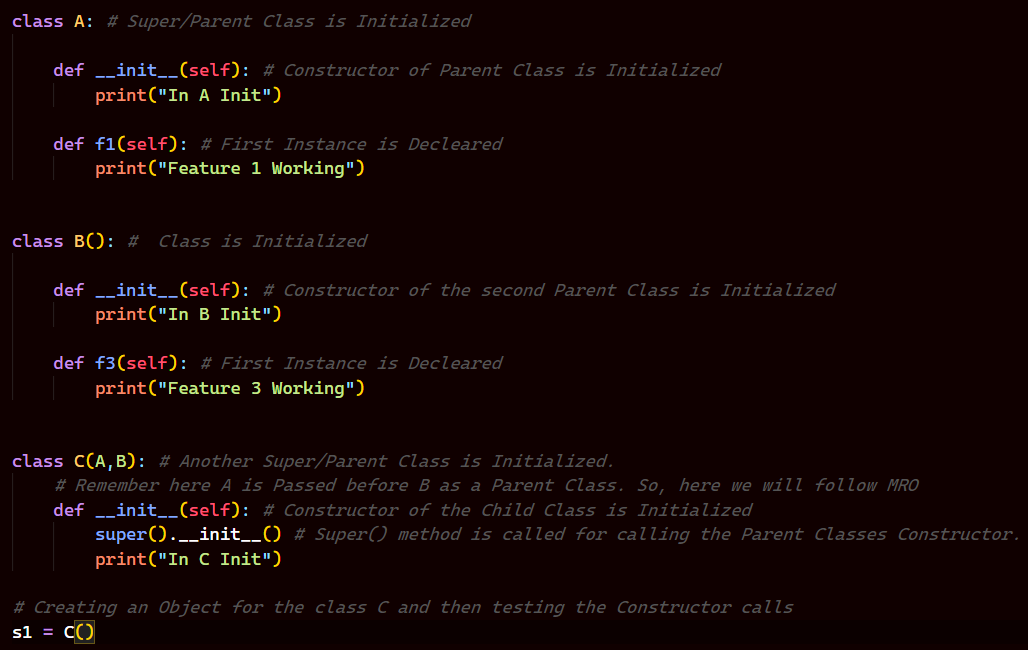
*Now, let’s imagine a scenario where we have two Classes Class A and Class B, now Class B is Inheriting features from Class A and now we define a Constructor in Class A or here the Super/Parent Class and now if we make an Object of Class B then we can access all the features but if we make an Object of Class A then we just can access features of Class A only and not Class B, now if call Object A or B then it will Constructor of Class A in both the Cases. But if we have a Constructor of Class B itself then we make an Object of B And call it then the Class B constructor will be called. But is there a possibility that I can call both the constructors of Class A and Class B, yes we have for that we use a Special method/function called Super(), let’s see all of these in Code.*

**

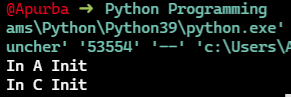
***Output:***

******

*Now, let’s say we have three classes Class A, Class B and Class C and then Class C inherits features from Class A and Class B, now we have a Constructor for all three classes and we have an Object for Class C then obviously constructor of Class C is called but since it has two Parent Class and if we mention Super in the C Class Constructor then it calls the Constructor of Class A since we have passed A,B in the* ***MRO*** *(****Method Resolution Order****), which says if you have multiple inheritance it will start from left 🡪 right. Let’s see the Code.*

**

***Output:***

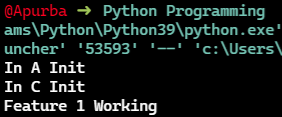
**

*As you can see due to the MRO rule of instruction Class A constructor is being called rather than calling both A and B.*

*Now, since the Class C has access to all the methods of A and B but super in Class C has access only to the Class A, so if we want to call a method using super in Class C then we can only call methods of Class A and not Class B. Let’s see the Code.*

**

***Output:***

**

*See, the contents for f1() method of Class A could be called from Class C using Super() method.*

***Note: To represent a Super/Parent Class we use the Super Method.***

***Polymorphism***

*Now, talking of Polymorphism breaking the word gives us, poly = many, morph = form, so an Object having different forms, or One form many uses so we use this concept many times in Industry Level Works, like* ***Loose coupling****,* ***Dependence Injection, Interfaces.***

*We can achieve Polymorphism in Python in 4 ways,*

1. *Duck Typing*
2. *Operator Overloading*
3. *Method Overloading*
4. *Method Overriding*

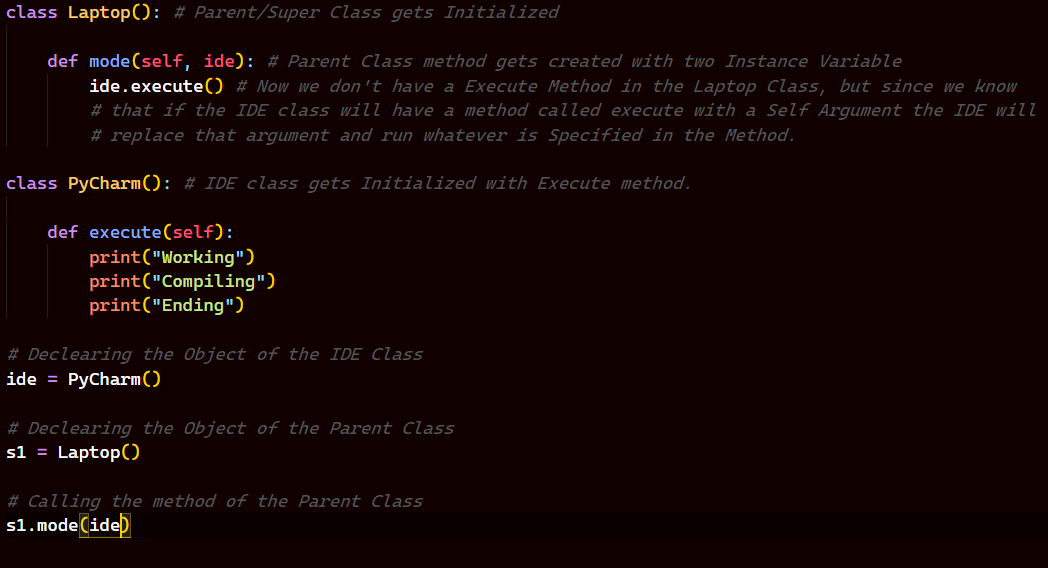
*Now, let’s talk about all of these in detail.*

***Duck Typing***

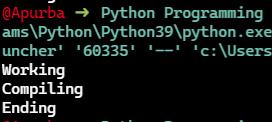
*So, in Python we have Dynamic Typing and it suggests that we will not have to mention Data Type of the Variables. Because, Variables in Python are just names to a certain Memory Address where the Value can Change so really don’t have a Specific type for that Variable. So, lets understand it by an Example.*

*Now imagine you have a Class called Laptop and there is a method in it which takes an Argument namely your IDE which you use, now as it passes the IDE it gets executed now we don’t have a method called Execute nor the IDE so we create a class for that IDE and we have the method execute in there. And then it does some work. Now the Idea is that it is not necessary that the IDE has to be only PyCharm or Vs Code it could be something else as well, so it defines that IDE can change it is not fixed provided that the new IDE class has a method named execute in it. That’s what is Duck Typing. The Type can change but the Method Remains the Same.*

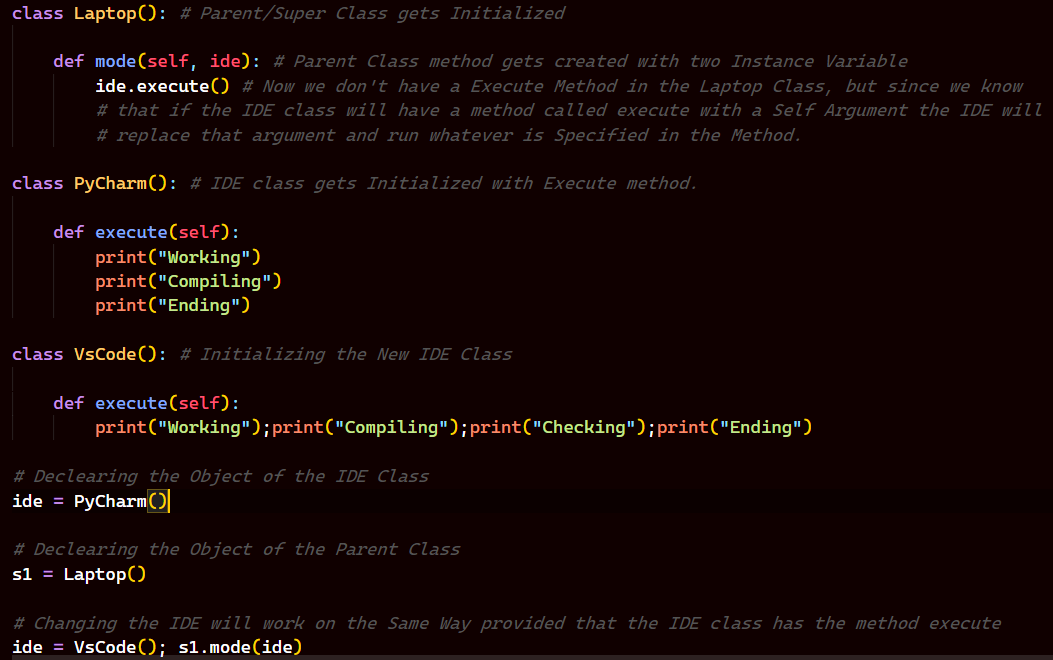
*Let’s see the Code for that.*

**

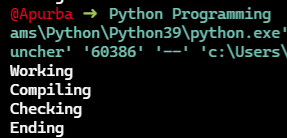
***Output:***

******

*Now, let’s see if we change the IDE what happens with the Code,*

**

***Output:***

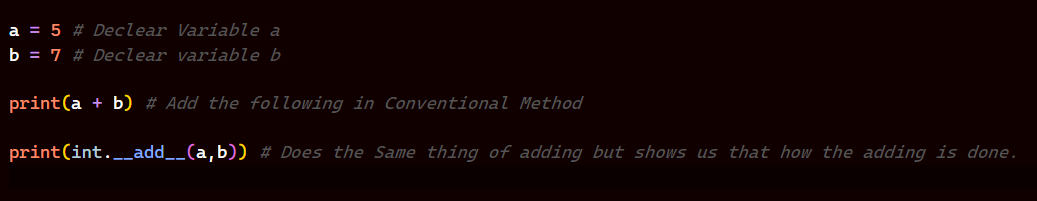
**

*So, as you see the IDE changes to Vs Code now.*

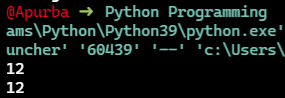
*That’s all about Duck Typing in Python.*

***Operator Overloading***

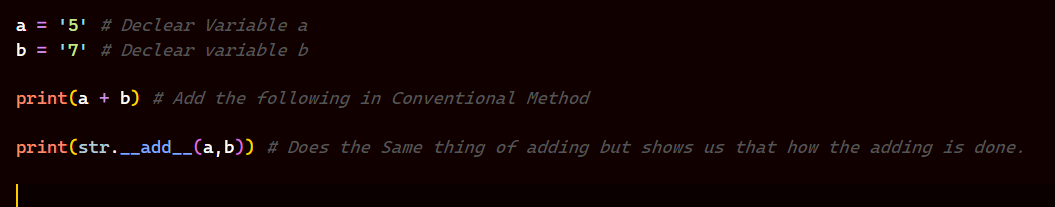
*So, now we know that we have Operators in Python. We have different of them, we have int, float, bool, etc. Now, let’s say you want to add two number so you will simply just add them with the ‘+’ symbol and you get the results and that’s really sugary like Synthetic Sugar, where everything is done for you. But, behind the scene something else is happening, so as we all know in Python Everything is an Object and these datatypes are individual classes in itself. So, they are of <class ‘int’>, <class ‘float’>, <class ‘bool’> and etc. So, they should also have some methods. So, in case of Addition when you add two numbers what happens behind the scene the ‘int’ class calls the “\_\_add\_\_” method in Python. Now, add also takes exactly two arguments as you have added the two numbers and gives you the same result. Let’s see it in action ..*

**

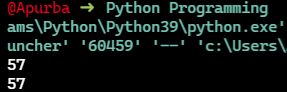
***Output:***

**

*Same goes for String as well, Let’s see*

**

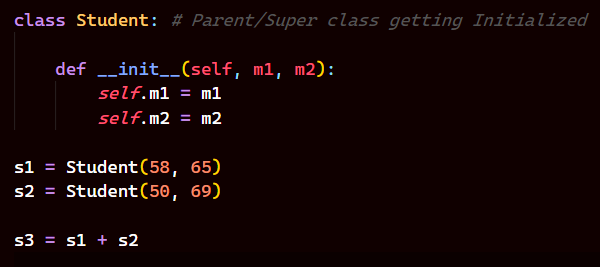
***Output:***

**

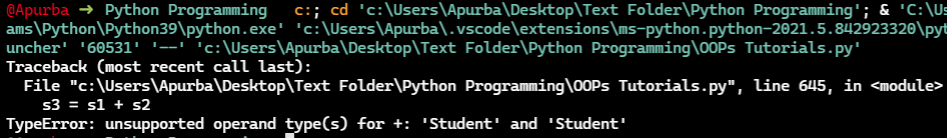
*So, as we see the moment we use ‘+’ in Python behind the scene “\_\_add\_\_()” is getting called, the moment we use ‘-‘ in Python behind the Scene “\_\_sub\_\_()” is getting called and so on. So, we have different methods for all types of Operators and they are being called* ***“Magic Methods”.***

*Now, let’s explore the concept of these Operators in a Class defined by us.*

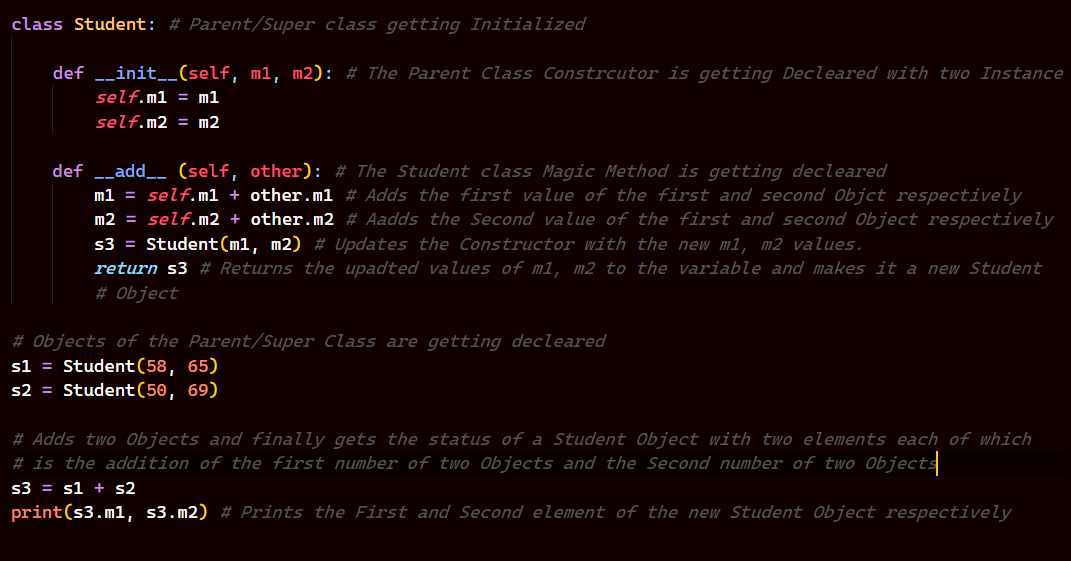
*So, say we have a Class Student and a constructor taking two values namely the marks or any int value. And we have two Objects of the same Class. Now, if we add the two Objects, let’s see what we get,*

**

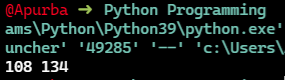
*The Output comes like,*

**

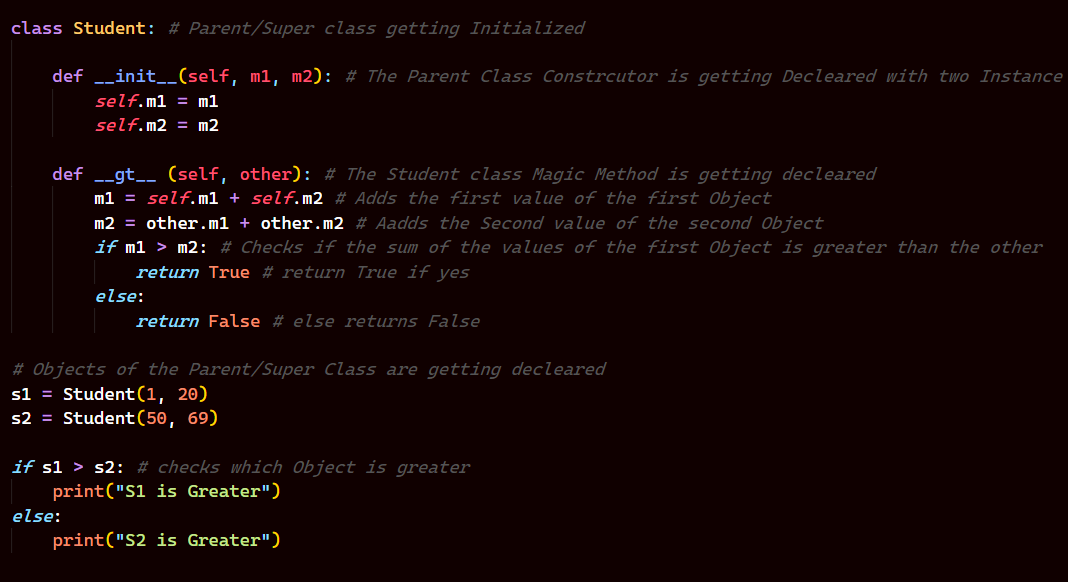
*It says Unsupported Operand Types for +: ‘Student’ and ‘Student’, but then we know that when we use ‘+’ the “\_\_add\_\_()” method is called in behind. But in our class we don’t have that method, so we need to define that ourselves in our class, and that’s what we called Operator Overloading, where we Overload the Operator by defining our own Magic Method related to it. Let’s see the Code for that,*

**

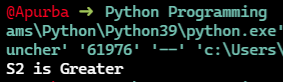
*The Output of it is,*

**

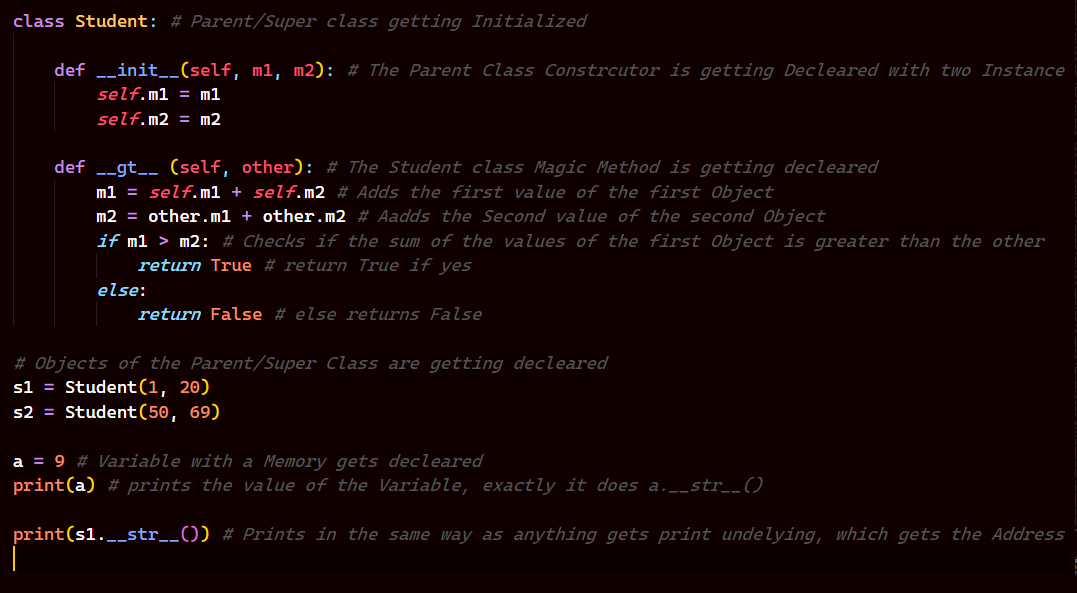
*Let’s see an Example where we compare two Objects of a Class.*

**

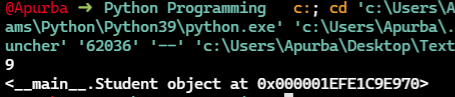
*And, the Output of the Code is,*

**

*There is one more thing to Look at. Now, say we have a Variable having Value 9 and if we print the Variable it prints it’s value and not the Memory Address of that Variable. But when we print an Object of class it shows us the Memory Address. What happens is that underlying a method named “\_\_str\_\_()” is getting called and it is printing the mentioned values. Now, we don’t want to see the Memory Address when we prints the Object. So, we will create our own “\_\_str\_\_()” method i.e overload the existing method. Let’s see that in Code.*

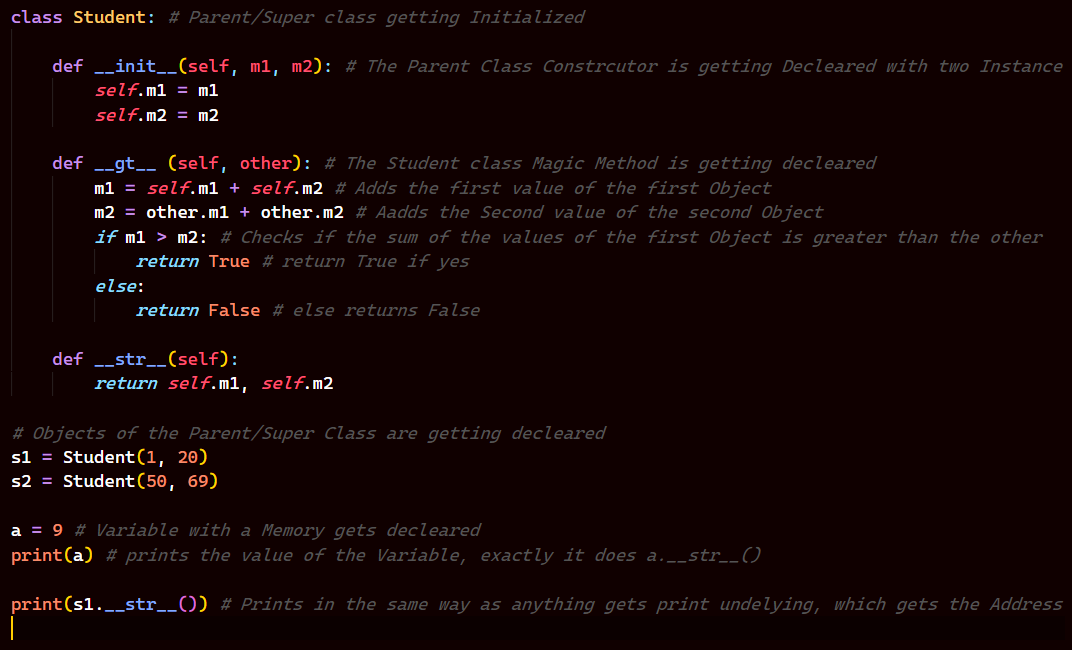
**

***Output:***

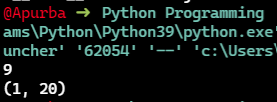
**

*Now, we want to Overload/Override the existing Operator and define our own such that if we print an Object, it’s values gets print.*

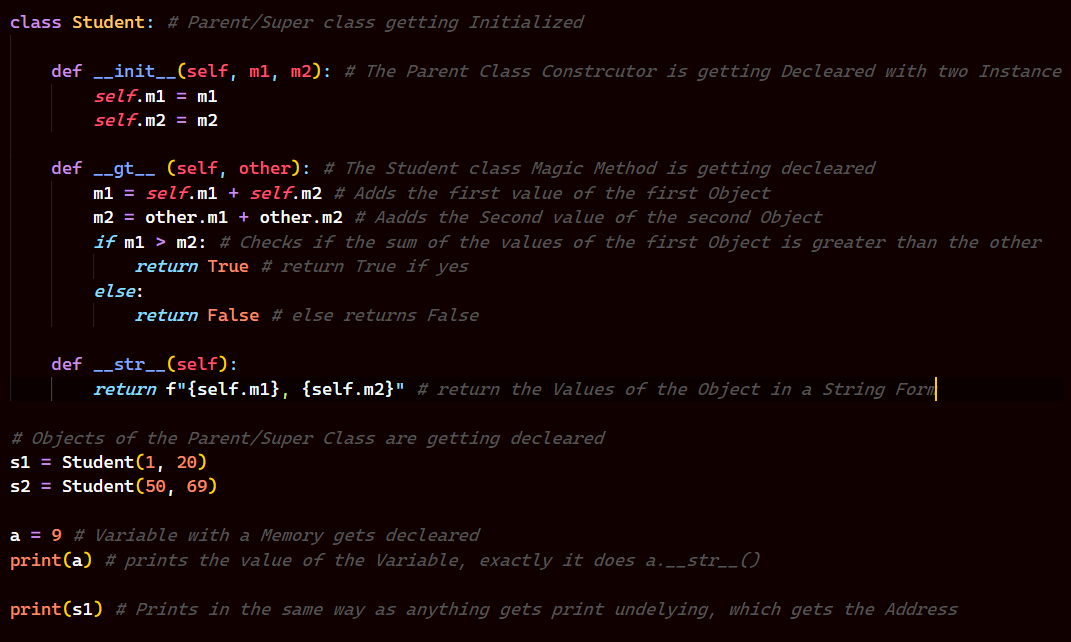
*Let’s see that,*

**

*Let’s See the Output,*

**

*But, if you see we explicitly mention the “\_\_str\_\_()” method in the print statement, if we remove that it will say* ***TypeError: \_\_str\_\_ returned non-string (type tuple),*** *now since we can return only str values from str magic method, we need to return it in the form of a String. Let’s see that in Code.*

**

*And the Output will be like,*

**

*So, these were some ways with which we perform Operator Overloading in Python.*

***Method Overloading & Overriding***

***Method Overloading***

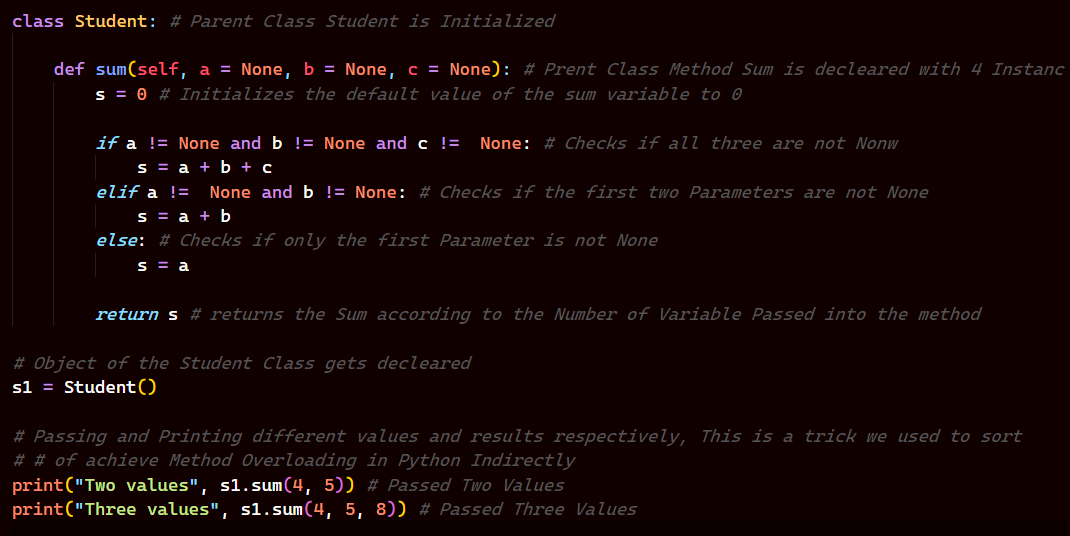
*Now, Method Overloading is a process where let’s say we have two Methods of same name with different Parameters or Arguments. Like say we have a Student Class with average method one taking two params and the other taking three params. That’s what we call Method Overloading. But in Python we generally don’t have this Concepts.*

***Method Overriding***

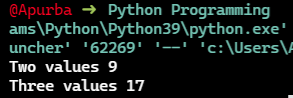
*Now, Method Overriding is a Process where let’s say we have two methods of same name with same Parameters or Arguments. Like say we have Student Class with avg method one taking 2 params and the other as well. Now Obviously we can’t have something like that in Same Class but if we have a concept of Inheritance where the Parent/Super and the Child Class has the same name method with same number of params. That’s what we call as Method Overriding.*

*Let’s start with Method Overloading first,*

*Now, imagine a scenario where we have a method called sum in a Class called Student and we add number using that method. Now, by default we pass 2 values and it accepts that, but then what if I send 3 values or maybe 1 value, So in other languages we can use Method Overloading we can Override the methods. But, Python does not support that so what we can do is we can get the default values of the params to None and then check which param is not None and accordingly return the values. Let’s see how it’s done in Code,*

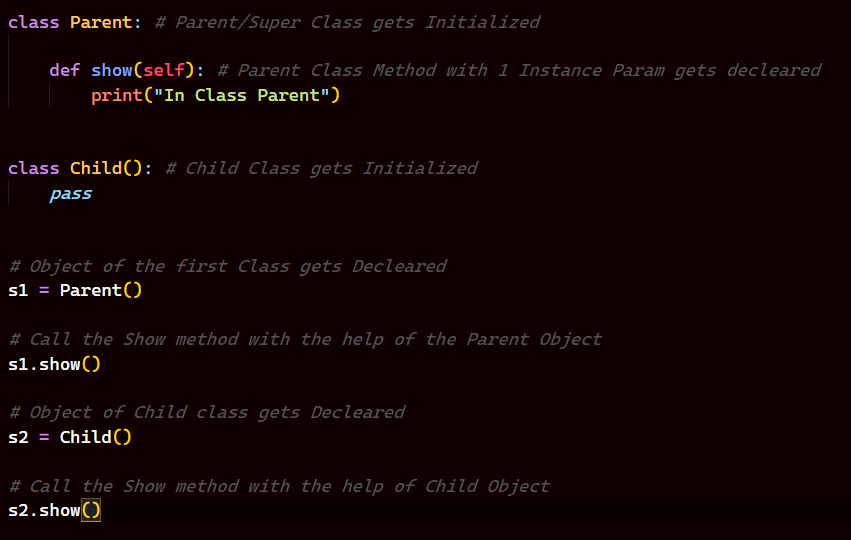
**

***Output:***

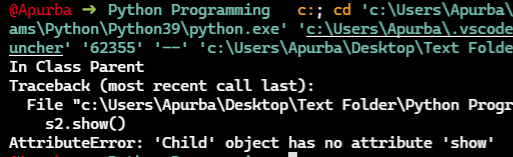
**

*Let’s start with Method Overriding now,*

*Now, imagine a scenario where we have a Parent Class and it has a method called Show. Now I created a Child Class and it has nothing in it. So if I were to make an Object of the Class Child, and with the help of that Object I call Show method it would show me Error. Let’s see that in Code,*

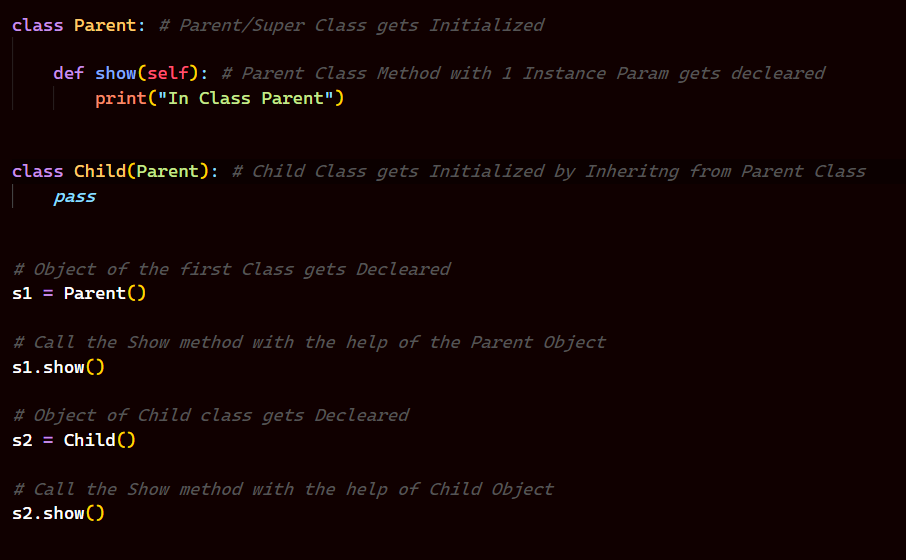
**

*And the Output is pretty Obvious that we will get an error in the s2.show() as we don’t have any methods written in Out Child Class.*

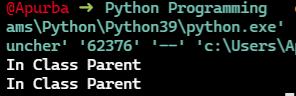
**

*As expected we got that,*

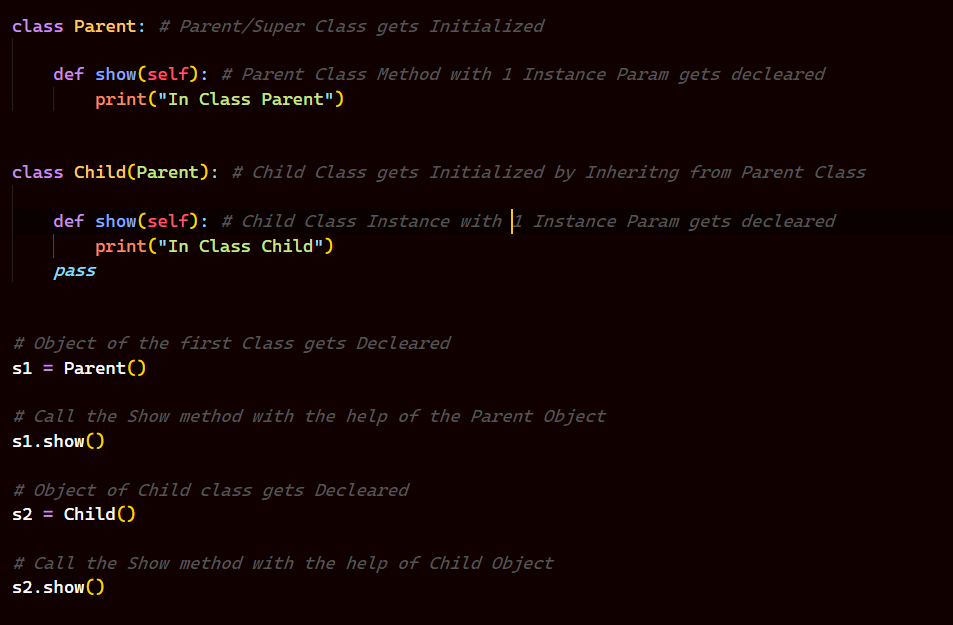
*Now, let’s say the case where the Child class is inheriting from Parent Class. Then if we write the same code it would show use the content of the show method in Parent Class as we have inherited all features of it. Let’s see the Code.*

**

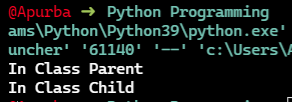
*And the Output is we will get “In Class Parent” two times.*

**

*Now, what if we had our own show method in the Child Class. Let’s see the Code and what the Output would be,*

**

***Output:***

**

*So, as we have our own show() method in Child Class now. So, the existing show() method gets Override which we inherited from the Parent Class and now the show method of child class is shown as we call the method using the Object of Child Class. This is what we call* ***Method Overriding*** *in Python.*

***Abstract Classes***

*Now, Python by default does not support Abstract Classes and the way we achieve it is using a Third Party Module called* ***ABC*** *Module. ABC Module stand for* ***Abstract Base Classes****.*

*Now, let’s start and ask a question,*

*What is Abstract Class and why do we need it?*

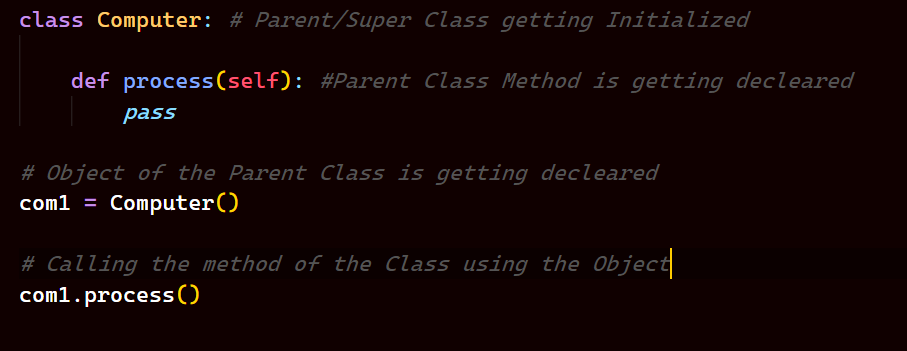
*Let’s start with, What is an Abstract Class?*

*Now, let’s create a Class called Computer with a method called process.*

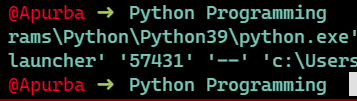
***Note: A Method consists of a Name method i.e the params it accepts and a Module i.e the Body of the method is named as Module.***

*Now, the method does not have anything here, it just have a pass statement. So, this type of methods which has a Declaration but not a*

*Definition(body/Module) is called as Abstract Method, and the classes which has Abstract Methods are called Abstract Classes. Now, Python does not Support Abstract Classes by default. So, if you make an Abstract Class and run the Code it runs fine. Let’s see it in Action.*

**

***Output:***

******

*As we see that there is no Output, because we have written pass in the method so nothing happens.*

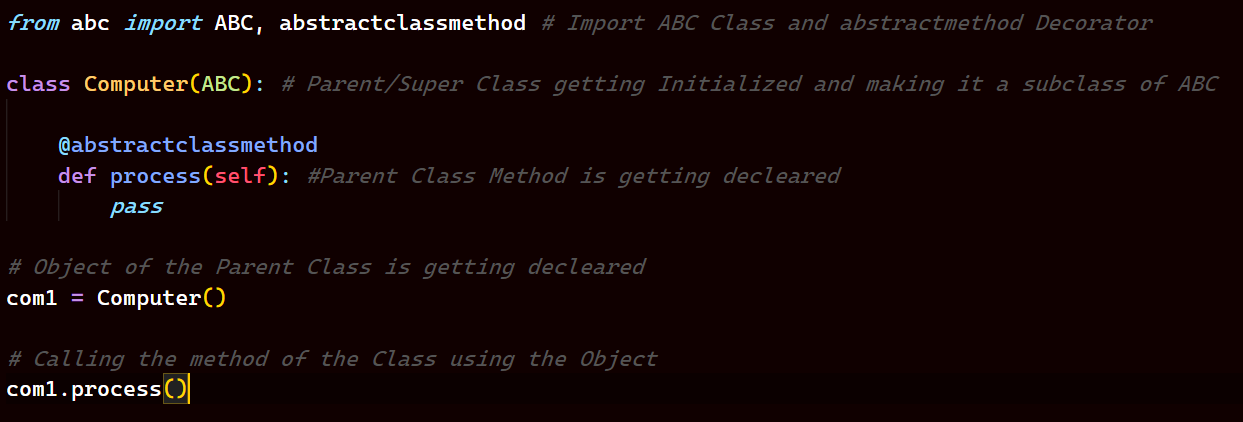
*But, the catch is that had Python supported Abstract Classes then it would have given error. As,* ***Abstract Class does not encourage making Objects of it.***

*Now, let’s see how can we make Abstract Classes using ABC module.*

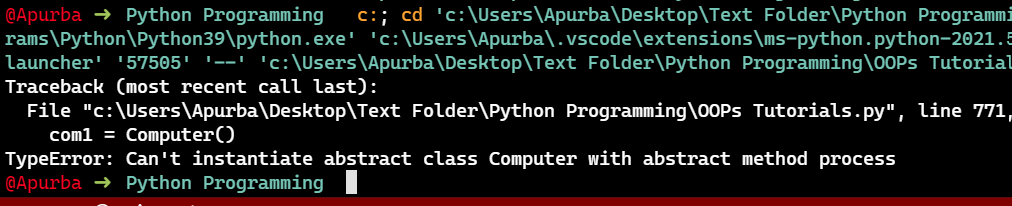
*So, we import two things from ABC module,*

1. *ABC*
2. *abstractclassmethod*

*now we make out Class a Sub Class of ABC and implement the abstractclassmethod as a decorator in the method of our class. Let’s see how can we do that in our Code.*

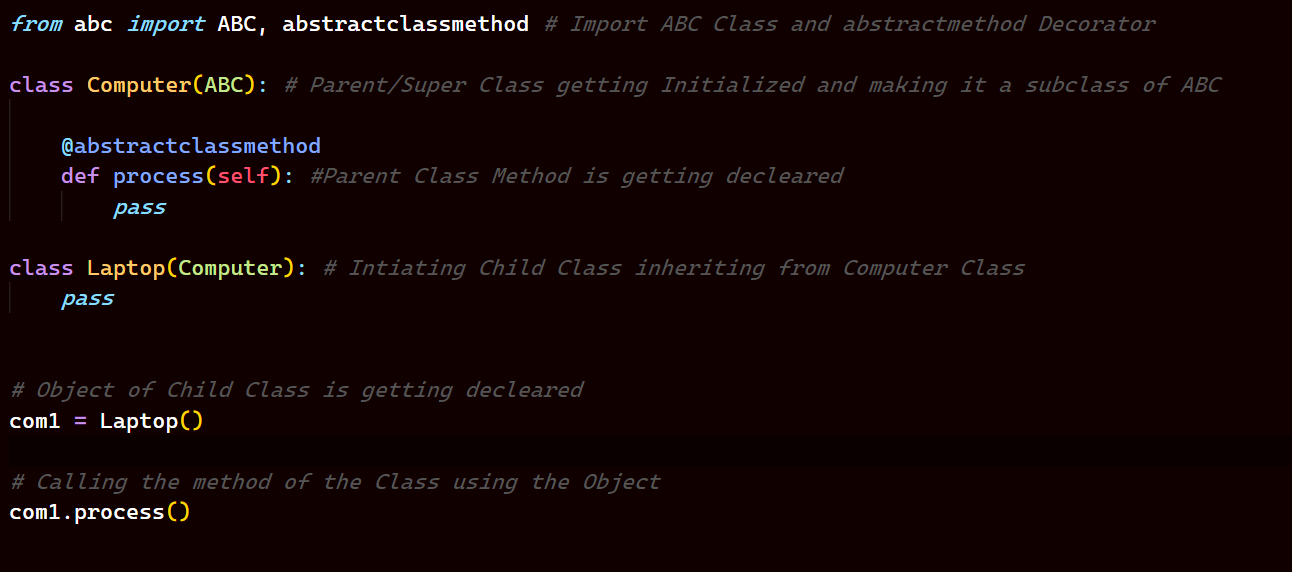
**

*Now, since we have successfully created our Class Computer as an Abstract Class, so now running the code will give error as we know that Abstract Classes cannot have Objects of it,*

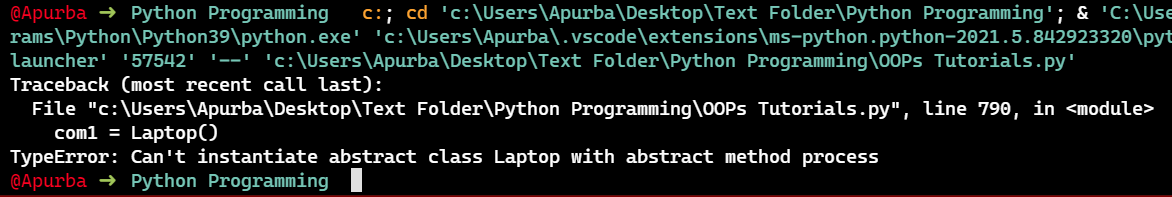
**

*As we see, that it gives Error as we expected.*

*Now, let’s see if we work on Inheritance and then create a Sub Class of Computer and the try to run the method of the Parent Class. Let’s see what happens.*

**

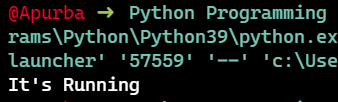
*The Output comes to be,*

**

*So, even based on Inheritance model we get the Same Error. We can Omit this error only when we defined the method getting called when we define it exclusively in Child Class.*

**

*The Output is,*

**

*So, as we see it the process method in the Child Class is running.*

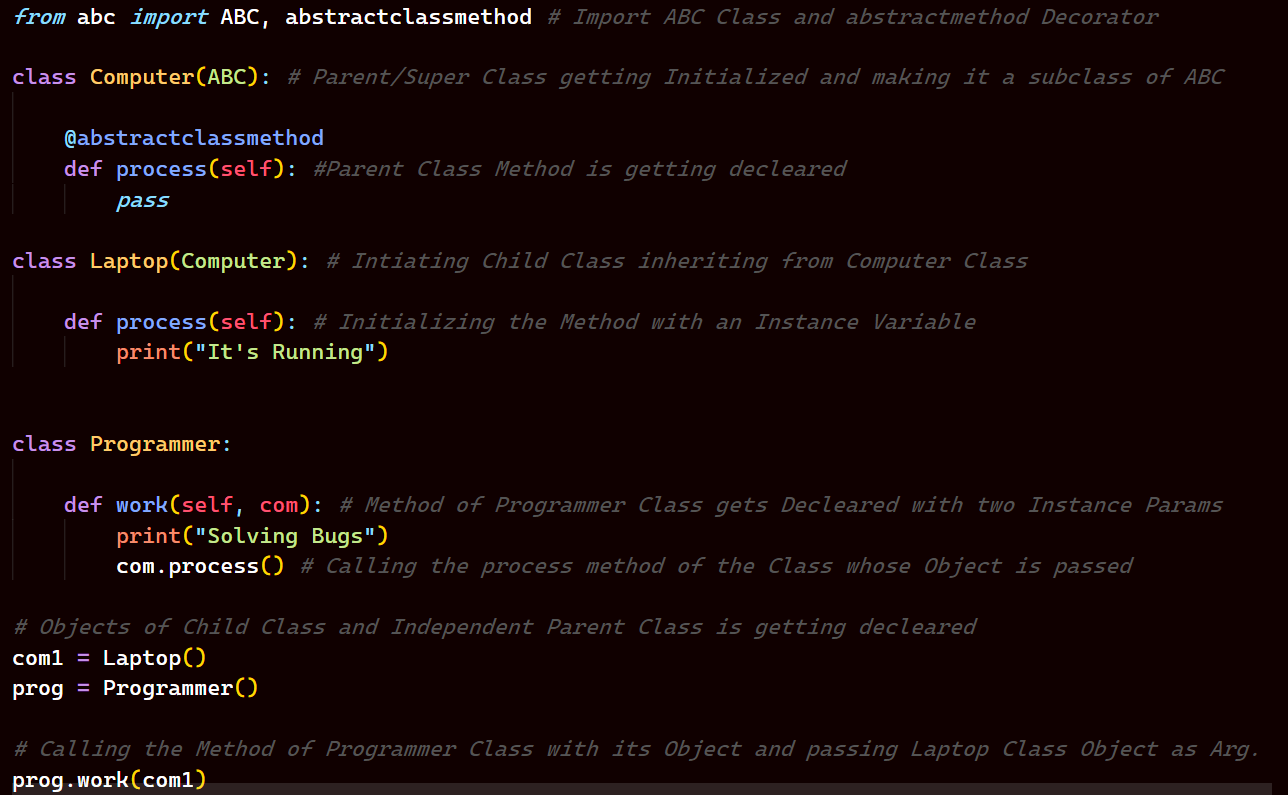
***Note: Abstract Class should have atleast one Abstract Method.***

*Why do we need Abstract Method?*

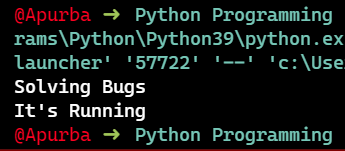
*Obviously, why do we even need an Abstract Class, even if we don’t use the Class Computer then also Our Code does not changes it works fine. Even what does the Abstract Class does to us, it just sits idle and serves no Purpose to us, now if we had methods which our Child Class needs then might be the Case that the Abstract Class could be of some use, but then if it has only Abstract Methods then it is of no use right? No, we still need it, it has uses if you follow the OOPs process/way of writing Codes. Let’s see it how?*

*Just to understand, let’s say that you have a Class Programmer with a Method in it and having a Module within it and a Class Laptop with a method in it. Now we create an Object of the Programmer Class and the Laptop Class, so if we call the method it will do the job in the Module of the Method. And then we call the method of Laptop Class from the Method of the Programmer Class.*

*Now let’s Play around with the Code and see what happens and Understand, Why Abstract Classes are necessary?*

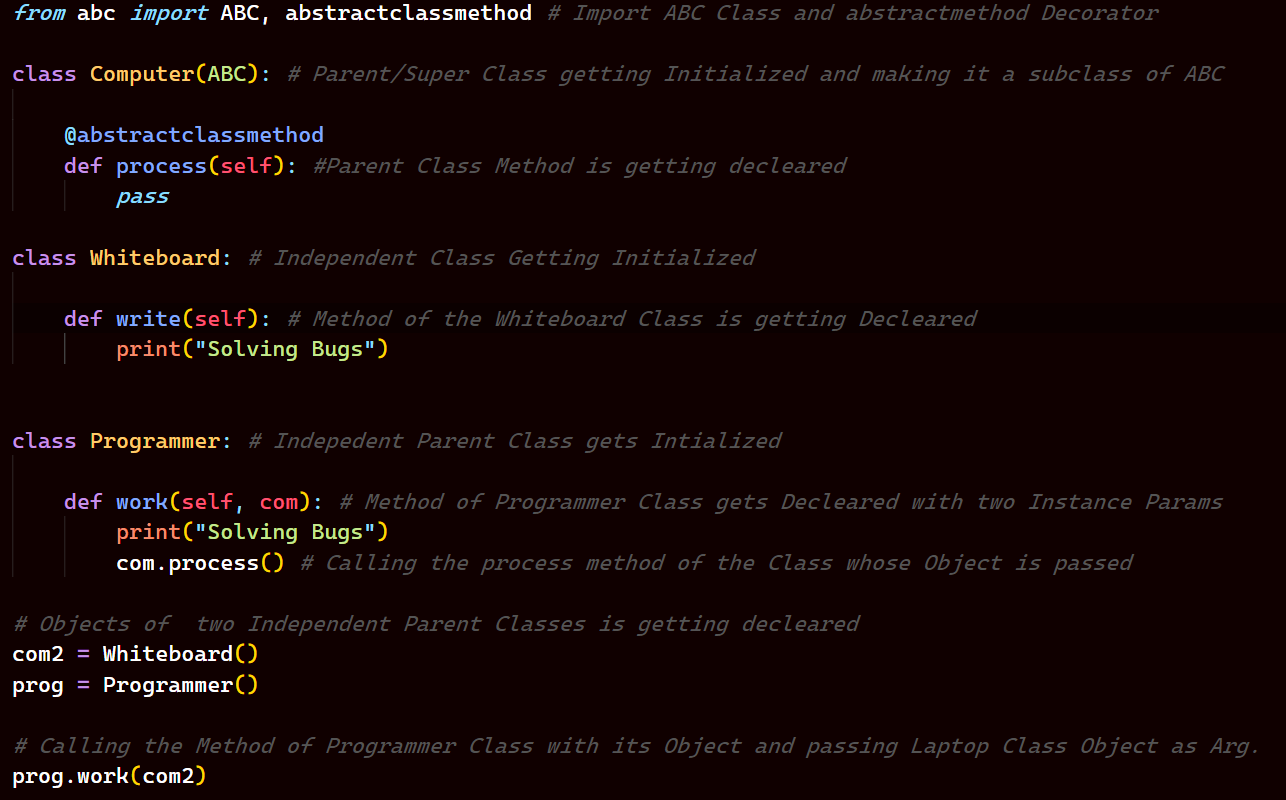
**

*Let’s see the Output:*

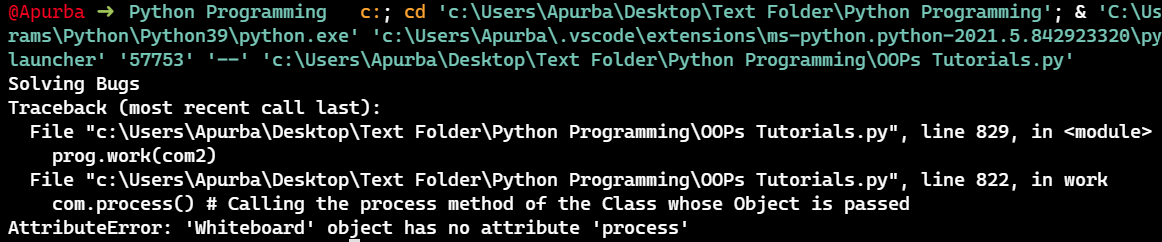
**

*As expected we first call the Method of the Programmer Class and then using the Object of the Laptop Class which is passed as the Argument into the Method of the Programmer Class we call the Method of the Laptop Class.*

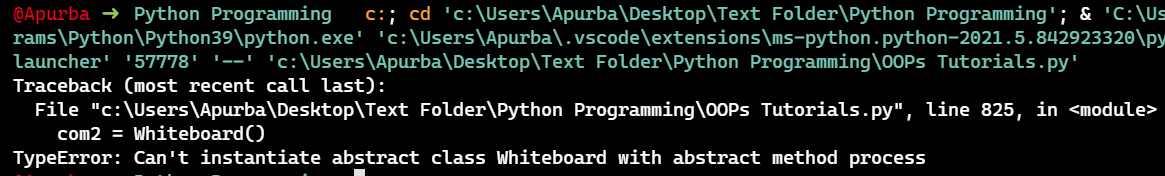
*But now, what if we had another Class called Whiteboard, and we wanted to pass it’s Object into the Programmer Classes Method as Argument. What happens now?*

**

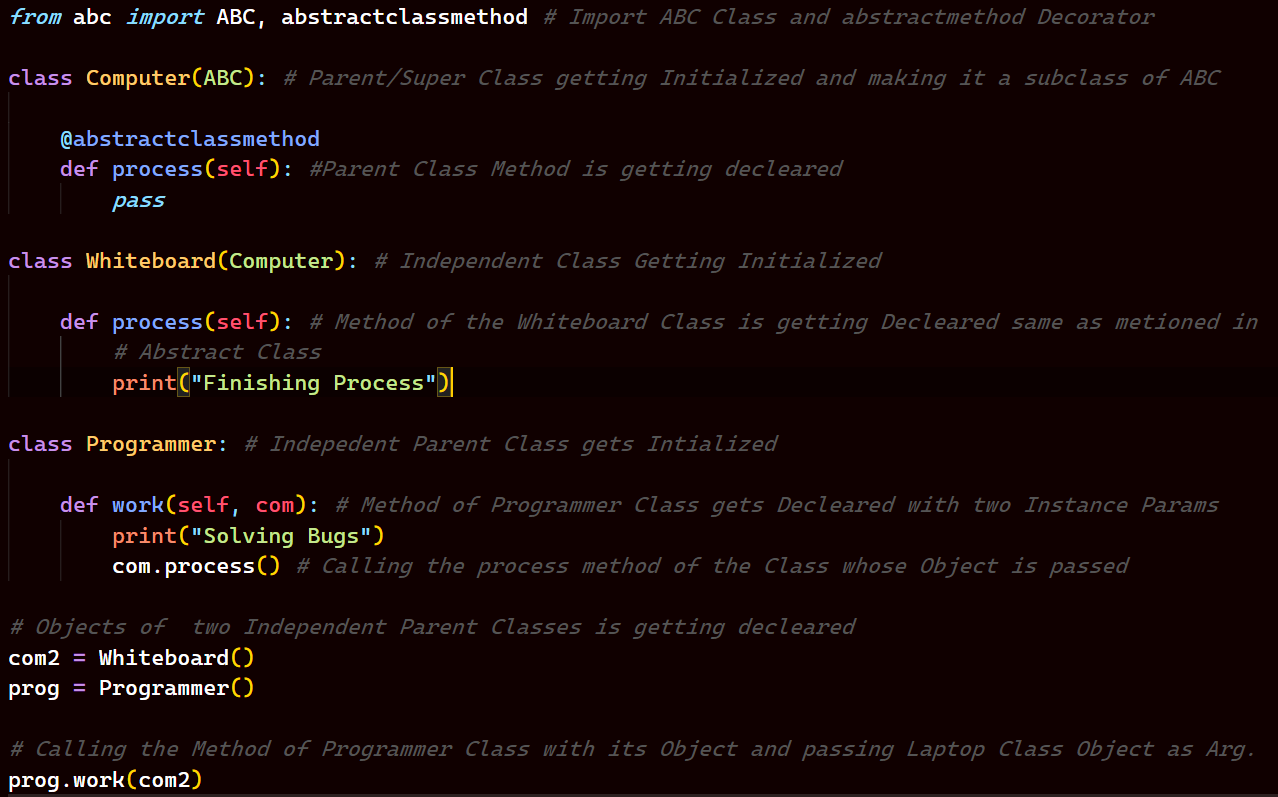
*The Output comes to be,*

**

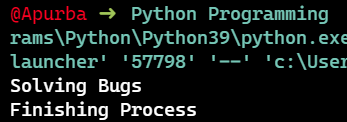
*So, we see that it is said that Whiteboard has no Attribute Called process(), Now looking at it deeply, Whiteboard or the Code has no fault here , as there is no Compulsion for it to have the Attribute called Process and that’s why we have not define it. Now let’s imagine a situation like this that we are working on some Code and there is a method we have and we will use that method repeated times. And whatever class of same type we want to create we want to have that method called process() in that Class, that’s where Abstract Classes comes into Picture. If we make any class we make a sub Class of the Abstract Class then it is necessary for that classes to have the methods with same Attribute defined into the Abstract Class. So, here if we make the Class Whiteboard a Sub Class of the Computer Class then the Class Whiteboard will have a Compulsion of having an Attribute called process(), if it does not have it. So, we will get the following error,*

**

*So, now if we make Class Whiteboard a sub class of the class Computer and define a method called Process() in the Class Whiteboard then let’s see what happens.*

**

*The Output comes to be,*

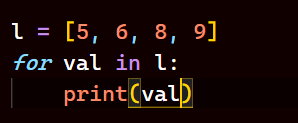
**

*So, now we see that the Code runs fine.*

*Hence, we can say that Abstract Class comes in handy when we want some classes to have some specific methods defined always, then we can bound them to Abstract Classes and define the required methods in the Abstract Class. It is about designing your Class.*

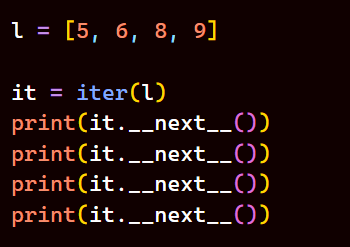
***Iterators***

*Now, we know of iterators, when we use a loop. Imagine a Scenario when we see that there is a list and we need to print the Values of the list and we could potentially do that by running a loop. This is the way we do that,*

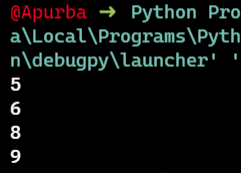
**

*Now, what if I had to say we could do this by using iterators, Let’s see how do we do that.*

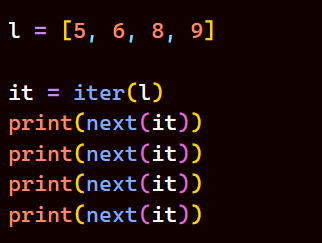
*First, we use a function called Iterator, and what it does is it converts your argument into an Iterable Object. And then we have a function which call the values from the iterables, the name of the functions are “\_\_iter\_\_()” and “\_\_next\_\_()”. Let’s see the Code in action.*

**

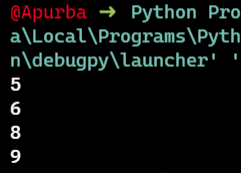
*Here “\_\_next\_\_()” is a method of the iterator. So, each time we call the next function it gives the current value. The Output will be,*

**

*We could also write the code like this,*

**

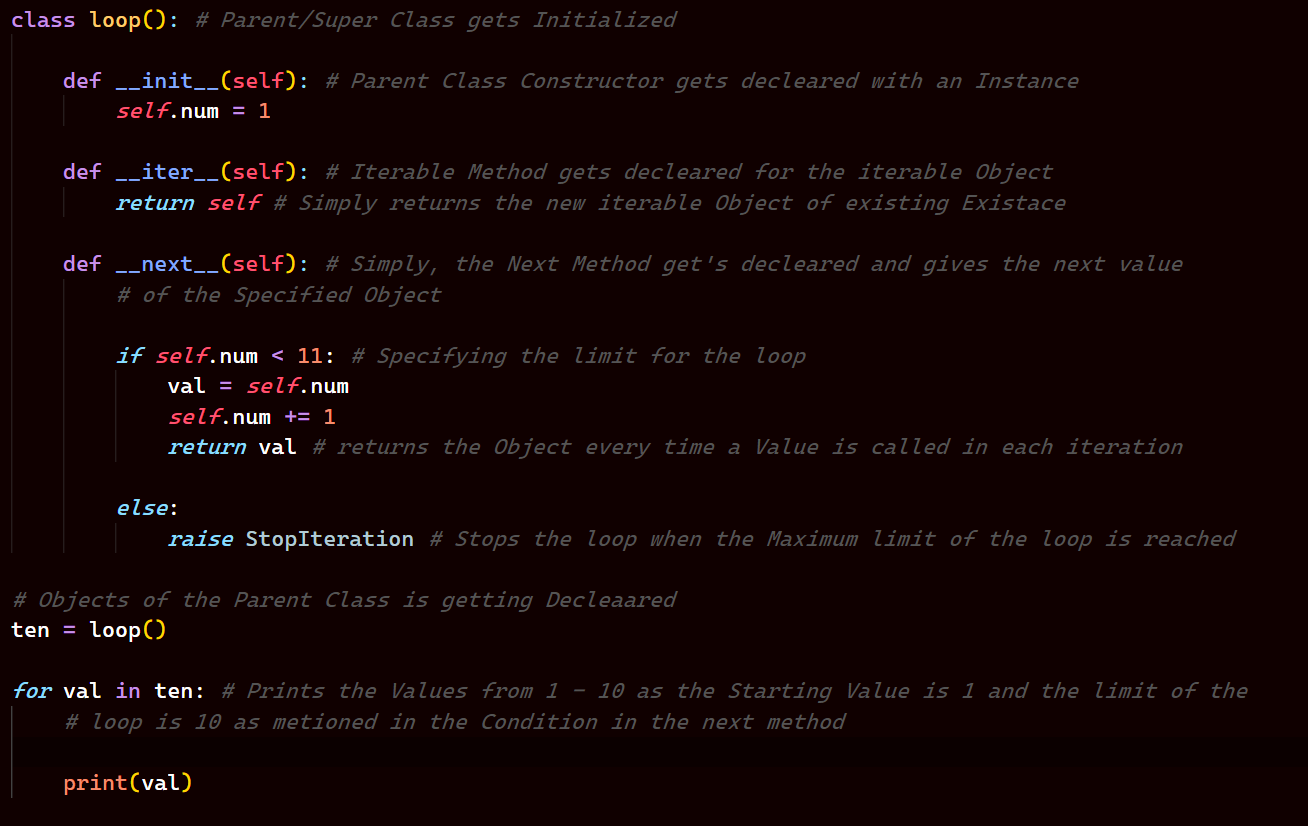
*It also gives the same Output like,*

**

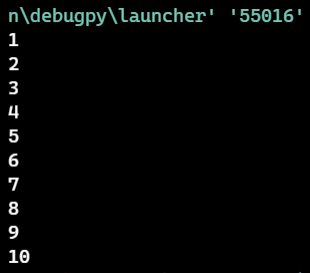
*Ok now we have seen the usage of iterators and it’s method next on built-in Objects.*

***Note:*** *In case of for loop in Python, behind the Scene it calls the iterator and next Function.*

*Now, what we saw was the application of all of these on built-in Objects, now what if we had our own Object means our Own Class, how would we write the Code in that case. Let’s see the Code in Action,*

**

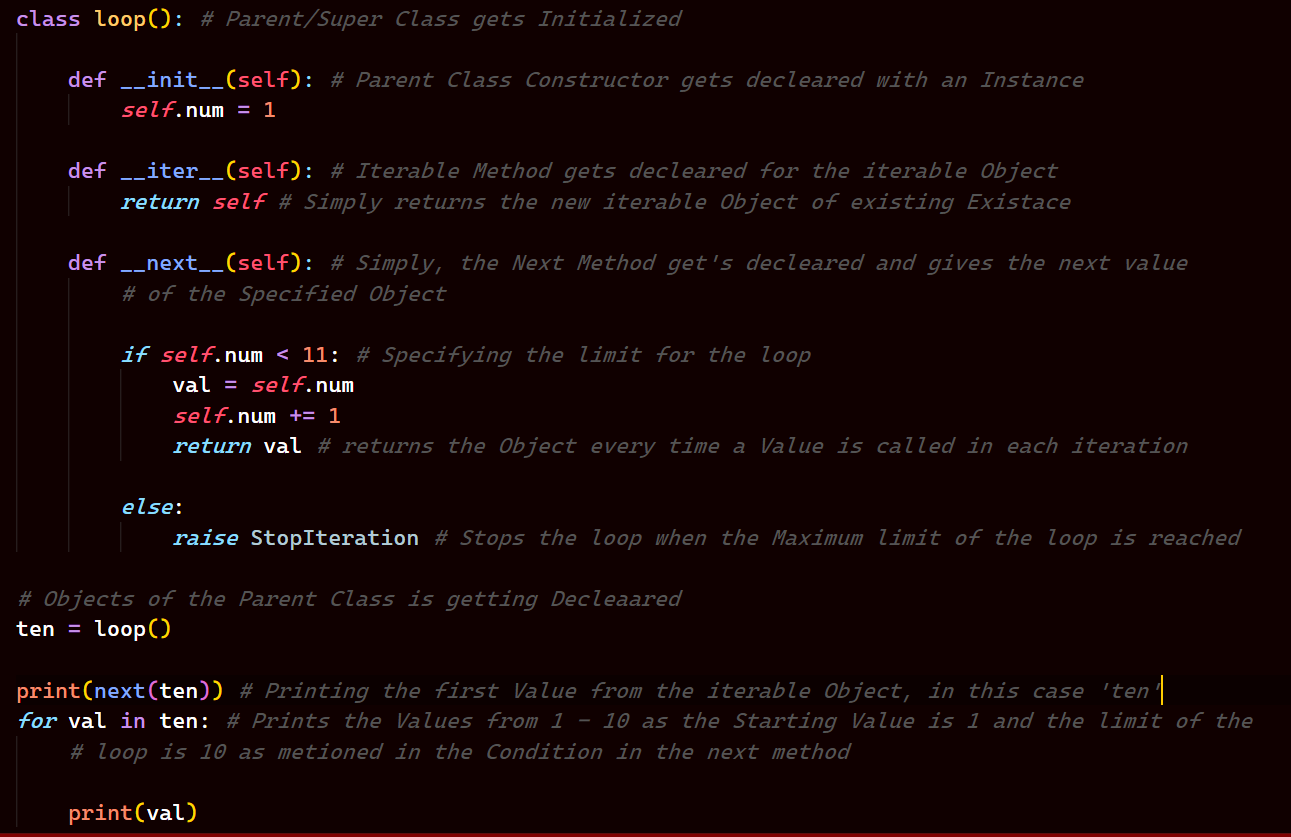
*Output:*

******

***Task: Create your own for loop range function with the help of the Concepts learned.***

*Now, let’s see a scenario and understand a tricky part.*

*Analyze the Code and predict the Output,*

******

*Now, here as you can see we have already print the first value of the iterable. Then the Output should give 1, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10.*

*Let’s see the Output,*

******

*Strange, the Output is the same as before, it is because the next function remembers the values it has given currently and if we try to pass the same iterable Object in this case “ten” it will start from where left off, so first we did print 1 and then inside the loop it started from 2 as it remembered that 1 is already being print in the Console from the same iterable Object.*