PYTHON OOPs

Encapsulation : It is the process of restricting access to methods and variables and preventing from the data being modified both accidently and intentionally.

· Python has different levels of restrictions that control the accessing of data.

· Variables and methods can be public , private or protected. These are made by using the underscores symbols.

Public :

· Public variables and methods can be freely modified and run from anywhere, either inside or outside of the class. To create a public variable or method, don't use any underscores.

Private Variables :

· The private variable or method can be accessed from within its own class or object.

· Private variable value cannot be modified from outside of a class.

· Private variables and methods are declared by two underscores.

Example :

```
class Cricket:
    __country = "Australia"
    __name = ""
    def __init__(self):
        self.__country = "England"
        self.__name = "Sachin"
    def details(self):
    print ("player country" + str(self.__country))
sport = Cricket()
```

```
sport.details()

sport.__country = "India"

sport.details()
```

Output :

```
player countryEngland

player countryEngland
```

· In the example we declared two private variable __country and __name which can be used inside the class.

· In example even though we changed the value of country to “india” but the country name will not change this is because the variables exist independently.

· To overcome this drawback we can use an method called set method which set the value to private variable.

Example :

```
class Cricket:

    __country = "Australia"

    __name = ""

    def __init__(self):

        self.__country = "England"

        self.__name = "Sachin"

    def details(self):

        print ("player country" + str(self.__country))

    def setcountry(self,country):

        self.__country = country

sport = Cricket()

sport.details()
```

```
sport.setcountry("India")

sport.details()
```

Output :

```
player countryEngland

player countryIndia
```

· Here we change the value of country to “India” by using a setcountry method.

Private Methods : When a class with some methods was created which contains a private method.

· Object for the class is created to access methods in the class.

· If tried to access private methods it will display an error that the method doesnot exist.

Example :

```
class Software:

    def __init__(self):

        self.__language

    def role(self):

        print ("Programmer")

    def __language(self):

        print ("best language")

it = Software()

it.role()

it.__language()
```

Output :

```
Programmer

Traceback (most recent call last):
```

File "C:\Users\gsanjeevareddy\Desktop\encapsulation.py", line 10, in <module>

it.__language()

AttributeError: 'Software' object has no attribute '__language'.

· It show the error when accessing private method because it cannot be accessed directly from outside the class.

· Python provide a different method that can be used to access private method from outside the class is

objectname._classname.__methodname

Example :

it._Software__language()

Output :

Programmer

best language

Setter and Getter :

· The methods that are used to for interacting with encapsulated variables are by setter and getter.

· Setter method is used to set values to variable and getter is used to retrieve value from the variable.

· These are used because the methods are exist within the class with which accessing private variables is easy when compared to outside class.

Example :

```
class Cricket:
    __country = "Australia"
    __name = ""
    def __init__(self):
        self.__country = "England"
```

```
        self.__name = "Sachin"
    def details(self):
    print ("player country" + str(self.__country))
    def setcountry(self, country):
        self.__country = country
    def getcountry(self):
        return self.__country
sport = Cricket()
sport.details()
sport.setcountry("India")
sport.details()
print (sport.getcountry())
```

Output :

```
 player countryEngland
 player countryIndia
 India
```