PYTHON OOPs

Data Hiding : An object attributes may or may not be visible outside the class definition.

· Attributes with double underscore(__) prefix are not visible or accessed directly to outsiders.

· Python protect these attributes by internally changing the name by including the class name.

Syntax :

object._Classname__attributename

· To access such attributes we have to use attribute name along with class name and object.

Example :

```
class Exponent:
    __a = 4
    def power(self, b):
        self.__a **= b
        print (self.__a)
Obj = Exponent()
Obj.power(2)
Obj.power(5)
print (Obj.__a)
```

Output :

```
16
1048576
Traceback (most recent call last):
File"C:\Users\gsanjeevareddy\Desktop\datahiding.py", line 9, in
<module>
```

```
print (Obj.__a)

AttributeError: 'Exponent' object has no attribute '__a'
```

· Here in the example we gave a attribute with double underscores as __a.

· We used the exponent Class to calculate power of the value.

· We have object as Obj with which we access attribute and calculate power.

· Here when we try to access __a it will show an error as AttributeError that class has no attribute '__a' . This is because it will not be visible outside the class.

· To overcome this method python provide us with different syntax.

· For above example, Obj._Exponent__a is used to access __a attribute. This is because Python internally changes the attribute to include with class name.

```
print (Obj._Exponent__a)
```

· Replacing the print with above code.

Output :

```
16

1048576

1048576
```

· Now it won't display any error and give the value associated with __a attribute.