**Propriety:**

class *MyClass*():

def \_\_init\_\_(*self,myattr*):

self.*priv\_attr* = *myattr*

def *get\_attr*(*self*):

return *self.priv\_attr*

def *set\_attr*(*self,attr*):

self.*priv\_attr = attr*

*attr = property(get\_attr,set\_attr)*

BASH INPUT:

>> obj = MyClass(‘Python’)

>> obj.attr

>> obj.attr = ‘prova’

>>obj.attr

BASH OUTPUT:

‘Python’

‘prova’

This class has 2 methods that are called getter and setter. The getter, that is the *get\_attr()* method, allows us to obtain the *priv\_attr* value without declare it as a public variable. Otherwise we got the setter, that is the *set\_attr* that allows us to set the variable’s value.

We also have the word *property* that allows us to simply call the instance we set the propriety method to automatically recognize if it has to do the set or the get method on that instance.

class *MyClass*():

def \_\_init\_\_(*self,myattr*):

self.\_\_*priv\_attr* = *myattr*

@property #decorator that says that the function that comes right after this declaration is a GETTER

def attr(self):

return self.\_\_priv\_attr

@attr.setter #setter @name.setter

def attr(self,myattr):

self.\_\_priv\_attr = myattr

**Excercize:**

1. Hide “amount” attribute
2. Define a property, using a getter decorator called “AMOUNT”, that returns “amount” attribute
3. Define, via setter decorator, as above called “AMOUNT” that modifies “amount” attribute in these modes:
   1. Invoke withdraw w/ starter amount
   2. Invoke deposit w/ amount requested by setter