Python 3: The Python Environment

Lesson 2, Project 1

Handed in: 3 Jan 2016 08:24:30PM Graded: 5 Jan 2016 12:29:07PM

Here are your instructions:

Create a **Python3\_Homework02** project and assign it to your **Python3\_Homework** working set. In the **Python3\_Homework02/src** folder, create a file named **coconuts.py**, with an inventory class that tracks different types of coconuts from around the world. The different types of coconuts must have these weight attribute values:

|  |  |
| --- | --- |
| **Type** | **Weight** |
| South Asian | 3 |
| Middle Eastern | 2.5 |
| American | 3.5 |

The inventory class must have the following methods:

* **add\_coconut()** accepts a coconut as an argument. Other types throw an AttributeError.
* **total\_weight()** provides the total weight of coconuts.

For this project, you may find the **isinstance** built-in useful.

You must Include a **test\_coconuts.py** program to confirm all the Inventory class methods. The tests must check:

1. That different coconut types each have a different weight.
2. That if a string object is passed to the **Inventory.add\_coconut** method, an AttributeError is thrown.
3. That if 2 South Asian, 1 Middle Eastern, and 3 American coconuts are added to the inventory, the **Inventory.total\_weight()** method returns 19.

When they are working, submit **coconuts.py** and **test\_coconuts.py**.

##### **Your Comment:**

*no comment given*

##### **Items Handed In**

* [Open Project Handed In](https://students.oreillyschool.com/student/project/?/.handin/147-6583-1/com.ost.mboyd.147.6583.1.Python3Homework02.zip)

### **Overall Comments:**

Good job experimenting with deepcopy. So far our coconuts objects are very shallow so in terms of adding weights, even  
references to the same object in memory would be OK.  
  
A class hierarchy one might use:  
  
class Coconut:  
 pass  
  
class SouthAsian(Coconut):  
 weight = 3  
  
class MiddleEastern(Coconut):  
 weight = 2.5  
  
class American(Coconut):  
 weight = 3.5  
  
Then nut = American() gets the weight needed.  
  
Since amt default is 1, and 0 amt makes little sense, I'd have the range() go from 1 to amt + 1 i.e. 1 to amt inclusive:  
  
 def addCoconut(self, coco, amt = 1):  
 """  
 Adds one or more coconuts to the inventory's list.  
 The object passed must be of type "Coconut".  
 If an amount higher than 1 is provided, then copies of the coconut will make up the rest.  
 The coconuts are copied so that they aren't all pointing to the same object, and can have independent properties.  
 """  
 if type(coco) == Coconut:  
 for i in range(1, amt+1):  
 if i == 1:  
 self.inv.append(coco)  
 else:  
 newCoco = deepcopy(coco)  
 self.inv.append(newCoco)  
 else:  
 raise AttributeError("A {0} is not a coconut".format(type(coco)))  
  
-Kirby

### **Grade:**

Great