**The Account Assignment***decorators and properties*

You are to create a file named **Account.py** containing the definition of two classes: **Transaction** and **Account**. You must include some simple doctests for each.

The **Transaction** class that takes an amount, a date, a currency (default **“USD”**—U.S. dollars), a USD conversion rate (default **1**), and a description (default **None**). All of the data attributes must be **private**. Provide the following read-only properties: **amount**, **date**, **currency**, **usd\_conversion\_rate**, **description**, and **usd** (calculated from **amount\*usd\_conversion\_rate**). This class can be implemented in about sixty lines including some simple doctests.

The **Account** class that holds an account number (**account\_number**), an account name (**account\_name**), and a list of **Transactions**. The **account\_number** should be a read-only property; the **account\_name** should be a read-write property with an assertion to ensure that the name is at least four characters long. The class should support the built-in **len()** function (returning the number of transactions), and should provide two calculated read-only properties: **balance**, which should return the account’s balance in USD and **all\_usd**, which should return **True** if all the transactions are in USD and **False** otherwise. Three other methods should be provided: **apply()** to apply (add) a transaction, **save(),** and **load().** The **save()** and **load()** methods should use a binary pickle with the filename being the account number with extension **.acc**; they should save and load a list containing the account number, the name, and transactions list.

This class can be implemented in about ninety lines with some simple doctests that include saving and loading—use code such as   
 **name = os.path.join(tempfile.gettempdir(), account\_name)**

to provide a suitable temporary filename, and make sure you delete the temporary file after the

tests have finished.

Note:

pickle command:

**fh = None**

**try:**

**data = *<whatever you want to pickle>***

**fh = open(*<your file name>*, "wb")**

**pickle.dump(data, fh, pickle.HIGHEST\_PROTOCOL)**

**except (EnvironmentError, pickle.PicklingError) as err:**

**raise SaveError(str(err))**

**finally:**

**if fh is not None:**

**fh.close()**

Obviously, you need to replace ***<whatever you want to pickle>*** and ***<your file name>*** with real strings.

unpickle command:

**fh = None**

**try:**

**fh = open(*<your file name>*, "rb")**

**data = pickle.load(fh)**

**# assert that number stored in the pickle matches self’s number**

**extract name and Transactions**

**except (EnvironmentError, pickle.UnpicklingError) as err:**

**raise LoadError(str(err))**

**finally:**

**if fh is not None:**

**fh.close()**