Lab 15 – Barista OOP

Goals

- Practice with object oriented programming
- Practice creating classes with methods and attributes
- Practice creating class variables and private instance variables
- Using instance variables and their associated methods

Setup

- Create a new Python project
- Use the following naming convention
  `ITP115_l#_lastname_firstname`
  (replace *lastname* with your last/family name, *firstname* with your first name, and # with the lab number)
- Create a Python file named **`Coffee`**
- Create a Python file named **`Barista`**
- Your new file must begin with comments in the following format *(replace the name and email with your actual information):*

```
# Name
# ITP 115, Spring 2017
# Lab L#
# USC email
```

Requirements

In this lab, you will create two classes to model a coffee shop interaction with a barista. You will be defining two classes, **`Coffee`** and **`Barista`**. The Barista class will contain an attribute which is a list of Coffee objects.

Provided for you *(do not modify)*

- **`main`** is completed for you already in a file named **`CoffeeShopProgram.py`**. Be sure to place the file in the same directory as your other two files. Since it is already defined, your class methods need to be correctly defined and the exact naming conventions of the class methods must be used in order for the correct output to occur. Here is a brief description of what main does:
  - Create a **`Barista`** object.
  - Take coffee orders from the user as long as the barista is still accepting orders. This is accomplished by using the **`Barista`** class method **`isAcceptingOrders`**.
  - The barista "makes" and serves all of the drinks after accepting 5 orders using the **`Barista`** class method **`makeDrinks`**.

- Ask the user if they would like to continue.

What you need to write

- **`Coffee`** Class, define this class in a file called **`Coffee.py`**
  - Class variable:
    - Create a class variable called **`statuses`** which is a **`list`** containing the strings "ordered" and "completed". You will use this class variable in the string method as described below.
  - Instance attributes/variables (to be assigned in **`__init__`**). Make all of these **private**:
    - **`size`**: the size of the drink (small, medium, large)
    - **`desc`**: description of the drink (a string)
    - **`customer`**: the name of the customer who ordered the drink (a string)
    - **`status`**: 0 if the drink has not been made, 1 if the drink is completed. When new coffee objects are created, the status will be set to 0 initially
  - Constructor Method:
    - **`__init__`**
      - Inputs (3): size of the drink, a description of the drink, and the name of the customer who ordered the drink
      - Return: none
      - Set the size, description, and customer attributes of the coffee object. Set the status attribute to 0 (representing an uncompleted drink)
  - Instance Methods:
    - **`setCompleted`**
      - Input: none
      - Return: none
      - Set the status attribute to 1, indicating that the drink is completed
    - **`__str__`**
      - Input: none
      - Return: a string containing information about the coffee object
      - The message should be a nicely formatted string with information about the coffee object's size, description, the customer who ordered it, and status. Be sure you use the class variable, **`statuses`**, in the message (use in combination with **`status`** to get the correct string version of the status).

- **`Barista`** Class. Define this class in a new file called **`Barista.py`**
  - Class variable:
    - Create a class variable called **`maxOrders`** and set its value to 5. This represents the maximum number of drinks the barista can keep track of at a time.
  - Instance attributes/variables (to be assigned in **`__init__`**). Make all of these **private**:
    - **`name`**: a string representing the barista's name
    - **`orders`**: a **`list`** of **`Coffee`** objects representing the drinks the barista has to make.
  - Constructor Method:
    - **`__init__`**
      - Input: (1) the name of the barista
      - Return: none
      - Set the name of the barista. Set the orders to an empty list.
  - Instance Methods:
    - **`takeOrder`**
      - Input: none
      - Return: none
      - Take a coffee order by asking the user to input their name, the size of the drink, and the type of the drink that they are ordering. Use these three inputs to create a new **`Coffee`** object and add it to the list of orders. Print out the created drink.
    - **`isAcceptingOrders`**
      - Input: none
      - Return: **`Boolean`**
      - Return **`True`** if the barista can still take more orders (i.e. the list of **`coffee`** objects is less than **`maxOrders`**). Return **`False`** otherwise.
    - **`makeDrinks`**
      - Input: none
      - Return: none
      - Go through the list of orders and "make" all of the drinks by using the **`Coffee`** class's **`setCompleted`** method. Print out each completed order.
      - After making all of the drinks, reset the barista's list of orders back to an empty list.
    - **`__str__`**
      - Input: none
      - Return: a string containing a message with the barista's name

Sample Output

```
Please enter your name:
> Trina

~ Welcome to the Coffee Shop! ~

Hello, my name is Trina, and I am your barista.
What drink do you want? flat white
What size would you like? small
What is your name? Tiffany
small flat white for Tiffany (ordered)

Hello, my name is Trina, and I am your barista.
What drink do you want? chai latte
What size would you like? medium
What is your name? Huy
medium chai latte for Huy (ordered)

Hello, my name is Trina, and I am your barista.
What drink do you want? macchiato
What size would you like? medium
What is your name? Michael
medium macchiato for Michael (ordered)

Hello, my name is Trina, and I am your barista.
What drink do you want? latte
What size would you like? large
What is your name? Josie
large latte for Josie (ordered)

Hello, my name is Trina, and I am your barista.
What drink do you want? coffee
What size would you like? small
What is your name? Brandon
small coffee for Brandon (ordered)
```

```
Here are the completed orders:
      small flat white for Tiffany (completed)
      medium chai latte for Huy (completed)
      medium macchiato for Michael (completed)
      large latte for Josie (completed)
      small coffee for Brandon (completed)
Would you like to continue? (y/n)
> n
Goodbye, please come again!
```

Deliverables and Submission Instructions

- A compressed folder (zip file) containing you Python code. This can be done by:
    a. Windows *(you must find the folder on your computer—this can't be done within PyCharm):*
        i. Select your lab folder
        ii. Right click
        iii. Send to ->
        iv. Compressed (zipped) folder
        v. Rename this folder with the following name:
            `ITP115_l#_lastname_firstname`
            (replace # with this assignment number)
        vi. Submit this zipped folder through Blackboard
    b. OSX *(you must find the folder on your computer—this can't be done within PyCharm)*:
        i. Select your lab folder
        ii. Right click
        iii. Compress 1 item
        iv. Rename this folder with the following name:
            `ITP115_l#_lastname_firstname`
            (replace # with this assignment number)
        v. Submit this zipped folder through Blackboard