Self-Evaluation for Lab 3 – Classes

Your name:	Patryk Kostek
Date:	4/27/2023

Instructions

After completing this evaluation you will submit this form along with screen shots of each of your applications running on your machine. Your source code will be submitted by providing the url for your github repository. You may make corrections to your work and submit an updated evaluation in your final version of the lab.

General comments and notes:

This lab was very fun. I was looking forward to learning about classes as in it seems like a very important concept in programing.

One thing that you learned from completing the lab:

I learned the proper way to implement a class in python. This is very helpful to keep code clean in a multi-file format. This is very applicable to the real word as most examples of code I have seen of completed applications has a class implemented.

Programming style for all programs	
Is proper indentation used?	Yes
Are comments used appropriately?	No
Do variable names use camelCase or snake_case ?	camelCase
Do function/method names use camelCase or snake_case ? Do methods pass parameters and return values appropriately?	camelCase

Professional development tools and techniques	
Created a private git repository for the lab?	Yes
Committed at least once for each problem?	Yes
Pushed to and pulled from the remote repository at regular intervals?	Yes
Url for github repo: https://github.com/LCC-CIT-Programming-CS162P/classes_lab-Perrtyk	

Customer Class – Problem 1	
Completed Customer Class? • Instance variables are camelCase or snake_case and private? • Implements all methods in the specification? • Implements both getters and setters? • Method names are camelCase or snake_case? • Methods are public? • Completed Customer Tests? Tests all methods in the class? Screen shot is included?	Yes Yes Yes camelCase No, they are private Yes

```
Testing constructor with parameters. Expect customer Patryk Kostek:
Customer(First Name: Patryk, Last Name: Kostek, email: test@aol.com)

Testing getters.  Expect individual attributes for customer Patryk.
First Name: Patryk, Last Name: Kostek, Email: test@aol.com.

Testing property setters. Expect individual changes to customer from Patryk to John.
Customer(First Name: John, Last Name: Mckinney, email: john.mckinney@company.com)

Error: An exception was thrown setting First Name to a integer.
First Name cannot be blank. <class 'int'> 123
Error: An exception was thrown setting Last Name to a integer.
Last Name cannot be blank. <class 'int'> 123
Error: An exception was thrown setting email to a blank string.
Email cannot be blank. <class 'str'>

An attribute error was throws in testEncapsultion
'Customer' object has no attribute '__firstName'

Process finished with exit code 0
```

Card Class – Problem 2	
Completed Card Class? • Instance variables are camelCase or snake_case and private? • Implements all methods in the specification? • Implements both getters and setters where appropriate? Setters validate value and data that it out of range throws an exception? • Method names are camelCase or snake_case? • Methods are public? • Completed Card Tests? Tests all methods in the class? Setter tests validation and exception? Screen shot is included?	camelCase Yes Yes camelCase No, methods are private Yes

```
---Testing Constructor---
Expect 2 of Diamonds.
 Card(2 of Diamonds)

---Testing Property Getters---
Expect individual attributes for card 2 of diamonds:
Suit: 2, Value: 2.

---Testing Property Setters---
Testing property setters. Expect individual attributes change from 2 of diamonds to 3 of hearts.
Card(3 of Hearts)

---Testing Property Setters With Validation---
Expect Error: String
Error: An exception was thrown setting card value to string.
Value must be an integer between 1 and 13.
Expect Error: Not in Range
Error: An exception was thrown setting suit to negative integer.
Suit must be an integer between 1 and 4.

---Testing Encapsulation---
Expect Error: Attribute
Error: An attribute error was throws in testEncapsultion
'Card' object has no attribute '__suit'

Testing isAce. Expect True: True
Testing isAce. Expect False: False

Testing isBlack. Expect True: True
Testing isBlack. Expect False: False

Testing isClub. Expect True: True
Testing isClub. Expect False: False

Testing isDiamond. Expect True: True
Testing isDiamond. Expect False: False

Testing isFaceCard. Expect True: True
Testing isFaceCard. Expect False: False

Testing isHeart. Expect True: True
Testing isHeart. Expect True: False

Testing isRed. Expect True: True
Testing isRed. Expect False: False

Testing hasMatchValue: Expect Card(2,3) and Card(4,3) to be True: True
Testing hasMatchValue: Expect Card(4,3) and Card(4,4) to be False: False

Testing hasMatchingSuit: Expect Card(2,5) and Card(2,9) to be True: True
Testing hasMatchingSuit: Expect Card(2,9) and Card(4,4) to be False: False
```