Name: Jean-Erick Voigt

Date: 9 September 2017

Current Module: OOP Python

Project Name: Bank\_of\_Nerds

Project Goals:

Project was designed to evaluate our ability to use classes and class objects in order to simulate a bank that has accounts. We had to have a menu that allowed customers to be able to at least have three accounts. A checking, savings and IRA. The customers were then supposed to be able to withdraw, deposit, and exchange currency for these accounts at a bare minimum.

Considerations:

<A bulleted list of considerations and conditions which affected the design of your project>

* stdin
* Class objects
* Importing classes from different modules
* Dictionaries for customer accounts
* List of customers
* List of customer accounts
* Navigation of session to allow customers to do a multitude of transactions before they exit

Initial Design:

This program was split into multiple class modules and a driver file. The class objects that this project utilized were account, bank, customer. Since everything was done between the bank and the user, the bank object was able to interface with the customers and accounts from within its class. The mainloop function encompassed the main program. This mainloop was sandwiched in a try and except block in order to account for keyboard interrupts and value errors.

Data Flow:

Initially, The user comes to a home screen where he is prompted to enter if he is an existing customer, new customer or the option to exit. If new customer is selected then the user is prompted to enter his name which will then be stored as a customer class object. This name is checked for errors such as being a numeric name, or being too short. If these errors are made the user is prompted to enter his name again. If the user enters existing customer then he is prompted to enter his name once again. This name is compared to the set of the customer list. This is to replicate uniqueness in the customer list of a bank. If I had more time I would give each user a userid as well and use the user id for storing. If the user chooses to exit the program gracefully exits using sys.exit(). If the user selects 22, which is not listed because it is for business purposes, the screen will display all customers, their accounts and the amount in dollars that they have in those accounts. It is a very elementary way of attempting to implement a backdoor in the program.

The user then is shown the main menu choice window which allows the user to add accounts, withdraw money, deposit money, go back to the other menu, and end session. The answer the user selects will be stored in a variable and executed in the menu execution function. If the user enters a dollar amount to deposit or withdraw that does not conform with the standard then they will be forced to reenter. The standard is either no decimal point which will be assumed to have no cents or one decimal point and rounded to the hundredth place.

Communication Protocol:

<Discuss any custom communication protocol you have used including the various request and responses that may be sent and received.>

Potential Pitfalls:

The pitfalls that I faced in this program were probably some error checking that I missed implementing. Also, structurally, pep8 and doc strings posed a problem, partially due to the fact that I am still getting accustomed to doing this. Also, I wanted to ensure that the “meat and bones” of the project were already implemented before I began working on those things.

Test Plan:

User Test:

I attempted to enter names that were too short. Names that were all numbers. I also entered names in existing customers that didn’t exist. All these tests produced the desired results. Regarding money, I entered money that did not meet the standard of how money should be entered. I tried to withdraw more money than the account had. These tests also passed.

Conclusion:

This projects biggest issue was the small things that had to be done because there was a lot of coding that was done. For example, the pep8 and doc strings would have taken me a good portion of time just doing that and that is not writing new code. Besides that, working with classes made a lot of things easier, however, the classes had to be made adequately in order to function as desired. The suggestion to use the \_\_str\_\_ method definitely made my code cleaner and easier to read in my opinion.