CSCI E-97
Fall 2019
Assignment 1
Due: September 18, 2019
Evelyn Taylor-McGregor

Design changes

- The design document specifies that all accounts should be initialized with a balance of 0. I created a second constructor that takes balance as an argument. I used this to create the master account and to create new copies of accounts when creating new blocks that were initialized with each account's balance after the last block.

- Similarly, I created two constructors for the `CommandProcessorException`, one with line number and one without. I found it simplest to add line number within the `ProcessCommandFile`. However, I also wanted to throw a `CommandProcessorException` from the `ProcessCommand` method that does not have line number within its scope.

- The largest deviation from the design documentation that I made was in regard to managing pending transactions within a block.

  The main constraint I was grappling with were that the `accountBalanceMap` should only contain committed transactions, meaning that until the block is finalized, the `accountBalanceMap` should reflect committed transactions from the previous block. However, as transactions are validated and added to the block there needs to be some way to maintain the "pending" balance for each account, and to keep track of any accounts created while the block is unfinalized.

  To accomplish this I created a `pendingAccountBalancesMap`, which is a map of account IDs to accounts, like the `accountBalanceMap`. This map is used to validate all transactions while the block is filling up, and to check whether an account has been created previously. When the block is ready to be finalized, I set the `accountBalanceMap` equal to the `pendingAccountBalancesMap`.

  I chose to make this change because it made sense to me to have the block manage the accounts, so it also made sense for the blocks to manage the running, uncommitted balance of each account.

Design Feedback

1. **Did the design document help with the implementation?**

Yes.

In particular, I found that the design document helped a lot with the first 75% of the implementation, and less so with the last 25%. For the parts I found trickier, I did not find that the design document clarified any issues for me. If I had created the design document myself, I would have thought through the trickier parts in advance and the design document would remind me of the decisions I'd made. Referring to a design document that someone else wrote was harder; if I was working off this design document with a coworker, I would have set up a meeting to discuss certain elements of the design in more detail (the `accountBalanceMap`, for example).

2. **How could the design have been better, clearer, or made the implementation easier?**

I found the class diagram and the class dictionaries the most useful. The high-level explanations of the design could have been more structured to make them easier to internalize. I found myself reading and re-reading large portions of the document without seeing the connection to the implementation. To avoid re-reading, I made a list of specific requirements based on the text in the design document that I could check-off on a list – this worked well for me.