Practice Python Exam : Banking System

This assignment is entirely optional and designed to give you practice writing code and applying lessons and topics for the Python exam

This practice exam deals with the following topics:

- dictionaries
- file i/o
- unit testing

The Assignment

This Python exam will involve implementing a bank program that manages bank accounts and allows for deposits, withdrawals, purchases, sorting bank accounts based on different criteria, and exporting bank statements.

The program will initially load a list of accounts from a .txt file, and deposits and withdrawals from additional .csv files. Then it will parse and combine all of the data and store it in a dictionary.

Steps

1. Complete the required functions
    a. Implement all of the functions defined in ***bank_accounts.py***
        i. Docstrings have already been provided
        ii. You can create any number of helper functions (with docstrings)
        iii. The *main* function has already been implemented for you
        iv. Add brief comments to all non-trivial lines of code
    b. Pass all of the provided tests in ***bank_accounts_tests.py***
        i. Write additional test cases to each testing function where noted
2. Make sure your program and the testing file run without errors!

Unit Testing

To test your code, we have provided you with ALL of the unit tests for this assignment in

bank_accounts_tests_full.py.

Evaluation

1. Does your program work as expected?
    a. Can you make a deposit to or withdrawal from one of the accounts in the .txt file? Can you make multiple purchases at once? Can you export a correctly formatted bank statement? Can you sort all bank accounts based on user provided criteria?
    b. Does your program print clear and useful error messages when applicable? For example, if you try to withdraw from a non-existent account.
2. Did you implement the functions correctly?
    a. Does your program successfully load and parse the 3 files: accounts, deposits, and withdrawals, and store all the data in a dictionary database?
    b. Are you reusing functions in multiple places in your program? For example, the "purchase" function can use the "withdraw" function. Several functions can use the "get_account_info" function.
3. Unit testing
    a. Did you pass all of the provided unit tests?
    b. Did you write at least 2 additional test cases for each function, where noted?
    c. Do you test both typical examples and edge cases?
    d. Does your program pass all of your own tests?
4. Style
    a. Appropriate naming of variables
    b. Naming of helper functions (with docstrings)
    c. Clear comments in your code.