CASH CODING EXERCISE V5

Guidelines

Use your preferred object-oriented or functional language. Build a simple bank by writing code with tests that meets the requirements along with a README. We expect you to spend between 2 to 3 hours on this exercise. If you're spending longer than that you are probably over-engineering the solution. We review these anonymously to reduce bias; please **don't put your name in the code or host it in a public place**.

Requirements:

- Deposit, withdraw and maintain a balance for multiple customers
- Return a customer's balance and the bank's total balance
- Prevent customers from withdrawing more money than they have in their account

An example test scenario

When Alice deposits $30 and withdraws $20
Then Alice's balance will be $10 and the bank's balance will be $10
And Alice will be prevented from withdrawing $11 to prevent her balance going negative

What we're looking for:

- A concise and considered **README** that provides instructions and communicates your assumptions, design decisions and trade-offs. No novels please!
- **Tests** with good coverage of the app's behaviour that act as documentation for the code and includes the above example test scenario
- Your **analysis** of the above requirements and edge cases (please do not include anything outside these requirements!)
- A simple and extensible **technical design** that is **maintainable** and easily understandable by other engineers
- **Code fluency** and an idiomatic approach in your chosen coding language (any language is fine!)

What we're not looking for:

You don't need any way to interact with your code other than through tests. To help you focus on what we're interested in, don't spend time on:

- Code not needed to implement the requirements
- Command line tool
- User interface
- Any kind of server or client
- Persistence
- Multithreaded concurrency