Code Test: Banking

About This Test

The purpose of this code test is to show us your skills in:
- Code structure and OOP concepts
- Reliability and Testing
- Problem solving

Please keep these aspects in mind as you develop your solution. Also, your chosen implementation doesn't necessarily have to be the best you can think of, but one that you can implement in the allocated time.

Instructions

- The time for completing this test is 3 (three) hours
- The language used should be Ruby, unless previously agreed otherwise
- The resulting code should be directly executable
- All code should be tested
- All needed scripts for compiling, testing, deploying and executing should be included, accompanied by Readme file explaining their usage

Background

The software you write in this test will be used for banks. Banks have accounts. Accounts hold money. Transfers can be made between accounts. Banks store the history of transfers.

There can be two types of transfers:
- Intra-bank transfers, between accounts of the same bank. They don't have commissions, they don't have limits and they always succeed.
- Inter-bank transfers, between accounts of different banks. They have 5€ commissions, they have a limit of 1000€ per transfer. And they have a 30% chance of failure.

Part 1 (Recommended time: 45 minutes)

Define a set of data structures to accurately reflect banks, accounts and transfers. Make sure that new types of accounts and transfers can be added to the bank with minimal effort.

Part 2 (Recommended time: 135 minutes)

Create an initial system consisting of two banks with at least two accounts each. Create the different types of transfers for intra-bank and inter-bank transfers.

Jim has an account on the first bank and Emma has an account on the second bank. Jim owes Emma 20000€. Emma is already a bit angry, because she did not get the money although Jim told her that he already sent it.

Help Jim send his money by developing a transfer agent. This agent assures that everybody gets their money. When the agent receives an order to transfer money from account A to account B, he issues transfers considering commissions, transfer limits and possibility of transfer failures.

The banks keep a list of all transfers and can be asked for the history of transfers.

Questions:

Please also supply your answer to the following questions:

- How would you improve your solution?
- How would you adapt your solution if transfers are not instantaneous?