Michael Calabrese & Mikhail Prigozhiy

PA5: Multithreaded Bank System

For our project, we designed a bank system that resides completely in **shared memory** using multiprogramming (the extra credit options). We also implemented try-lock into our code (also extra credit). Our bank has the ability to **open** accounts, **start** an existing account, **credit** an account, **debit** an account, show the current **balance**, **finish** an account and **exit** from the bank completely.

Up to 20 clients can connect to our bank at a time. We achieved this by creating a bank struct that then holds an array of 20 account structs. Clients connect to the bank server with the command **./client [hostname]**, where the hostname can be any IP address such as 127.0.0.1 or an address of another computer. **Please keep input commands in lowercase** (thank you). Both server and client do not need to be running on the same machine in order to connect properly.

When a client connects to the bank, they will be greeted with commands like the ones described above. Because we decided to implement the extra credit shared memory and multiprogramming portion into our code, even when the bank server is shut down and is not running, information of all the accounts remains in tact, including balances and usernames and is printed on the server side every 20 seconds. Additionally, if the server is shut down, all clients connected to it will be immediately told that the connection has been closed and exit.

Once a session is started using **start [username]** as input, the client must **finish** their session when they are done using the bank services. If another client tries to start a username that is currently in session, they will be held on a waiting list until that session is finished. By placing try-locks on certain sessions, we were able to achieve locking certain instances of bank sessions once they are started so no other user can open a currently active session.

We put a great deal of effort into our bank server. Thank you testing out our bank!