Bank Project

by Wael Ayadi and Parth Mehrotra

Server Structure:

When main() is executed, the program initializes all of its variables and binds its server socket to port 8989. Then main() begins to accept client connections. Once a connection is made with a client, main() creates a new thread for communicating with that client and adds that thread to a linked-list of threads. Then, main() returns to listening for connections.

Each communication thread waits for its client to send a message. Once it receives a command from the client, the communication thread parses it and performs an action based on what command it received. All the error-checking is done in this thread. If "exit" is called, that respective communication thread sends a message to the client and then closes. When the communication thread closes, it is removed from the linked-list of threads.

Three operations are mutexed in this design: opening an account, starting an account, and changing an account's balance.

There is also a garbage collector thread running in the background. This thread continually scans the linked-list of threads, while attempting to find any closed connections. Once it finds one, it joins on it and frees any memory that it may have allocated. This thread is engineered such that it does not require a mutex.

This setup also implements a global server printout by using alarm() and a SIGALRM signal handler.

Closing a client, either by the "exit" command or by ctrl+C, will cause its respective thread on the server to close. Shortly afterwards, the thread garbage collector will find it and free it.

Shutting down the server using ctrl+C will invoke a SIGINT signal handler, which sends a message to all active client communications, notifying them of the server's shutting down. Then, the server will proceed to free all its allocated resources and join on all of its currently active threads before closing.