++++++++++++++++++++++++++++++++
+ Multithreaded Bank Server +
+ Readme +
+ Jessica Ackerman +
+ Steven Barrios +
++++++++++++++++++++++++++++++++

DESCRIPTION:
Our program was designed to implement a multithreaded bank server. We began by creating a multithreaded server and client programs. The server created new threads from session acceptor thread which handled the client threads. The client handler program was used to handle the clients requests about their accounts which included opening, deleting, credit, debit, balance finishing and exiting the session with the bank. The accounts were managed using a linked list data structure.

Server:
Our server implemented a multithreaded approach by using a while loop as the server was listening for client programs. In the createSessionAcceptorThread the socket was created, bound to the port and then while the number of connections was less than 20, and while the client was accepting threads the server would create new threads that with would pass to the session handler function. From there the session handler took in commands from the client and performed actions on the client's account based on those commands. Mutexes were placed around the resources while the it was processing a transaction and then released again. Once the client had finished their session, the client was disconnected and the number of connection was decreased. A signal flag was used to identify if a client was insession and from there what commands they were allowed to use based on that. The timer was implemented by using a thread that would put a mutex on all the threads except the timer thread, it would print the list and then sleep again for 20 seconds, and then release the mutex.

Client:
The client program was a standard c client program that connected to the server via a ip address and port number. All multithreaded, mutexes and timers were handled in the server program.

Accounts:
The accounts were implemented using a linked list. Each node held the account's name and their balance. Accounts were added and deleted by iterating through the linked list to find the user and adding or deleting accordingly. Credit and debit were also performed by iterating through the list and performing the transaction on the node's balance.

Notes:
The ip address and port number and hard coded into the client and the server.