Server Side:
Two signal handlers:

1. Sigc- this catches the signal from ctrl-c. It changes global variables that keep the loops in my client threads running so that they will exit. It also turns off my timer. There is a global linked list of all my pthreads and corresponding socket descriptors. I use that to send all my clients a shut down message, close the socket, and join the pthread. I also free here.
2. Sighandler-- this catches the signal my timer gives off. It then locks the semaphore and traverse through my global linked list of accounts, printing out the account name, balance, and whether it is in session or not.

Void* functions we pass into pthread:

1. connectionAcceptor
    a. This loop will run until a ctrl-c is caught. It accepts new connections, then spawns a thread clienthandler, sending it the sockfd as a parameter.
    b. It adds this thread to the global LL of pthreads
2. Clienthandler
    a. This reads in commands from the client with the sockfd passed in as a parameter.
        i. Create-- check the global LL to see is that account already exists. If it does, send an error message back to client, otherwise create and add the account
        ii. Serve--check if account exists, if not send error message back to client. If it does, check if it is already in session, if so send error message to clielnt, otherwise start serving it
        iii. Deposit--adds to balance if possible
        iv. Withdraw--subtracts from balance if possible
        v. Query--sends back balance if possible
        vi. End--ends service session if in session
        vii. Quit--closes socket, if serving, puts out of service.

Main:

1. Sets up sockets,
2. Sets up timer
3. Set up threads
4. Joins at end

Client side:
Two signal handlers are present:

1) sigc - Ctrl-C to shutdown client after waiting 5 seconds for last command to complete
2) Connecthandler - will try to connect to host server, will wait 3 seconds after trying again

Two void star functions to which we pass inside pthread create

1) cmdread to read in what the user inputs from STDIN
    a) Counts number of spaces to make sure user does not input invalid command

    b) Make sure the double value passed in after withdraw and deposit is a valid number
    c) Compares first token to make sure command is valid
    d) Will sleep before sending command for 2 second to simulate throttling
    e) If command is valid, function will send buffer to server
    f) If creating or serving, cannot input alphanumeric, only alphabetical accounts allowed!
    g) Else client will display an error message for invalid message
2) serverRec will receive a buffer from server for acknowledgement
3) In case of server shutdown, will compare message to proceed as planned
4) Will also catch quit received from server to make sure disconnect has gone off without error

In main we have basic commands such as setting up the client, setting up the parameters for the client

Will create two threads on successful connection and will join when user inputs quit.