- High level server design
    - Server
        - Manages information about each client using a hashmap or some other data structure, potentially with structs
        - Runs a goroutine each time a new client connects so it can communicate with it and send music data
        - Will need to store the server and listener ports of the clients, as well as the station they are currently on
        - For each station, we will need to keep track of the current bytes being sent out from the song to all of the clients currently on that station. Will also need to keep rate relatively constant
        - We will likely use channels because there is a bijection between threads and clients, so each thread will mostly need to change the information for its own client in the server, and there will not be much, if any, overlap
        - When a client changes stations, we will need to update the station listened to by that client. We will access this data structure when we send information to the listener, so that changes in the station take effect immediately.
            - There should not be issues with synchronization because the thread will only change the data for its corresponding client, and no other threads will have reason to change that data.
    - Client
        - Control
            - Communicates with server
        - Listener
            - Receives information from server based on control