おおまかな流れの説明

起動はコマンドラインから
./watercooler [-p ポート番号(省略時は50001)]
以降「課題5 TCP/UDP通信とブロードキャスト」の仕様に基づく.

クライアントとして動作する場合には1つのスレッドですべての動作を行う.
サーバとして動作する場合には「サーバの動作」の{1}, {2, 3}, {4, 5, 6}をそれぞれ1スレッドとして扱う.
ここで, {4,5,6}はクライアントと同じ動作である.

主な関数の機能と入出力仕様

関数の名前はここでの説明のために仮につけたものである.
入出力仕様で明記していない部分は無視できるものと考えた.

main
メイン関数.
コマンドライン引数の解釈, 「起動時の動作」,
client_main関数あるいはserver_main関数の呼び出しを行う.

client_main
クライアントとして動作するときに呼び出される.
サーバの情報をstruct sockaddr_in型の引数として受け取る.
初期化を行った後, tcp_client関数を呼び出す.
この初期化に, ユーザ名のキーボードからの読み取りや,
「JOIN username」のサーバへの送信も入っている.

server_main
サーバとして動作するときに呼び出される.
サーバの情報をstruct sockaddr_in型の引数として受け取る.
初期化を行った後, tcp_server関数, udp_listen関数, tcp_client関数を
それぞれスレッドとして呼び出す.
このとき, それぞれをスレッドとして呼び出すことに加え,
1. tcpサーバの初期化
2. tcp_server関数の呼び出し
3. udpサーバの初期化
4. udp_listen関数の呼び出し
5. tcpクライアントの初期化
6. tcp_client関数の呼び出し
の順で処理を行うことにより, 複数のクライアントが不定期にデータを送信しても
対応が可能になると考える.

tcp_server
スレッドとして呼び出される. 監視するtcpポートのソケットを引数として受け取る.
クライアントからJOIN, POST,QUITのメッセージを受け取り, 適切に処理する.
各クライアントの情報の管理もこの関数が行う.

udp_listen
スレッドとして呼び出される. 監視するudpポートのソケットを引数として受け取る.
クライアントからHELOのメッセージを受け取り, 適切に処理する.

tcp_client
サーバに送信するのに使うソケットを引数として受け取る.
サーバからMESGのメッセージを受け取り, 適切に処理する.
キーボードからの入力を受け取り, 適切に処理する.

データ構造に関する説明

クライアントの情報の管理には双方向循環リストを用いる.
先頭にはダミー要素を配置する.

ソースファイル群の構成に関する説明

server_main関数, tcp_server関数, udp_listen関数をまとめて1ファイル, そのヘッダが1ファイル.
client_main関数, tcp_client関数をまとめて1ファイル, そのヘッダが1ファイル.
main関数を1ファイル.
クライアントの情報を保持する双方向循環リストの定義に1ファイル, そのヘッダが1ファイル.
複数のファイルから参照する定数や宣言の一部をまとめて1ファイル, そのヘッダが1ファイル.

の構成としたい.