```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <signal.h>
#include <unistd.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <arpa/inet.h>
#include <pthread.h>

#define LENGTH 2048

// Global variables
volatile sig_atomic_t flag = 0;
int sockfd = 0;
char name[32];

void str_overwrite_stdout() {
  printf("%s", "> ");
  fflush(stdout);
}

void str_trim_lf (char* arr, int length) {
  int i;
  for (i = 0; i < length; i++) { // trim \n
    if (arr[i] == '\n') {
      arr[i] = '\0';
      break;
    }
  }
}

void catch_ctrl_c_and_exit(int sig) {
    flag = 1;
}

void send_msg_handler() {
  char message[LENGTH] = {};
    char buffer[LENGTH + 32] = {};

  while(1) {
    str_overwrite_stdout();
    fgets(message, LENGTH, stdin);
    str_trim_lf(message, LENGTH);

    if (strcmp(message, "exit") == 0) {
            break;
    } else {
      sprintf(buffer, "%s: %s\n", name, message);
      send(sockfd, buffer, strlen(buffer), 0);
    }

        bzero(message, LENGTH);
    bzero(buffer, LENGTH + 32);
  }
  catch_ctrl_c_and_exit(2);
}

void recv_msg_handler() {
	char message[LENGTH] = {};
  while (1) {
		int receive = recv(sockfd, message, LENGTH, 0);
    if (receive > 0) {
      printf("%s", message);
      str_overwrite_stdout();
    } else if (receive == 0) {
			break;
    } else {
			// -1
		}
		memset(message, 0, sizeof(message));
  }
}

int main(int argc, char **argv){
	if(argc != 2){
		printf("Usage: %s <port>\n", argv[0]);
		return EXIT_FAILURE;
	}

	char *ip = "127.0.0.1";
	int port = atoi(argv[1]);

	signal(SIGINT, catch_ctrl_c_and_exit);

	printf("Please enter your name: ");
  fgets(name, 32, stdin);
  str_trim_lf(name, strlen(name));


	if (strlen(name) > 32 || strlen(name) < 2){
		printf("Name must be less than 30 and more than 2 characters.\n");
		return EXIT_FAILURE;
	}

	struct sockaddr_in server_addr;

	/* Socket settings */
	sockfd = socket(AF_INET, SOCK_STREAM, 0);
  server_addr.sin_family = AF_INET;
  server_addr.sin_addr.s_addr = inet_addr(ip);
  server_addr.sin_port = htons(port);


  // Connect to Server
  int err = connect(sockfd, (struct sockaddr *)&server_addr, sizeof(server_addr));
  if (err == -1) {
		printf("ERROR: connect\n");
		return EXIT_FAILURE;
	}
```

```
        // Send name
        send(sockfd, name, 32, 0);

        printf("=== WELCOME TO THE CHATROOM ===\n");

        pthread_t send_msg_thread;
    if(pthread_create(&send_msg_thread, NULL, (void *) send_msg_handler, NULL) != 0){
                printf("ERROR: pthread\n");
        return EXIT_FAILURE;
        }

        pthread_t recv_msg_thread;
    if(pthread_create(&recv_msg_thread, NULL, (void *) recv_msg_handler, NULL) != 0){
                printf("ERROR: pthread\n");
                return EXIT_FAILURE;
        }

        while (1){
                if(flag){
                        printf("\nBye\n");
                        break;
        }
        }

        close(sockfd);

        return EXIT_SUCCESS;
}
```