Assignment - 3.2
Name: V. V. R. V. Sagar
Roll No: 24CSM2R25

Question:

a) Client uploads a text file to the server.
b) Server processes the file by removing all non-alphabetic characters (keeping only letters A-Z, a-z)
c) Server shares the modified file with all connected clients.

Program:

ServerFile.c

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

#define MAX_CLIENTS 10
#define BUFFER_SIZE 1024

// Client list and mutex for thread safety
int client_sockets[MAX_CLIENTS];
pthread_mutex_t client_mutex = PTHREAD_MUTEX_INITIALIZER;

// Function to remove non-alphabetic characters
void process_file_content(char *buffer) {
    int write_index = 0;
    for (int read_index = 0; buffer[read_index] != '\0'; read_index++) {
        if ((buffer[read_index] >= 'A' && buffer[read_index] <= 'Z') ||
            (buffer[read_index] >= 'a' && buffer[read_index] <= 'z')) {
            buffer[write_index++] = buffer[read_index];
        }
    }
    buffer[write_index] = '\0';
}

// Broadcast processed file to all clients
```

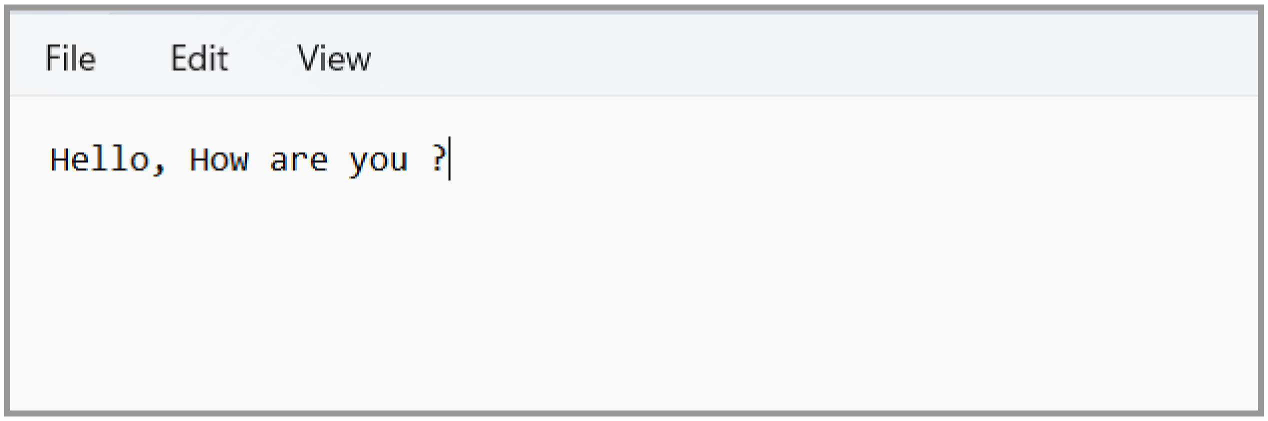
```
void broadcast_file(char *buffer) {
    pthread_mutex_lock(&client_mutex);
    for (int i = 0; i < MAX_CLIENTS; i++) {
        if (client_sockets[i] != 0) {
            send(client_sockets[i], buffer, strlen(buffer), 0);
        }
    }
    pthread_mutex_unlock(&client_mutex);
}

// Client handling thread
void *handle_client(void *socket_desc) {
    int sock = *(int*)socket_desc;
    char buffer[BUFFER_SIZE];
    int read_size;

    while ((read_size = recv(sock, buffer, BUFFER_SIZE, 0)) > 0) {
        buffer[read_size] = '\0';
        process_file_content(buffer);
        broadcast_file(buffer);
    }

    // Remove client socket
    pthread_mutex_lock(&client_mutex);
    for (int i = 0; i < MAX_CLIENTS; i++) {
        if (client_sockets[i] == sock) {
            client_sockets[i] = 0;
            break;
        }
    }
    pthread_mutex_unlock(&client_mutex);

    close(sock);
    free(socket_desc);
    return NULL;
}

int main() {
    int server_socket, client_socket, *new_sock;
    struct sockaddr_in server_addr, client_addr;
    socklen_t client_addr_len = sizeof(client_addr);
    pthread_t thread_id;

    // Create server socket
    server_socket = socket(AF_INET, SOCK_STREAM, 0);

    server_addr.sin_family = AF_INET;
    server_addr.sin_addr.s_addr = INADDR_ANY;
    server_addr.sin_port = htons(8888);

    // Bind socket
    bind(server_socket, (struct sockaddr *)&server_addr, sizeof(server_addr));
    listen(server_socket, MAX_CLIENTS);

    printf("Server listening on port 8888...\n");

    while (1) {
        client_socket = accept(server_socket, (struct sockaddr *)&client_addr, &client_addr_len);

        // Add to client list
        pthread_mutex_lock(&client_mutex);
        for (int i = 0; i < MAX_CLIENTS; i++) {
            if (client_sockets[i] == 0) {
                client_sockets[i] = client_socket;
                break;
            }
        }
        pthread_mutex_unlock(&client_mutex);

        // Create thread for new client
        new_sock = malloc(sizeof(int));
        *new_sock = client_socket;
        pthread_create(&thread_id, NULL, handle_client, (void*)new_sock);
    }

    return 0;
}
```

ClientFile.c

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

#define BUFFER_SIZE 1024
#define MAX_FILENAME 256

int sock;

void *input_thread(void *arg) {
    char filename[MAX_FILENAME];
    char buffer[BUFFER_SIZE];

    while (1) {
        printf("Enter filename to upload: ");
        fgets(filename, sizeof(filename), stdin);

        // Remove newline
        filename[strcspn(filename, "\n")] = 0;

        // Exit condition
        if (strcmp(filename, "quit") == 0) {
            break;
        }

        // Open file
        FILE *file = fopen(filename, "r");
        if (file == NULL) {
            perror("File open error");
            continue;
        }

        // Read and send file contents
        while (fgets(buffer, BUFFER_SIZE, file) != NULL) {
            if (send(sock, buffer, strlen(buffer), 0) < 0) {
                perror("Send failed");
                break;
            }
        }

        fclose(file);
        printf("File uploaded successfully.\n");
    }

    return NULL;
}

void *receive_thread(void *arg) {
    char buffer[BUFFER_SIZE];
    int bytes_received;

    while (1) {
        bytes_received = recv(sock, buffer, BUFFER_SIZE - 1, 0);
        if (bytes_received <= 0) {
            printf("Server disconnected\n");
            break;
        }

        buffer[bytes_received] = '\0';
        printf("\n\nReceived from server: %s\n", buffer);
    }

    return NULL;
}

int main() {
    pthread_t input_tid, receive_tid;
    struct sockaddr_in server_addr;

    // Create socket
    sock = socket(AF_INET, SOCK_STREAM, 0);
    if (sock < 0) {
        perror("Socket creation failed");
        exit(1);
    }

    server_addr.sin_family = AF_INET;
    server_addr.sin_port = htons(8888);
    server_addr.sin_addr.s_addr = inet_addr("127.0.0.1");

    // Connect to server
    if (connect(sock, (struct sockaddr *)&server_addr, sizeof(server_addr)) < 0) {
        perror("Connection failed");
        exit(1);
    }

    // Create input and receive threads
    pthread_create(&input_tid, NULL, input_thread, NULL);
    pthread_create(&receive_tid, NULL, receive_thread, NULL);

    // Wait for threads to complete
    pthread_join(input_tid, NULL);
    pthread_join(receive_tid, NULL);

    close(sock);
    return 0;
}
```

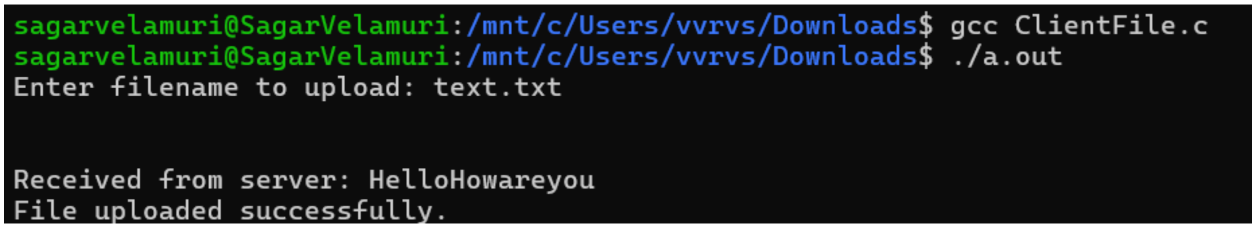
Output:

Text.txt file:



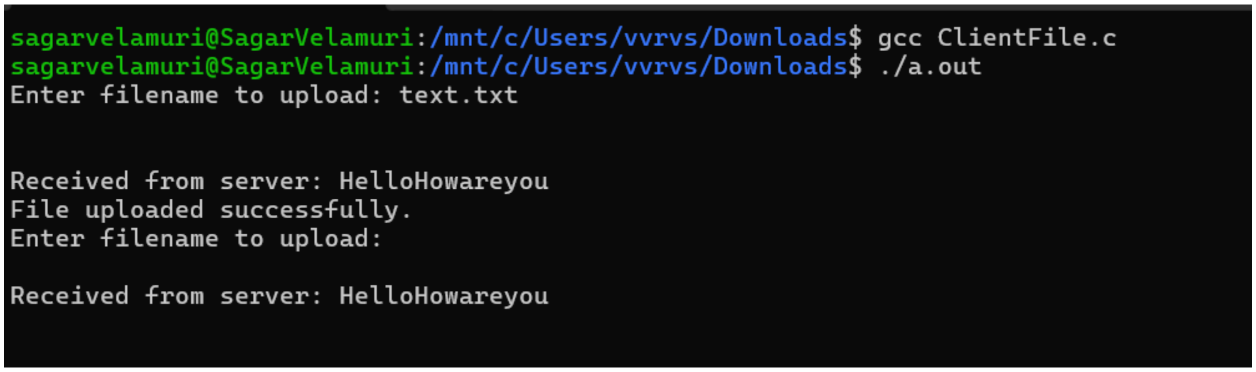
```
Hello, How are you ?
```

Client 1 output:



```
sagarvelamuri@SagarVelamuri:/mnt/c/Users/vvrvs/Downloads$ gcc ClientFile.c
sagarvelamuri@SagarVelamuri:/mnt/c/Users/vvrvs/Downloads$ ./a.out
Enter filename to upload: text.txt


Received from server: HelloHowareyou
File uploaded successfully.
```

Client 2 output:



```
sagarvelamuri@SagarVelamuri:/mnt/c/Users/vvrvs/Downloads$ gcc ClientFile.c
sagarvelamuri@SagarVelamuri:/mnt/c/Users/vvrvs/Downloads$ ./a.out
Enter filename to upload: text.txt


Received from server: HelloHowareyou
File uploaded successfully.
Enter filename to upload:

Received from server: HelloHowareyou
```