21L-1858 Umamah Hussain

**Task 1**

Server:

#include <stdio.h>

#include <string.h>

#include <sys/socket.h>

#include <arpa/inet.h>

#include <unistd.h>

int main(void)

{

int socket\_desc, client\_sock, client\_size;

struct sockaddr\_in server\_addr, client\_addr;

char server\_message[2000], client\_message[2000];

memset(server\_message, '\0', sizeof(server\_message));

memset(client\_message, '\0', sizeof(client\_message));

socket\_desc = socket(AF\_INET, SOCK\_STREAM, 0);

if (socket\_desc < 0)

{

printf("Could Not Create Socket. Error!!!!!\n");

return -1;

}

printf("Socket Created\n");

server\_addr.sin\_family = AF\_INET;

server\_addr.sin\_port = htons(2000);

server\_addr.sin\_addr.s\_addr = INADDR\_ANY;

if (bind(socket\_desc, (struct sockaddr \*)&server\_addr, sizeof(server\_addr)) < 0)

{

printf("Bind Failed. Error!!!!!\n");

return -1;

}

printf("Bind Done\n");

while (1)

{

if (listen(socket\_desc, 5) < 0)

{

printf("Listening Failed. Error!!!!!\n");

return -1;

}

printf("Listening for Incoming Connections.....\n");

client\_size = sizeof(client\_addr);

client\_sock = accept(socket\_desc, (struct sockaddr \*)&client\_addr, &client\_size);

if (client\_sock < 0)

{

printf("Accept Failed. Error!!!!!!\n");

return -1;

}

printf("Client Connected with IP: %s and Port No: %i\n", inet\_ntoa(client\_addr.sin\_addr), ntohs(client\_addr.sin\_port));

if (recv(client\_sock, client\_message, sizeof(client\_message), 0) < 0)

{

printf("Receive Failed. Error!!!!!\n");

return -1;

}

printf("Client Message: %s\n", client\_message);

// Extracting the ID from the client message

int client\_id = client\_message[strlen(client\_message) - 1] - '0';

sprintf(server\_message, "Hello I am server. Your received id is %d", client\_id);

if (send(client\_sock, server\_message, strlen(server\_message), 0) < 0)

{

printf("Send Failed. Error!!!!!\n");

return -1;

}

memset(server\_message, '\0', sizeof(server\_message));

memset(client\_message, '\0', sizeof(client\_message));

close(client\_sock);

}

close(socket\_desc);

return 0;

}

Client:

/\*

TCP\_Client. This Program will implement the Client Side for TCP\_Socket Programming.

It will get some data from user and will send to the server and as a reply from the

server, it will get its data back.

\*/

#include <stdio.h>

#include <string.h>

#include <sys/socket.h> //socket

#include <arpa/inet.h> //inet\_addr

int main(void)

{

int socket\_desc;

struct sockaddr\_in server\_addr;

char server\_message[2000], client\_message[2000];

//Cleaning the Buffers

memset(server\_message,'\0',sizeof(server\_message));

memset(client\_message,'\0',sizeof(client\_message));

//Creating Socket

socket\_desc = socket(AF\_INET, SOCK\_STREAM, 0);

if(socket\_desc < 0)

{

printf("Could Not Create Socket. Error!!!!!\n");

return -1;

}

printf("Socket Created\n");

//Specifying the IP and Port of the server to connect

server\_addr.sin\_family = AF\_INET;

server\_addr.sin\_port = htons(2000);

server\_addr.sin\_addr.s\_addr = inet\_addr("127.0.0.1");

//Now connecting to the server accept() using connect() from client side

if(connect(socket\_desc, (struct sockaddr\*)&server\_addr, sizeof(server\_addr)) < 0)

{

printf("Connection Failed. Error!!!!!");

return -1;

}

printf("Connected\n");

//Get Input from the User

printf("Enter Message: ");

gets(client\_message); //One is that gets() will only get character string data.

// will get only one variable at a time.

// reads characters from stdin and loads them into str

//Send the message to Server

if(send(socket\_desc, client\_message, strlen(client\_message),0) < 0)

{

printf("Send Failed. Error!!!!\n");

return -1;

}

//Receive the message back from the server

if(recv(socket\_desc, server\_message, sizeof(server\_message),0) < 0)

{

printf("Receive Failed. Error!!!!!\n");

return -1;

}

printf("Server Message: %s\n",server\_message);

memset(server\_message,'\0',sizeof(server\_message));

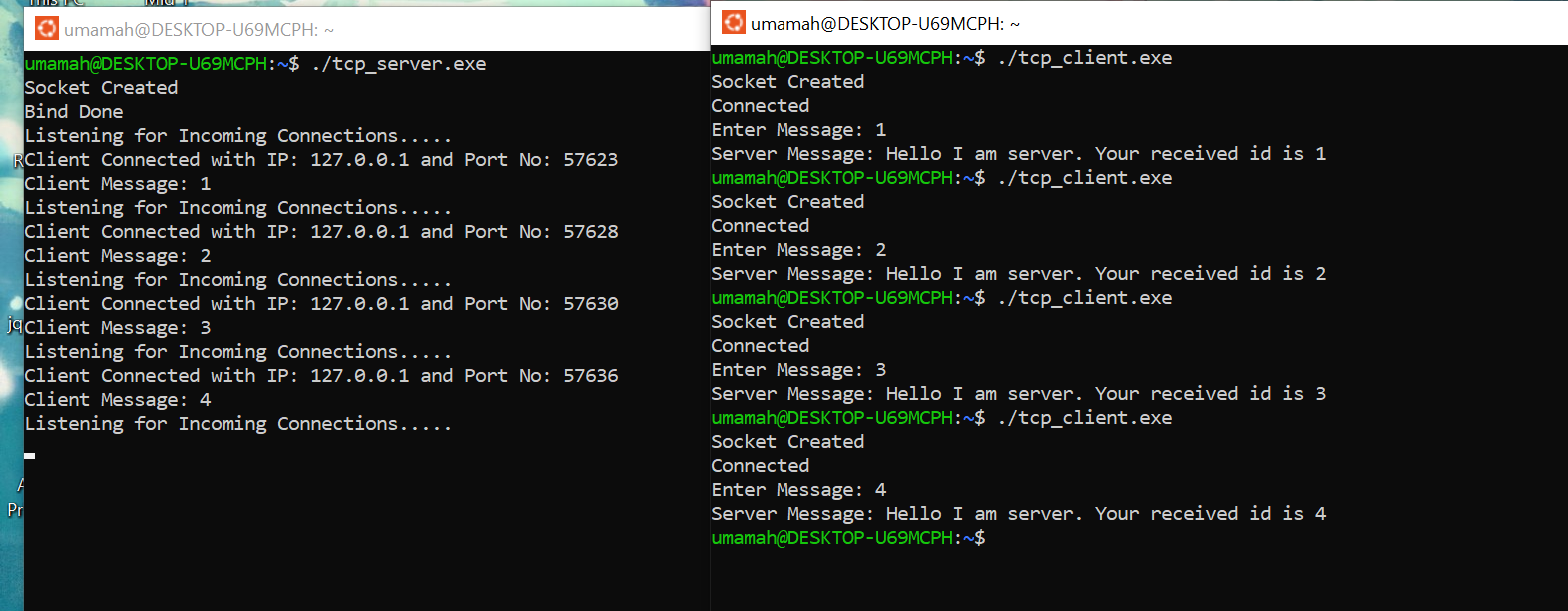
memset(client\_message,'\0',sizeof(client\_message));

//Closing the Socket

close(socket\_desc);

return 0;

}



**Task 2:**

Server:

#include <stdio.h>

#include <string.h>

#include <sys/socket.h>

#include <arpa/inet.h>

#include <unistd.h>

#include <ctype.h>

int containsVowels(char \*word) {

char vowels[] = "AaEeIiOoUu";

int i,j;

for ( i = 0; word[i] != '\0'; i++) {

for (j = 0; vowels[j] != '\0'; j++) {

if (word[i] == vowels[j]) {

return 1;

}

}

}

return 0;

}

void invertWord(char \*word) {

int length = strlen(word);

int i;

for ( i = 0; i < length / 2; i++) {

char temp = word[i];

word[i] = word[length - i - 1];

word[length - i - 1] = temp;

}

}

int main(void) {

int socket\_desc, client\_sock, client\_size;

struct sockaddr\_in server\_addr, client\_addr;

char server\_message[2000], client\_message[2000];

memset(server\_message, '\0', sizeof(server\_message));

memset(client\_message, '\0', sizeof(client\_message));

socket\_desc = socket(AF\_INET, SOCK\_STREAM, 0);

if (socket\_desc < 0) {

printf("Could Not Create Socket. Error!!!!!\n");

return -1;

}

printf("Socket Created\n");

server\_addr.sin\_family = AF\_INET;

server\_addr.sin\_port = htons(2000);

server\_addr.sin\_addr.s\_addr = INADDR\_ANY;

if (bind(socket\_desc, (struct sockaddr \*)&server\_addr, sizeof(server\_addr)) < 0) {

printf("Bind Failed. Error!!!!!\n");

return -1;

}

printf("Bind Done\n");

while (1)

{

if (listen(socket\_desc, 5) < 0) {

printf("Listening Failed. Error!!!!!\n");

return -1;

}

printf("Listening for Incoming Connections.....\n");

client\_size = sizeof(client\_addr);

client\_sock = accept(socket\_desc, (struct sockaddr \*)&client\_addr, &client\_size);

if (client\_sock < 0) {

printf("Accept Failed. Error!!!!!!\n");

return -1;

}

printf("Client Connected with IP: %s and Port No: %i\n", inet\_ntoa(client\_addr.sin\_addr), ntohs(client\_addr.sin\_port));

if (recv(client\_sock, client\_message, sizeof(client\_message), 0) < 0) {

printf("Receive Failed. Error!!!!!\n");

return -1;

}

printf("Client Message: %s\n", client\_message);

// Tokenize the string into words

char \*token = strtok(client\_message, " ");

while (token != NULL) {

if (containsVowels(token)) {

invertWord(token);

}

strcat(server\_message, token);

strcat(server\_message, " ");

token = strtok(NULL, " ");

}

if (send(client\_sock, server\_message, strlen(server\_message), 0) < 0) {

printf("Send Failed. Error!!!!!\n");

return -1;

}

memset(server\_message, '\0', sizeof(server\_message));

memset(client\_message, '\0', sizeof(client\_message));

close(client\_sock);

}

close(socket\_desc);

return 0;

}

Client:

#include <stdio.h>

#include <string.h>

#include <sys/socket.h>

#include <arpa/inet.h>

#include <unistd.h>

int main(void) {

int socket\_desc;

struct sockaddr\_in server\_addr;

char server\_message[2000], client\_message[2000];

memset(server\_message, '\0', sizeof(server\_message));

memset(client\_message, '\0', sizeof(client\_message));

socket\_desc = socket(AF\_INET, SOCK\_STREAM, 0);

if (socket\_desc < 0) {

printf("Could Not Create Socket. Error!!!!!\n");

return -1;

}

printf("Socket Created\n");

server\_addr.sin\_family = AF\_INET;

server\_addr.sin\_port = htons(2000);

server\_addr.sin\_addr.s\_addr = inet\_addr("127.0.0.1");

if (connect(socket\_desc, (struct sockaddr \*)&server\_addr, sizeof(server\_addr)) < 0) {

printf("Connection Failed. Error!!!!!");

return -1;

}

printf("Connected\n");

printf("Enter a string: ");

fgets(client\_message, sizeof(client\_message), stdin);

if (send(socket\_desc, client\_message, strlen(client\_message), 0) < 0) {

printf("Send Failed. Error!!!!\n");

return -1;

}

if (recv(socket\_desc, server\_message, sizeof(server\_message), 0) < 0) {

printf("Receive Failed. Error!!!!!\n");

return -1;

}

printf("Server Message: %s\n", server\_message);

printf("after reversing: ");

// Tokenize the string into words

char \*token = strtok(server\_message, " ");

while (token != NULL) {

if (strchr(token, 'a') == NULL && strchr(token, 'e') == NULL && strchr(token, 'i') == NULL &&

strchr(token, 'o') == NULL && strchr(token, 'u') == NULL &&

strchr(token, 'A') == NULL && strchr(token, 'E') == NULL && strchr(token, 'I') == NULL &&

strchr(token, 'O') == NULL && strchr(token, 'U') == NULL) {

int i;

for (i = strlen(token) - 1; i >= 0; i--) {

printf("%c", token[i]);

}

printf(" ");

} else {

printf("%s ", token);

}

token = strtok(NULL, " ");

}

printf("\n");

memset(server\_message, '\0', sizeof(server\_message));

memset(client\_message, '\0', sizeof(client\_message));

close(socket\_desc);

return 0;

}

