C.N LAB ASSIGNMENT - 5

NAME: RUDRASISH MISHRA
ROLL NO: 1906649
SECTION : IT-8

Q1: Write client and server side socket program to exchange the “Hello” message between them. Make sure the server does not stop after responding to the client.

CODE:

SERVER:

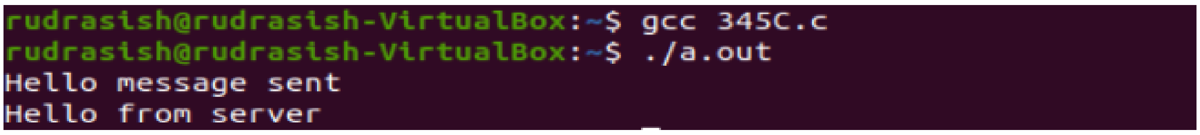
```
#include <unistd.h>
#include <stdio.h>
#include <sys/socket.h>
#include <stdlib.h>
#include <netinet/in.h>
#include <string.h>
#define PORT 8080
int main(int argc, char const *argv[])
{
    int server_fd, new_socket, valread;
    struct sockaddr_in address;
    int opt = 1;
    int addrlen = sizeof(address);
    char buffer[1024] = {0};
    char *hello = "Hello from server";
    if ((server_fd = socket(AF_INET, SOCK_STREAM, 0)) == 0)
    {
        perror("socket failed");
        exit(EXIT_FAILURE);
    }

    if (setsockopt(server_fd, SOL_SOCKET, SO_REUSEADDR | SO_REUSEPORT,
                                                  &opt, sizeof(opt)))
    {
        perror("setsockopt");
        exit(EXIT_FAILURE);
    }
    address.sin_family = AF_INET;
    address.sin_addr.s_addr = INADDR_ANY;
    address.sin_port = htons( PORT );


    if (bind(server_fd, (struct sockaddr *)&address,sizeof(address))<0)
    {
        perror("bind failed");
        exit(EXIT_FAILURE);
    }
    if (listen(server_fd, 3) < 0)
    {
        perror("listen");
        exit(EXIT_FAILURE);
    }
    if ((new_socket = accept(server_fd, (struct sockaddr *)&address,
                       (socklen_t*)&addrlen))<0)
    {
        perror("accept");
        exit(EXIT_FAILURE);
    }
    valread = read( new_socket , buffer, 1024);
    printf("%s\n",buffer );
    send(new_socket , hello , strlen(hello) , 0 );
    printf("Hello message sent\n");
    return 0;
}
```

CLIENT:

```
#include <stdio.h>
#include <sys/socket.h>
#include <arpa/inet.h>
#include <unistd.h>
#include <string.h>
#define PORT 8080

int main(int argc, char const *argv[])
{
    int sock = 0, valread;
    struct sockaddr_in serv_addr;
    char *hello = "Hello from client";
    char buffer[1024] = {0};
    if ((sock = socket(AF_INET, SOCK_STREAM, 0)) < 0)
    {
        printf("\n Socket creation error \n");
        return -1;
    }

    serv_addr.sin_family = AF_INET;
    serv_addr.sin_port = htons(PORT);


    if(inet_pton(AF_INET, "127.0.0.1", &serv_addr.sin_addr)<=0)
    {
        printf("\nInvalid address/ Address not supported \n");
        return -1;
    }

if (connect(sock, (struct sockaddr *)&serv_addr, sizeof(serv_addr)) < 0)
    {
        printf("\nConnection Failed \n");
        return -1;
    }
    send(sock , hello , strlen(hello) , 0 );
    printf("Hello message sent\n");
    valread = read( sock , buffer, 1024);
    printf("%s\n",buffer );
    return 0;
}
```

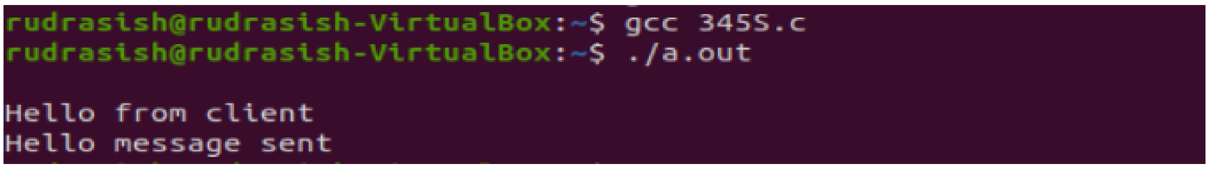
OUTPUT:

SERVER SIDE:



```
rudrasish@rudrasish-VirtualBox:~$ gcc 345C.c
rudrasish@rudrasish-VirtualBox:~$ ./a.out
Hello message sent
Hello from server
```

CLIENT SIDE:



```
rudrasish@rudrasish-VirtualBox:~$ gcc 345S.c
rudrasish@rudrasish-VirtualBox:~$ ./a.out

Hello from client
Hello message sent
```

Q2: Write client and server side socket program, where the client sends an array of unsorted integers,and the server sorts it and sends it to the client.

CODE:

SERVER:

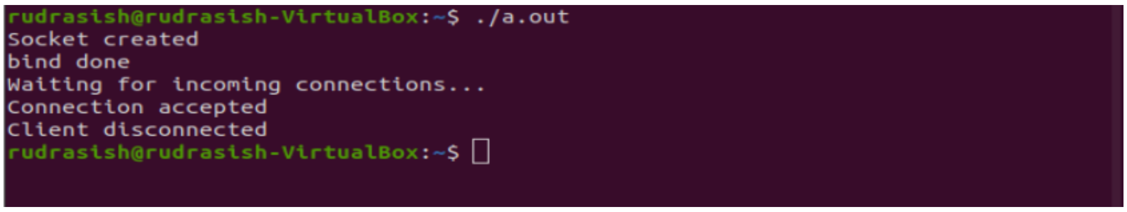
```
#include <arpa/inet.h>
#include <stdio.h>
#include <string.h>
#include <sys/socket.h>
#include <unistd.h>

void bubble_sort(int[], int);
int main(int argc, char* argv[])
{
    int socket_desc, client_sock, c, read_size;
    struct sockaddr_in server, client;
    int message[10], i;
    socket_desc = socket(AF_INET, SOCK_STREAM, 0);
    if (socket_desc == -1)
    {
        printf("Could not create socket");
    }
    puts("Socket created");
    server.sin_family = AF_INET;
    server.sin_addr.s_addr = INADDR_ANY;
    server.sin_port = htons(8880);
    if (bind(socket_desc, (struct sockaddr*)&server, sizeof(server)) < 0)




    {
        perror("bind failed. Error");
        return 1;
    }
    puts("bind done");
    listen(socket_desc, 3);
    puts("Waiting for incoming connections...");
    c = sizeof(struct sockaddr_in);
   client_sock = accept(socket_desc, (struct sockaddr*)&client, (sock-
len_t*)&c);

    if (client_sock < 0)
    {
        perror("accept failed");
        return 1;
    }

    puts("Connection accepted");
    while ((read_size = recv(client_sock, &message, 10 * sizeof(int), 0))
> 0)
 {
        bubble_sort(message, 10);
        write(client_sock, &message, 10 * sizeof(int));
    }

    if (read_size == 0) {
        puts("Client disconnected");
    }
    else if (read_size == -1) {
        perror("recv failed");
    }

    return 0;
}
  void bubble_sort(int list[], int n)
{
    int c, d, t;

    for (c = 0; c < (n - 1); c++) {
        for (d = 0; d < n - c - 1; d++) {
            if (list[d] > list[d + 1]) {


                t = list[d];
                list[d] = list[d + 1];
                list[d + 1] = t;
            }
        }
    }
}
```

CLIENT:

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

int main(int argc, char* argv[])
{
    int sock;
    struct sockaddr_in server;
    int server_reply[10];
    int number[10] = { 5, 4, 3, 8, 9, 1, 2, 0, 6 }, i, temp;
    sock = socket(AF_INET, SOCK_STREAM, 0);
    if (sock == -1)
    {
        printf("Could not create socket");
    }
    puts("Socket created");

    server.sin_addr.s_addr = inet_addr("127.0.0.1");
    server.sin_family = AF_INET;
    server.sin_port = htons(8880);

    if (connect(sock, (struct sockaddr*)&server, sizeof(server)) < 0)
    {
        perror("connect failed. Error");
        return 1;
    }
    puts("Connected\n");
    if (send(sock, &number, 10 * sizeof(int), 0) < 0)
    {
        puts("Send failed");
        return 1;
    }
   if (recv(sock, &server_reply, 10 * sizeof(int), 0) < 0) {
        puts("recv failed");
        return 0;
    }
    puts("Server reply :\n");
    for (i = 0; i < 10; i++)
    {
        printf("%d\n", server_reply[i]);
    }
    close(sock);
    return 0;
}
```

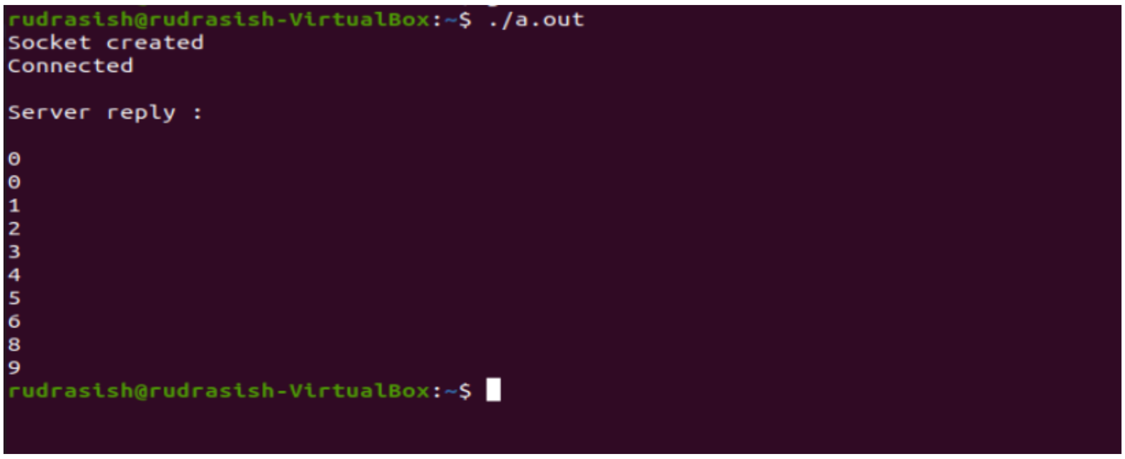
OUTPUT:

SERVER SIDE:



```
rudrasish@rudrasish-VirtualBox:~$ ./a.out
Socket created
bind done
Waiting for incoming connections...
Connection accepted
Client disconnected
rudrasish@rudrasish-VirtualBox:~$
```

CLIENT SIDE:



```
rudrasish@rudrasish-VirtualBox:~$ ./a.out
Socket created
Connected

Server reply :

0
0
1
2
3
4
5
6
8
9
rudrasish@rudrasish-VirtualBox:~$
```

Q3: Write client and server side socket program to count the occurrences of a word in a text file at the server. The client sends the word to the server.

CODE:

SERVER:

```
#include <unistd.h>
#include <stdio.h>
#include <sys/socket.h>
#include <stdlib.h>
#include <netinet/in.h>
#include <string.h>
#define PORT 8080

int search(char *word)
{
FILE *fptr;
char    *buffer;
long    numbytes;
```

```
fptr = fopen("/home/kiit/Desktop/Input.txt","r");
if(fptr == NULL)
   {
      printf("Error!");
      exit(1);
   }
fseek(fptr, 0L, SEEK_END);
numbytes = ftell(fptr);
fseek(fptr, 0L, SEEK_SET);
buffer = (char*)calloc(numbytes, sizeof(char));
if(buffer == NULL)
    return 1;

fread(buffer, sizeof(char), numbytes, fptr);
fclose(fptr);

int i,j=0, ctr=0,occ=0;
char newString[50];
for(i=0;i<=(strlen(buffer));i++){
     if(buffer[i]==' '||buffer[i]=='\0'||buffer[i]=='\n'){
     newString[j]='\0';

     if(strcmp(newString, word)==0)
     occ++;

     j=0;
     }
     else  {

     newString[j]= buffer[i];
     j++;
     }
}

free(buffer);
return occ;
}

int main()
{
    int server_fd, new_socket, valread;
    struct sockaddr_in s_address, c_address ;
    int addrlen = sizeof(s_address);
    char buffer[1024] = {0};
      char result[1024];
    if ((server_fd = socket(AF_INET, SOCK_STREAM, 0)) < 0)
    {
        perror("socket failed");
        exit(EXIT_FAILURE);
    }

 memset(&s_address, 0, sizeof(s_address));

    s_address.sin_family = AF_INET;
    s_address.sin_addr.s_addr = INADDR_ANY;
    s_address.sin_port = htons( PORT );
    if (bind(server_fd, (struct sockaddr *)&s_address,
sizeof(s_address))<0)
    {
        perror("bind failed");
        exit(EXIT_FAILURE);
    }

     if (listen(server_fd, 3) < 0)
    {
        perror("listen");
        exit(EXIT_FAILURE);
    }
    if ((new_socket = accept(server_fd, (struct sockaddr *)&c_address,
(socklen_t*) &addrlen))<0)
    {
        perror("accept");
        exit(EXIT_FAILURE);
    }

valread = recv( new_socket , buffer, 1024, 0);
int count= search(buffer);

 printf("Word received from the client\n");
sprintf(result, "%d", count);
    send(new_socket ,result , sizeof(result) , 0 );
printf("Result sent to Client\n");
     close(new_socket);
    return 0;
}
```

CLIENT:

```
#include <stdio.h>
#include <sys/socket.h>
#include <arpa/inet.h>
#include <unistd.h>
#include <string.h>
#include <stdlib.h>
#define PORT 8080

int main()
{
    int sock = 0, valread;
    struct sockaddr_in serv_addr;
    char word[50];
    char buffer[1024];
if ((sock = socket(AF_INET, SOCK_STREAM, 0)) < 0)
    {
        printf("\n Socket creation error \n");
        return -1;
    }

    serv_addr.sin_family = AF_INET;
    serv_addr.sin_port = htons(PORT);
    serv_addr.sin_addr.s_addr = INADDR_ANY;

    if (connect(sock, (struct sockaddr *)&serv_addr, sizeof(serv_addr)) <
0)
    {
        printf("\nConnection Failed \n");
        return -1;
    }
printf("Enter a word: ");
gets(word);

    send(sock , word , strlen(word) , 0 );

printf("Word sent to server\n");

    valread = recv( sock , buffer, 1024, 0);
    printf("Count= %s \n", buffer);
    return 0;
}
```

Q4: Write a client and server side socket program to make a single client server chat application.

CODE:

SERVER:

```
#include <stdio.h>
#include <netdb.h>
#include <netinet/in.h>
#include <stdlib.h>
#include <string.h>
#include <sys/socket.h>
#include <sys/types.h>
#define MAX 80
#define PORT 8080
#define SA struct sockaddr

void func(int sockfd)
{
    char buff[MAX];
    int n;
    for (;;)
{
        bzero(buff, MAX);
        read(sockfd, buff, sizeof(buff));
        printf("From client: %s\t To client : ", buff);
        bzero(buff, MAX);
        n = 0;
        while ((buff[n++] = getchar()) != '\n');

        write(sockfd, buff, sizeof(buff));
        if (strncmp("exit", buff, 4) == 0) {
            printf("Server Exit...\n");
            break;
        }
    }
}

int main()
{
    int sockfd, connfd, len;
    struct sockaddr_in servaddr, cli;
    sockfd = socket(AF_INET, SOCK_STREAM, 0);
    if (sockfd == -1) {
        printf("socket creation failed...\n");
        exit(0);
    }
    else
        printf("Socket successfully created..\n");
    bzero(&servaddr, sizeof(servaddr));
    servaddr.sin_family = AF_INET;
    servaddr.sin_addr.s_addr = htonl(INADDR_ANY);
    servaddr.sin_port = htons(PORT);
    if ((bind(sockfd, (SA*)&servaddr, sizeof(servaddr))) != 0) {
        printf("socket bind failed...\n");
        exit(0);
    }
    else
        printf("Socket successfully binded..\n");
    if ((listen(sockfd, 5)) != 0) {
        printf("Listen failed...\n");
        exit(0);
    }
    else
        printf("Server listening..\n");
    len = sizeof(cli);
    connfd = accept(sockfd, (SA*)&cli, &len);
    if (connfd < 0) {
        printf("server acccept failed...\n");
        exit(0);
    }
    else
        printf("server acccept the client...\n");
    func(connfd);
    close(sockfd);
}
```

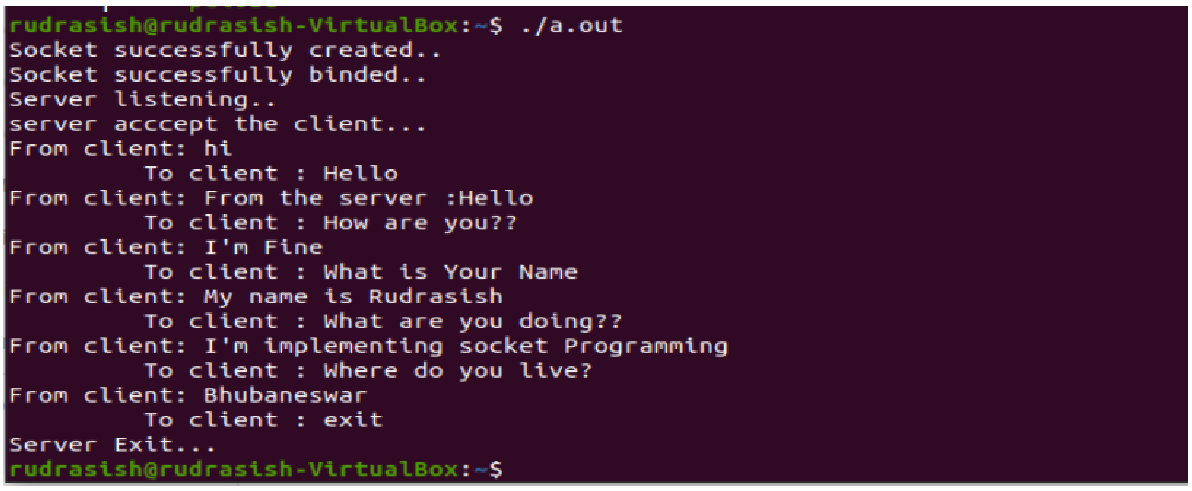
CLIENT:

```
#include <netdb.h>
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <sys/socket.h>
#define MAX 80
#define PORT 8080
```

```
#define SA struct sockaddr

void func(int sockfd)
{
    char buff[MAX];
    int n;
    for (;;) {
        bzero(buff, sizeof(buff));
        printf("Enter the string : ");
        n = 0;
        while ((buff[n++] = getchar()) != '\n');
        write(sockfd, buff, sizeof(buff));
        bzero(buff, sizeof(buff));
        read(sockfd, buff, sizeof(buff));
        printf("From Server : %s", buff);
        if ((strncmp(buff, "exit", 4)) == 0)
        {
            printf("Client Exit...\n");
            break;
        }
    }
}

int main()
{
    int sockfd, connfd;
    struct sockaddr_in servaddr, cli;
    sockfd = socket(AF_INET, SOCK_STREAM, 0);
    if (sockfd == -1)
    {
        printf("socket creation failed...\n");
        exit(0);
    }
    else
        printf("Socket successfully created..\n");
    bzero(&servaddr, sizeof(servaddr));
    servaddr.sin_family = AF_INET;
    servaddr.sin_addr.s_addr = inet_addr("127.0.0.1");
    servaddr.sin_port = htons(PORT);
    if (connect(sockfd, (SA*)&servaddr, sizeof(servaddr)) != 0)
    {
        printf("connection with the server failed...\n");
        exit(0);
    }
    else
        printf("connected to the server..\n");
    func(sockfd);
    close(sockfd);
}
```

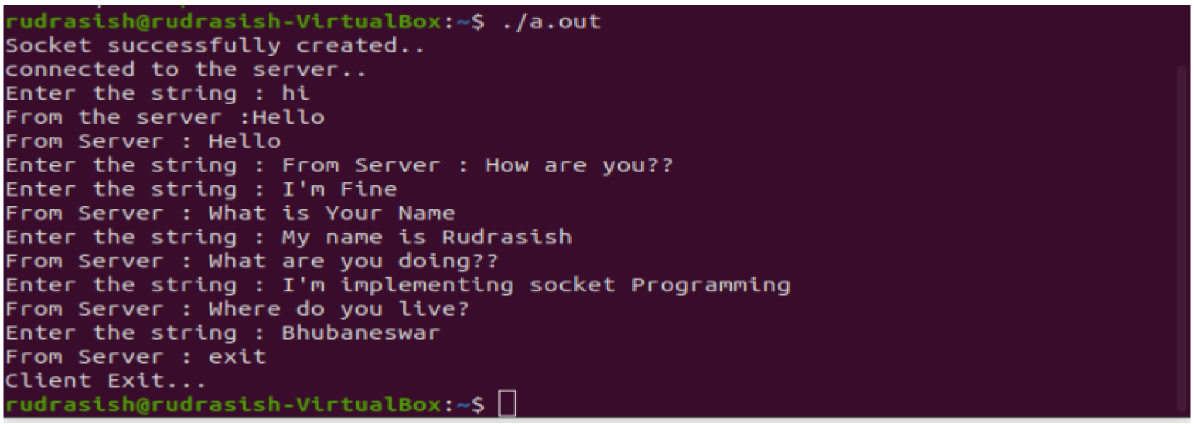
OUTPUT:

SERVER SIDE:



```
rudrasish@rudrasish-VirtualBox:~$ ./a.out
Socket successfully created..
Socket successfully binded..
Server listening..
server acccept the client...
From client: hi
         To client : Hello
From client: From the server :Hello
         To client : How are you??
From client: I'm Fine
         To client : What is Your Name
From client: My name is Rudrasish
         To client : What are you doing??
From client: I'm implementing socket Programming
         To client : Where do you live?
From client: Bhubaneswar
         To client : exit
Server Exit...
rudrasish@rudrasish-VirtualBox:~$
```

CLIENT SIDE:



```
rudrasish@rudrasish-VirtualBox:~$ ./a.out
Socket successfully created..
connected to the server..
Enter the string : hi
From the server :Hello
From Server : Hello
Enter the string : From Server : How are you??
Enter the string : I'm Fine
From Server : What is Your Name
Enter the string : My name is Rudrasish
From Server : What are you doing??
Enter the string : I'm implementing socket Programming
From Server : Where do you live?
Enter the string : Bhubaneswar
From Server : exit
Client Exit...
rudrasish@rudrasish-VirtualBox:~$
```

NAME: RUDRASISH MISHRA
ROLL NO: 1906649
SECTION : IT-8