C.N LAB ASSIGNMENT - 4

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

Q1:Socket Programming between Client and Server to convert lowercaseword/sentence to upper case word/sentence in UDP.

CODE:

SERVER:

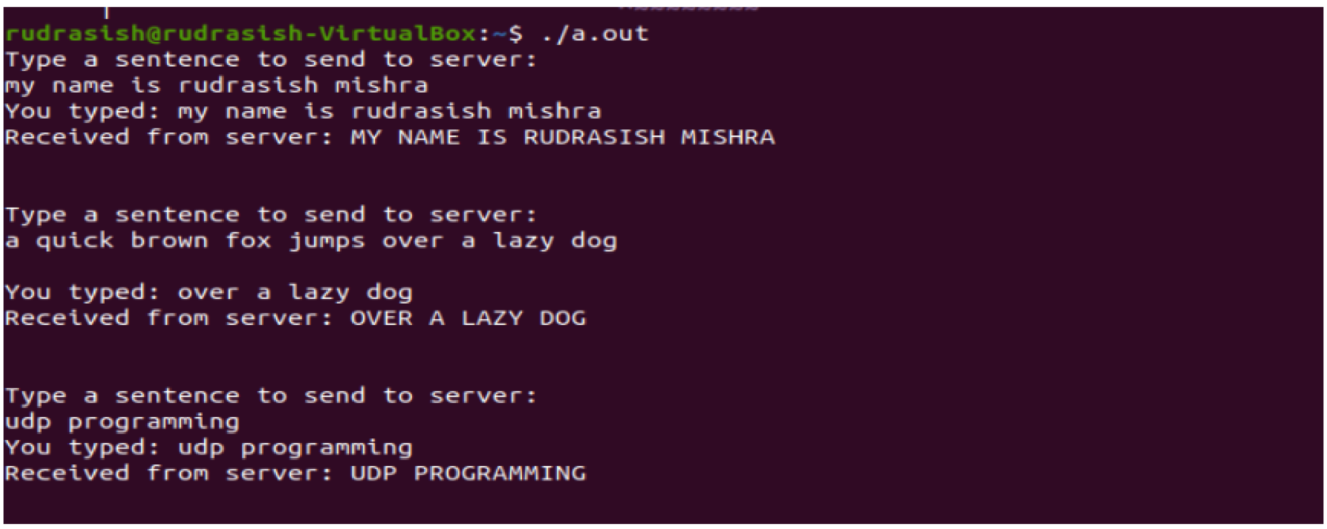
```
#include <stdio.h>

#include <sys/socket.h>

#include <netinet/in.h>

#include <string.h>

#include <stdlib.h>



int main()

{

  int welcomeSocket, newSocket, portNum, clientLen, nBytes;

  char buffer[1024];

  struct sockaddr_in serverAddr;

  struct sockaddr_storage serverStorage;

  socklen_t addr_size;

  int i;

  welcomeSocket = socket(PF_INET, SOCK_STREAM, 0);

  portNum = 7891;

  serverAddr.sin_family = AF_INET;

  serverAddr.sin_port = htons(portNum);

  serverAddr.sin_addr.s_addr = inet_addr("127.0.0.1");
  memset(serverAddr.sin_zero, '\0', sizeof serverAddr.sin_zero);

bind(welcomeSocket, (struct sockaddr *) &serverAddr, sizeof(serverAddr));

if(listen(welcomeSocket,5)==0)

    printf("Listening\n");

  else

    printf("Error\n");

    addr_size = sizeof serverStorage;

 while(1)

{

 newSocket=accept(welcomeSocket,(struct sockaddr *)&serverStorage,&addr_size);

    if(!fork()){

      nBytes = 1;

      while(nBytes!=0)

{        nBytes = recv(newSocket,buffer,1024,0);

        for (i=0;i<nBytes-1;i++)

       {

          buffer[i] = toupper(buffer[i]);

        }

       send(newSocket,buffer,nBytes,0);

      }

      close(newSocket);

      exit(0);

    }

    else

{    close(newSocket);

    }

  }
```

```
return 0;

}
```

CLIENT:

```
#include <stdio.h>

#include <sys/socket.h>

#include <netinet/in.h>

#include <string.h>


int main(){

  int clientSocket, portNum, nBytes;

  char buffer[1024];

  struct sockaddr_in serverAddr;

  socklen_t addr_size;

  clientSocket = socket(PF_INET, SOCK_STREAM, 0);

  portNum = 7891;

  serverAddr.sin_family = AF_INET;

  serverAddr.sin_port = htons(portNum);

  serverAddr.sin_addr.s_addr = inet_addr("127.0.0.1");

  memset(serverAddr.sin_zero, '\0', sizeof serverAddr.sin_zero);

addr_size = sizeof serverAddr;

  connect(clientSocket, (struct sockaddr *) &serverAddr, addr_size);

while(1){

    printf("Type a sentence to send to server:\n");

    fgets(buffer,1024,stdin);
    printf("You typed: %s",buffer);


    nBytes = strlen(buffer) + 1;


    send(clientSocket,buffer,nBytes,0);

    recv(clientSocket, buffer, 1024, 0);

    printf("Received from server: %s\n\n",buffer);

  }

return 0;

}
```

OUTPUT:



```
rudrasish@rudrasish-VirtualBox:~$ ./a.out
Type a sentence to send to server:
my name is rudrasish mishra
You typed: my name is rudrasish mishra
Received from server: MY NAME IS RUDRASISH MISHRA


Type a sentence to send to server:
a quick brown fox jumps over a lazy dog

You typed: over a lazy dog
Received from server: OVER A LAZY DOG


Type a sentence to send to server:
udp programming
You typed: udp programming
Received from server: UDP PROGRAMMING
```

Q2: Write a client and server program where client will send a 3 digit number and server will check whether that is palindrome or not and send it to client.

CODE:

SERVER:

```
#include <arpa/inet.h>
#include <netinet/in.h>
#include <stdio.h>
#include <string.h>
#include <sys/socket.h>
#include <sys/stat.h>
```

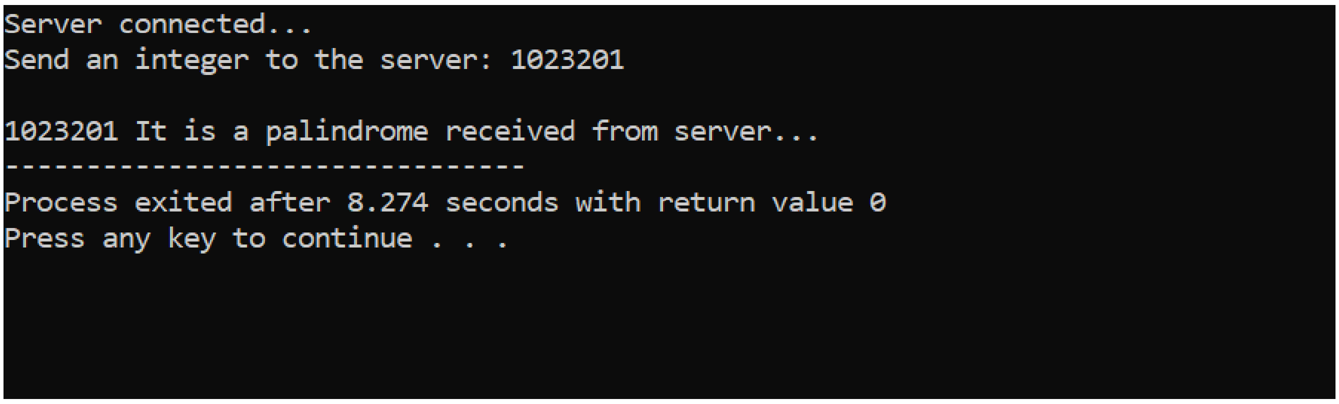
```
#include <sys/types.h>

main()
{
    struct sockaddr_in client, server;
    int s, n, sock, g, j, left, right, flag;
    char b1[20], b2[10], b3[10], b4[10];
    s = socket(AF_INET, SOCK_STREAM, 0);
    server.sin_family = AF_INET;
    server.sin_port = 2000;
    server.sin_addr.s_addr = inet_addr("127.0.0.1");
    bind(s, (struct sockaddr*)&server, sizeof server);
    listen(s, 1);
    n = sizeof client;

    sock = accept(s, (struct sockaddr*)&client, &n);
    for (;;) {
        recv(sock, b1, sizeof(b1), 0);
        printf("\nThe string received is:%s\n", b1);
        if (strlen(b1) == 0)
            flag = 1;
        else {
            left = 0;
            right = strlen(b1) - 1;
            flag = 1;
            while (left < right && flag) {
                if (b1[left] != b1[right])
                    flag = 0;
                else {
                    left++;
                    right--;
                }
            }
        }
        send(sock, &flag, sizeof(int), 0);
        break;
    }
    close(sock);
    close(s);
}
```

CLIENT:

```
#include <arpa/inet.h>
#include <netinet/in.h>
#include <stdio.h>
#include <string.h>
#include <sys/socket.h>
#include <sys/stat.h>
#include <sys/types.h>
```

```
main()
{
    struct sockaddr_in client;
    int s, flag;
    char buffer[20];
    s = socket(AF_INET, SOCK_STREAM, 0);
    client.sin_family = AF_INET;
    client.sin_port = 2000;
    client.sin_addr.s_addr = inet_addr("127.0.0.1");
    connect(s, (struct sockaddr*)&client, sizeof client);

    for (;;) {
        printf("\nSend an integer to the server: ");
        scanf("%s", buffer);

        printf("\nClient: %s", buffer);
        send(s, buffer, sizeof(buffer), 0);
        recv(s, &flag, sizeof(int), 0);

        if (flag == 1) {
            printf("\nServer: It is a Palindrome received from Server\n");
            break;
        }
        else {
            printf("\nServer: The Number is not a palindromereceived from server\n");
            break;
        }
    }

    close(s);
}
```

OUTPUT:



```
Server connected...
Send an integer to the server: 1023201

1023201 It is a palindrome received from server...
--------------------------------
Process exited after 8.274 seconds with return value 0
Press any key to continue . . .
```

Q3:Write a client and server program in udp where client and server communication will be done.if server receives a exit message from client then the connection will stops.

CODE:

SERVER:

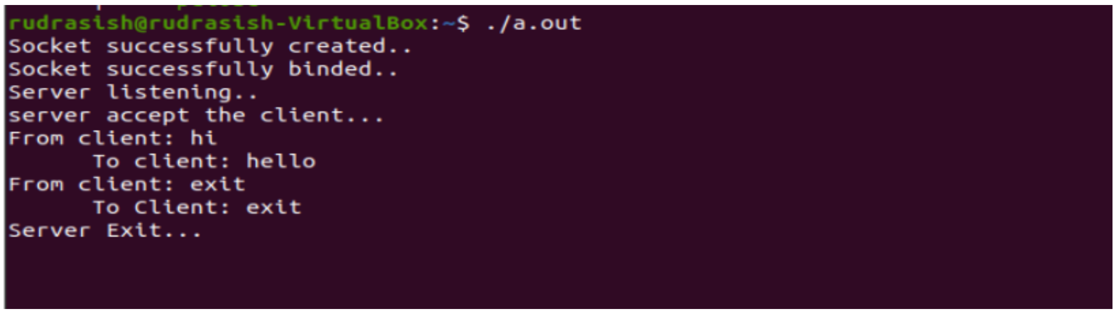
```
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <string.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <arpa/inet.h>
#include <netinet/in.h>

#define PORT     8080
#define MAXLINE 1024


int main() {
    int sockfd;
    char buffer[MAXLINE];
    char *hello = "Hello from server";
    struct sockaddr_in servaddr, cliaddr;
    if ( (sockfd = socket(AF_INET, SOCK_DGRAM, 0)) < 0 )
   {
        perror("socket creation failed");
        exit(EXIT_FAILURE);
    }

    memset(&servaddr, 0, sizeof(servaddr));
    memset(&cliaddr, 0, sizeof(cliaddr));

    servaddr.sin_family    = AF_INET; // IPv4
    servaddr.sin_addr.s_addr = INADDR_ANY;
    servaddr.sin_port = htons(PORT);

    if (strncmp("exit", buff, 4) == 0)
       {
            printf("Server Exit...\n");
            break;
        }
    if ( bind(sockfd, (const struct sockaddr *)&servaddr,
            sizeof(servaddr)) < 0 )
    {
        perror("bind failed");
        exit(EXIT_FAILURE);
    }

    int len, n;

    len = sizeof(cliaddr);

    n = recvfrom(sockfd, (char *)buffer, MAXLINE,
                MSG_WAITALL, ( struct sockaddr *) &cliaddr,
                &len);
    buffer[n] = '\0';
    printf("Client : %s\n", buffer);
    sendto(sockfd, (const char *)hello, strlen(hello),
        MSG_CONFIRM, (const struct sockaddr *) &cliaddr,
            len);
    printf("Hello message sent.\n");

    return 0;
}
```

CLIENT:

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

#define PORT     8080
#define MAXLINE 1024


int main()
{
    int sockfd;
    char buffer[MAXLINE];
    char *hello = "Hello from client";
    struct sockaddr_in     servaddr;
    if ( (sockfd = socket(AF_INET, SOCK_DGRAM, 0)) < 0 )
    {
        perror("socket creation failed");
        exit(EXIT_FAILURE);

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


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

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

    int n, len;

    sendto(sockfd, (const char *)hello, strlen(hello),
        MSG_CONFIRM, (const struct sockaddr *) &servaddr,
            sizeof(servaddr));
    printf("Hello message sent.\n");

    n = recvfrom(sockfd, (char *)buffer, MAXLINE,
                MSG_WAITALL, (struct sockaddr *) &servaddr,
                &len);
    buffer[n] = '\0';
    printf("Server : %s\n", buffer);

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

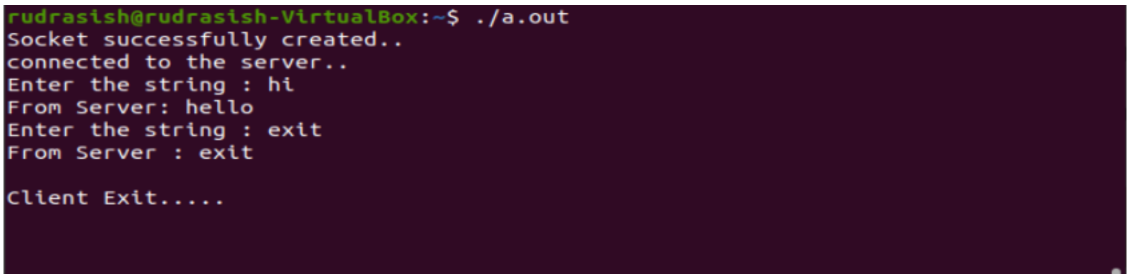
OUTPUT:

SERVER SIDE:



```
rudrasish@rudrasish-VirtualBox:~$ ./a.out
Socket successfully created..
Socket successfully binded..
Server listening..
server accept the client...
From client: hi
	To client: hello
From client: exit
	To Client: exit
Server Exit...
```

CLIENT SIDE:



```
rudrasish@rudrasish-VirtualBox:~$ ./a.out
Socket successfully created..
connected to the server..
Enter the string : hi
From Server: hello
Enter the string : exit
From Server : exit

Client Exit.....
```

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