END SEMESTER ASSESSMENT (ESA)
B.TECH. (CSE)
IV SEMESTER

UE20CS253 – COMPUTER NETWORKS LABORATORY

PROJECT REPORT

ON

STRING PROCESSING SEVER USING TCP/IP

SUBMITTED BY

NAME	SRN
1) EMIL BLUEMAX	PES2UG20CS431
2) J.P.DANIEL CHRISTOPHER	PES2UG20CS433
3) KARAN CHOUHAN D	PES2UG20CS437

JANUARY – MAY 2022

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

ELECTRONIC CITY CAMPUS,

BENGALURU – 560100, KARNATAKA, INDIA

ABSTRACT OF THE PROJECT:

TCP Server –

1)using create(), Create TCP socket.

2)using bind(), Bind the socket to server address.

3)using listen(), put the server socket in a passive mode, where it waits for the client to approach the server to make a connection

4)using accept(), At this point, connection is established between client and server, and they are ready to transfer data.

TCP Client –

1)Create TCP socket.

2)connect newly created client socket to server.

In this project we are implementing string processing server in C program using TCP/IP persistent protocol .First we run the server which passively waits for the client to send the request. Then we run the client program.The server processes the string based on the choice sent by the client. The string sent by the client is in the format <choice>#<string> . The string processing functions we have implemented are : converting to upper , lower case ; checking if string is a palindrome ; length of a string; checking if a string is a anagram, pangram , isogram , reversing a string . We exit the application by entering “exit”. And then the application terminates TCP/IP connection .

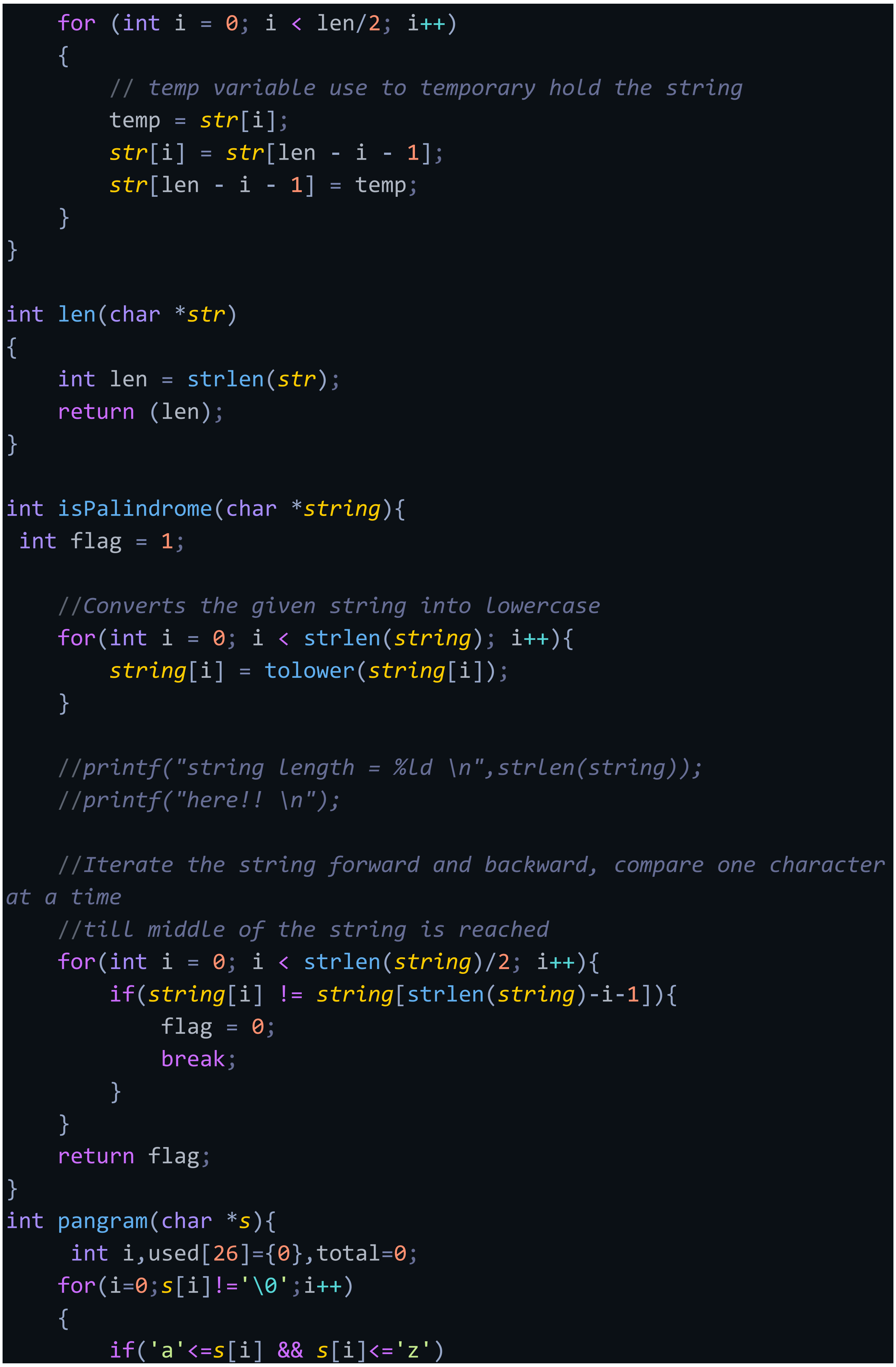
CODE:

Str_client.c

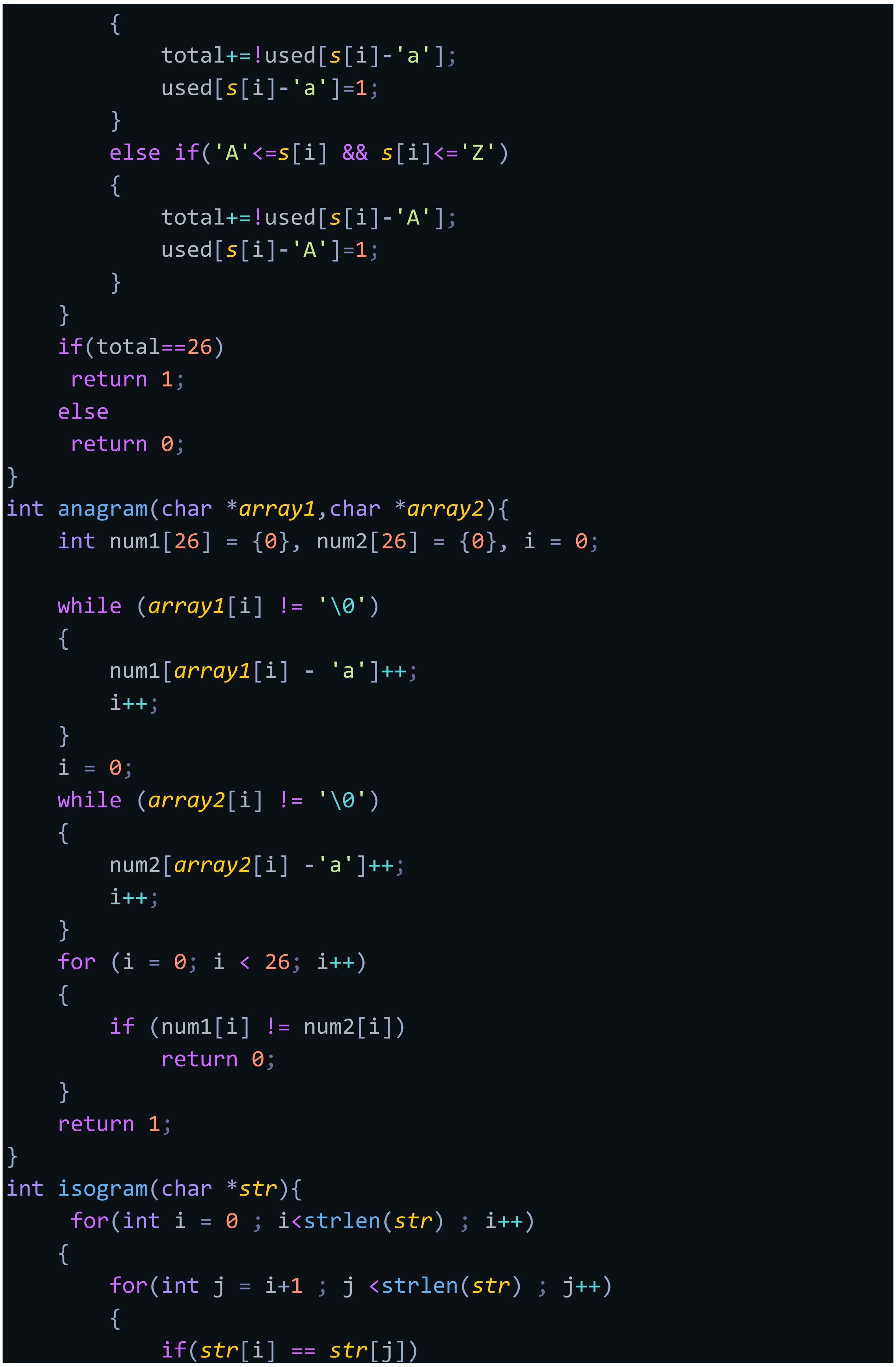
```c
#include <netdb.h>
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <ctype.h>
#include <unistd.h>
#include <arpa/inet.h>
#include <sys/socket.h>
#define MAX 800
#define PORT 8080
#define SA struct sockaddr
void func(int sockfd)
{
    char buff[MAX];
    int cnt = 0;
    int n,c;
    for (;;) {
        bzero(buff, sizeof(buff));
        printf("\nEnter the string to be processed in the given
format \n");
        printf("1#<str> for converting <str> to UPPER CASE \n");
        printf("2#<str> for converting <str> to LOWER CASE \n");
        printf("3#<str> to check if <str> is a PALINDROME \n");
        printf("4#<str> for REVERSING the <str> \n");
        printf("5#<str> for finding the LENGTH of the <str> \n");
        printf("6#<str> for checking if <str> is a PANAGRAM\n");
        printf("7#<str>#<str> for checking if  the <str> and <str>
are ANAGRAMS  \n");
        printf("8#<str> for checking if <str> is a ISOGRAM\n");
        printf("exit for EXITING the application \n");
        printf("\n Enter the string : ");

        n = 0;
        while ((buff[n++] = getchar()) != '\n');

        printf("sending : %s \n",buff);
        write(sockfd, buff, sizeof(buff));
        if ((strncmp(buff, "exit", 4)) == 0)
        {
            printf("Client Exit...\n");
            break;
        }

        bzero(buff, sizeof(buff));
        read(sockfd, buff, sizeof(buff));
        printf("From Server  : %s \n", buff);

    }
}

int main()
{
    int sockfd, connfd;
    struct sockaddr_in servaddr, cli;

    // socket create and verification
    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));

    // assign IP, PORT
    servaddr.sin_family = AF_INET;
    servaddr.sin_addr.s_addr = inet_addr("127.0.0.1");
    servaddr.sin_port = htons(PORT);

    // connect the client socket to server socket
    if (connect(sockfd, (SA*)&servaddr, sizeof(servaddr)) != 0) {
        printf("connection with the server failed...\n");
        exit(0);
    }
    else
        printf("connected to the server..\n");
    // function for chat
    func(sockfd);

    // close the socket
    close(sockfd);
}
```

Str_server.c

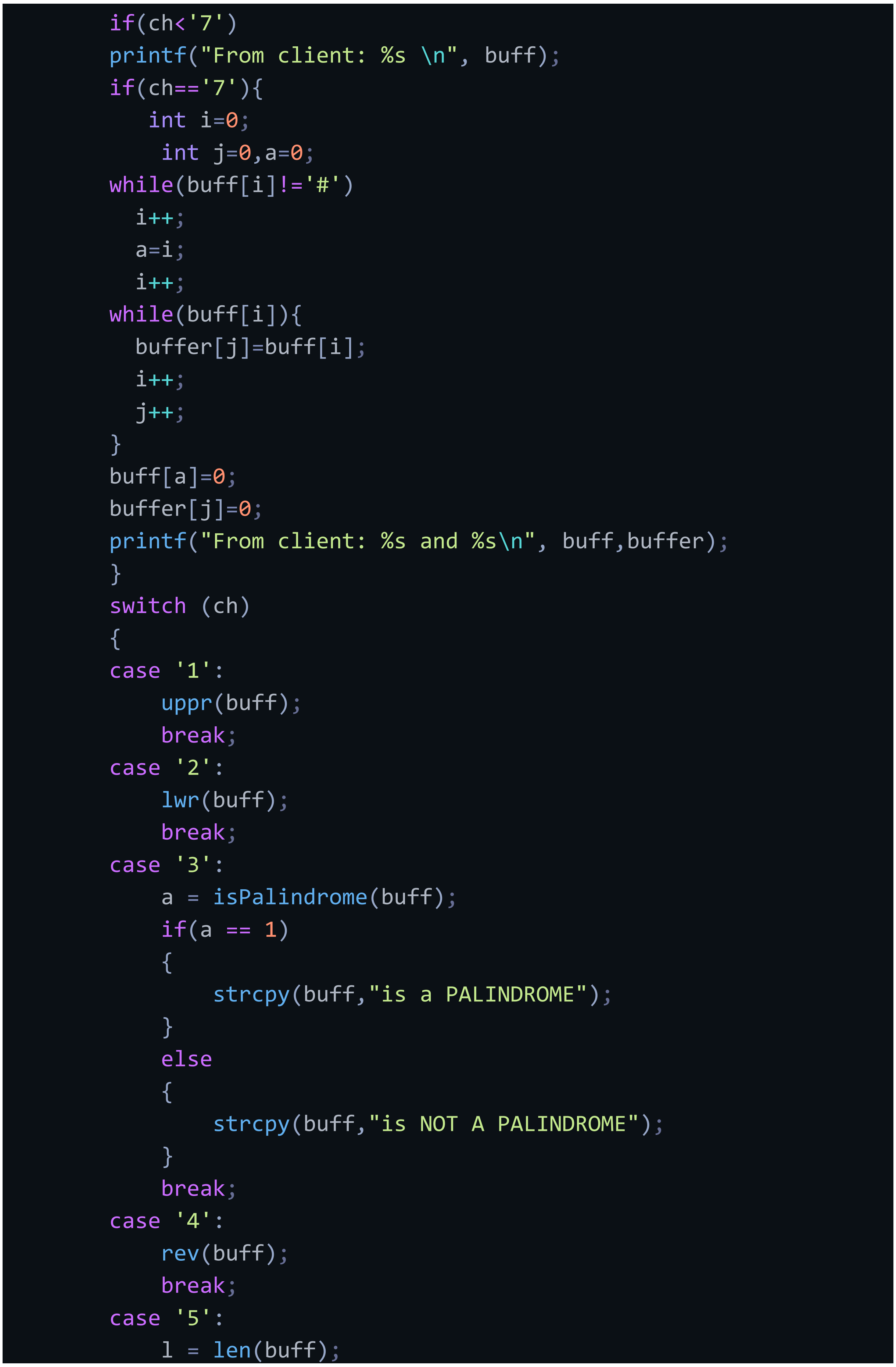
```
#include <stdio.h>
#include <netdb.h>
#include <netinet/in.h>
#include <stdlib.h>
#include <ctype.h>
#include <string.h>
#include <unistd.h>
#include <sys/socket.h>
#include <sys/types.h>
#define MAX 800
#define PORT 8080
#define SA struct sockaddr
void uppr(char *buff){
    int i=0;
    while(buff[i])
        {
            buff[i] = toupper(buff[i]);
            i++;
        }
}
void lwr(char *buff){
    int i=0;
    while(buff[i])
        {
            buff[i] = tolower(buff[i]);
            i++;
        }
}

void rev(char *str){
       int len = strlen(str); // use strlen() to get the length of
str string
      char temp;
    // use for loop to iterate the string
```



```c
    for (int i = 0; i < len/2; i++)
    {
        // temp variable use to temporary hold the string
        temp = str[i];
        str[i] = str[len - i - 1];
        str[len - i - 1] = temp;
    }
}

int len(char *str)
{
    int len = strlen(str);
    return (len);
}

int isPalindrome(char *string){
 int flag = 1;

    //Converts the given string into lowercase
    for(int i = 0; i < strlen(string); i++){
        string[i] = tolower(string[i]);
    }

    //printf("string length = %ld \n",strlen(string));
    //printf("here!! \n");

    //Iterate the string forward and backward, compare one character
at a time
    //till middle of the string is reached
    for(int i = 0; i < strlen(string)/2; i++){
        if(string[i] != string[strlen(string)-i-1]){
            flag = 0;
            break;
        }
    }
    return flag;
}
int pangram(char *s){
     int i,used[26]={0},total=0;
    for(i=0;s[i]!='\0';i++)
    {
        if('a'<=s[i] && s[i]<='z')
```



```c
        {
            total+=!used[s[i]-'a'];
            used[s[i]-'a']=1;
        }
        else if('A'<=s[i] && s[i]<='Z')
        {
            total+=!used[s[i]-'A'];
            used[s[i]-'A']=1;
        }
    }
    if(total==26)
     return 1;
    else
     return 0;
}
int anagram(char *array1,char *array2){
    int num1[26] = {0}, num2[26] = {0}, i = 0;

    while (array1[i] != '\0')
    {
        num1[array1[i] - 'a']++;
        i++;
    }
    i = 0;
    while (array2[i] != '\0')
    {
        num2[array2[i] -'a']++;
        i++;
    }
    for (i = 0; i < 26; i++)
    {
        if (num1[i] != num2[i])
            return 0;
    }
    return 1;
}
int isogram(char *str){
     for(int i = 0 ; i<strlen(str) ; i++)
    {
        for(int j = i+1 ; j <strlen(str) ; j++)
        {
            if(str[i] == str[j])
            {
                return 0;
            }
        }
    }
    return 1;
}
// Function designed for chat between client and server.
void func(int connfd)
{
    char buff[MAX];
    int n,a=0,l;
    char b[MAX]; // copy of the string sent from the client

    // infinite loop for chat
    for (;;) {
        bzero(buff, MAX);

        // read the message from client and copy it in buffer
        read(connfd, buff, sizeof(buff));


        // if msg contains "Exit" then server exit and chat ended.
        if (strncmp("exit", buff, 4) == 0)
        {
            printf("Server Exit...\n");
            break;
        }

        char ch = buff[0]; //Contains the info what operation to
perform
        strcpy(b,buff);
        bzero(buff, MAX);
        int i = 0,j = 0;

        for (i = 2; i <strlen(b)-1; i++)
        {
          buff[i-2] = b[i]; // updating the buffer with just the
string to be processed
        }
char buffer[MAX];
        //buff[i-2]=0;
```

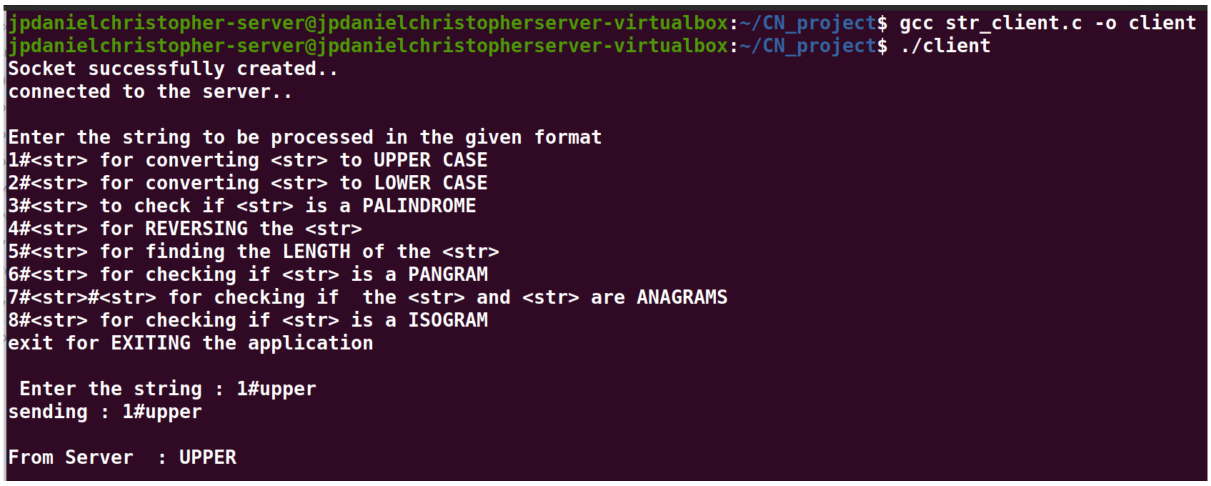


```
if(ch<'7')
printf("From client: %s \n", buff);
if(ch=='7'){
   int i=0;
    int j=0,a=0;
while(buff[i]!='#')
  i++;
  a=i;
  i++;
while(buff[i]){
  buffer[j]=buff[i];
  i++;
  j++;
}
buff[a]=0;
buffer[j]=0;
printf("From client: %s and %s\n", buff,buffer);
}
switch (ch)
{
case '1':
    uppr(buff);
    break;
case '2':
    lwr(buff);
    break;
case '3':
    a = isPalindrome(buff);
    if(a == 1)
    {
        strcpy(buff,"is a PALINDROME");
    }
    else
    {
        strcpy(buff,"is NOT A PALINDROME");
    }
    break;
case '4':
    rev(buff);
    break;
case '5':
    l = len(buff);
```

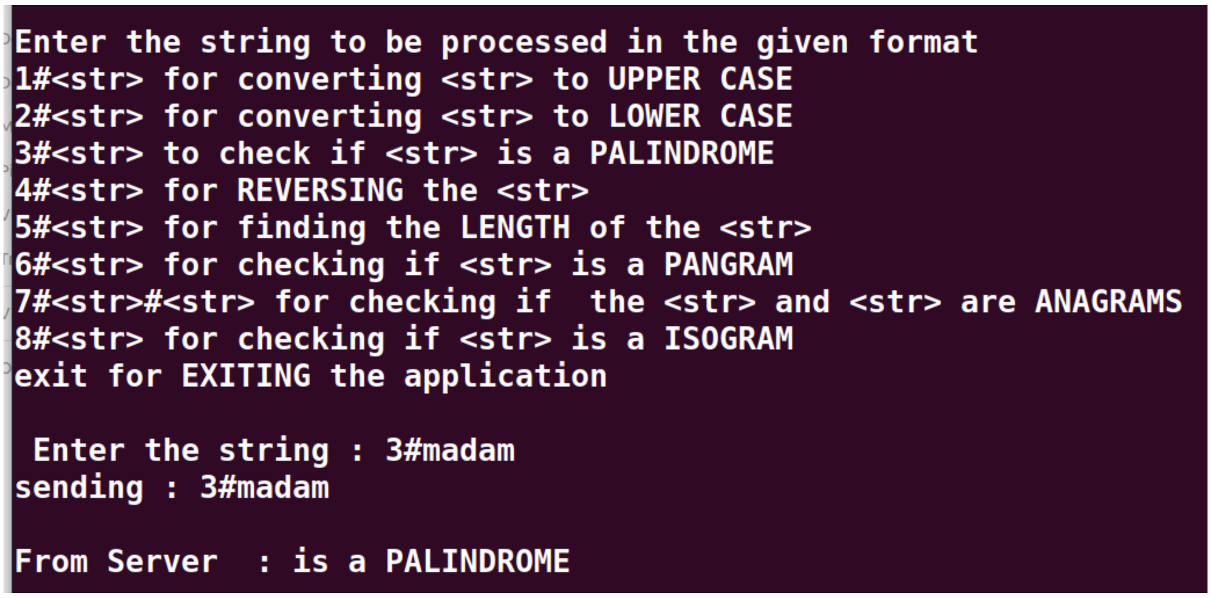
```
            bzero(buff, MAX);
            buff[0] = l+'0';
            //printf("length = %s \n",buff);
            break;
        case '6':
            a = pangram(buff);
            if(a == 1)
            {
                strcpy(buff,"is a PANGRAM");
            }
            else
            {
                strcpy(buff,"is NOT A PANGRAM");
            }
            break;
            case '7':a = anagram(buff,buffer);
            if(a == 1)
            {
                strcpy(buff,"are  ANAGRAMS");
            }
            else
            {
                strcpy(buff,"are not  ANAGRAMS");
            }

            break;
            case '8':
            a = isogram(buff);
            if(a == 1)
            {
                strcpy(buff,"is a ISOGRAM");
            }
            else
            {
                strcpy(buff,"is not  a ISOGRAM");
            }
            break;
            case '#':
            default :
              strcpy(buff,"Invalid option \nEnter choice#<str>
format \n");
              break;
```

```c
        };

        // and send that buffer to client
        write(connfd, buff, sizeof(buff));

    }
}

// Driver function
int main()
{
    int sockfd, connfd, len;
    struct sockaddr_in servaddr, cli;

    // socket create and verification
    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));

    // assign IP, PORT
    servaddr.sin_family = AF_INET;
    servaddr.sin_addr.s_addr = htonl(INADDR_ANY);
    servaddr.sin_port = htons(PORT);

    // Binding newly created socket to given IP and verification
    if ((bind(sockfd, (SA*)&servaddr, sizeof(servaddr))) != 0) {
        printf("socket bind failed...\n");
        exit(0);
    }
    else
        printf("Socket successfully binded..\n");

    // Now server is ready to listen and verification
    if ((listen(sockfd, 5)) != 0) {
        printf("Listen failed...\n");
        exit(0);
    }
    else
        printf("Server listening..\n");
    len = sizeof(cli);

    // Accept the data packet from client and verification
    connfd = accept(sockfd, (SA*)&cli, &len);
    if (connfd < 0) {
        printf("server accept failed...\n");
        exit(0);
    }
    else
        printf("server accept the client...\n");

    // Function for chatting between client and server
    func(connfd);

    // After chatting close the socket
    close(sockfd);
}
```

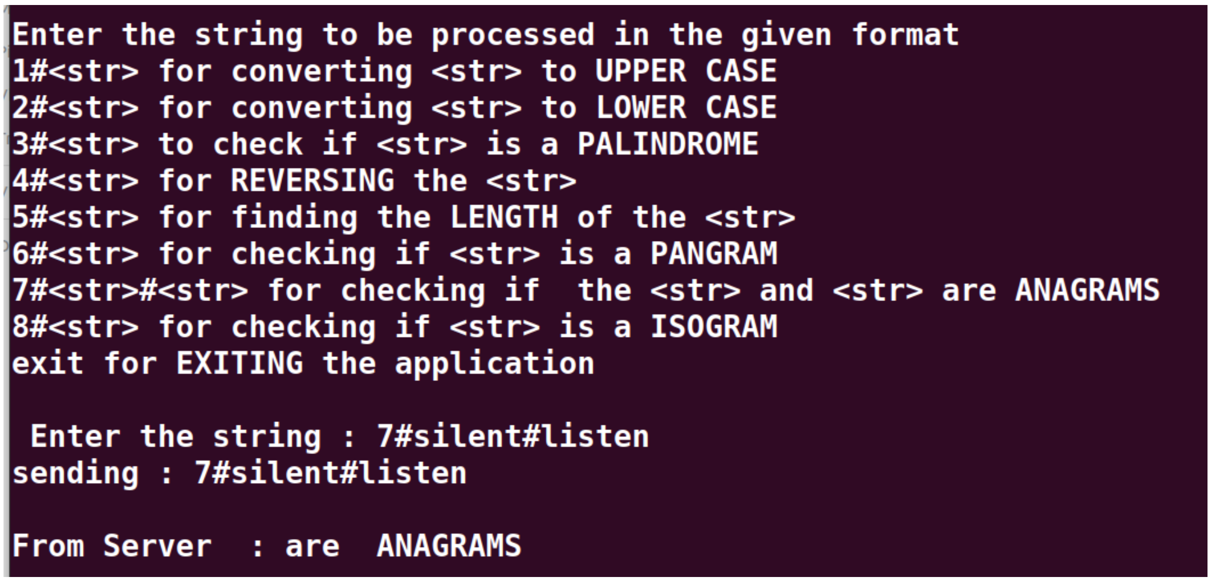
SCREEN SHOTS OF THE OUTPUT:



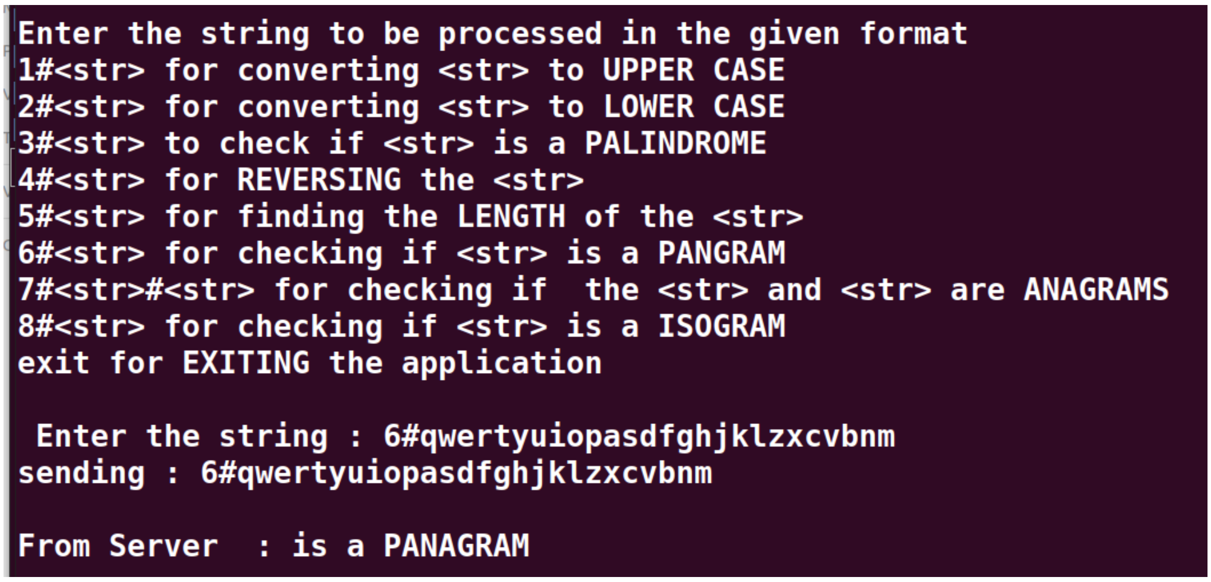
```
jpdanielchristopher-server@jpdanielchristopherserver-virtualbox:~/CN_project$ gcc str_client.c -o client
jpdanielchristopher-server@jpdanielchristopherserver-virtualbox:~/CN_project$ ./client
Socket successfully created..
connected to the server..

Enter the string to be processed in the given format
1#<str> for converting <str> to UPPER CASE
2#<str> for converting <str> to LOWER CASE
3#<str> to check if <str> is a PALINDROME
4#<str> for REVERSING the <str>
5#<str> for finding the LENGTH of the <str>
6#<str> for checking if <str> is a PANGRAM
7#<str>#<str> for checking if  the <str> and <str> are ANAGRAMS
8#<str> for checking if <str> is a ISOGRAM
exit for EXITING the application

 Enter the string : 1#upper
sending : 1#upper

From Server   : UPPER
```



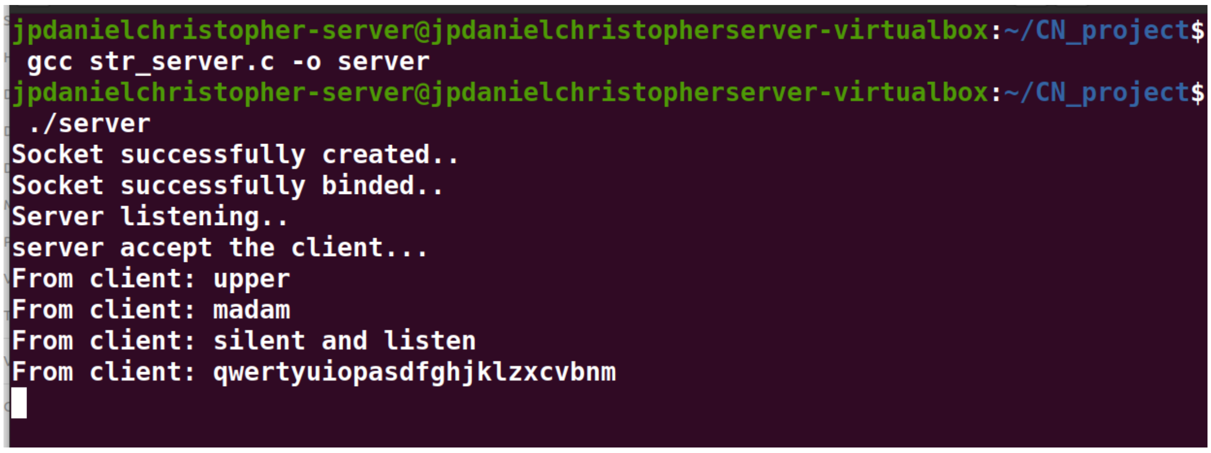
```
Enter the string to be processed in the given format
1#<str> for converting <str> to UPPER CASE
2#<str> for converting <str> to LOWER CASE
3#<str> to check if <str> is a PALINDROME
4#<str> for REVERSING the <str>
5#<str> for finding the LENGTH of the <str>
6#<str> for checking if <str> is a PANGRAM
7#<str>#<str> for checking if  the <str> and <str> are ANAGRAMS
8#<str> for checking if <str> is a ISOGRAM
exit for EXITING the application

 Enter the string : 3#madam
sending : 3#madam

From Server  : is a PALINDROME
```



```
Enter the string to be processed in the given format
1#<str> for converting <str> to UPPER CASE
2#<str> for converting <str> to LOWER CASE
3#<str> to check if <str> is a PALINDROME
4#<str> for REVERSING the <str>
5#<str> for finding the LENGTH of the <str>
6#<str> for checking if <str> is a PANGRAM
7#<str>#<str> for checking if  the <str> and <str> are ANAGRAMS
8#<str> for checking if <str> is a ISOGRAM
exit for EXITING the application

 Enter the string : 7#silent#listen
sending : 7#silent#listen

From Server  : are  ANAGRAMS
```



```
Enter the string to be processed in the given format
1#<str> for converting <str> to UPPER CASE
2#<str> for converting <str> to LOWER CASE
3#<str> to check if <str> is a PALINDROME
4#<str> for REVERSING the <str>
5#<str> for finding the LENGTH of the <str>
6#<str> for checking if <str> is a PANGRAM
7#<str>#<str> for checking if  the <str> and <str> are ANAGRAMS
8#<str> for checking if <str> is a ISOGRAM
exit for EXITING the application

 Enter the string : 6#qwertyuiopasdfghjklzxcvbnm
sending : 6#qwertyuiopasdfghjklzxcvbnm

From Server  : is a PANAGRAM
```



```
jpdanielchristopher-server@jpdanielchristopherserver-virtualbox:~/CN_project$
 gcc str_server.c -o server
jpdanielchristopher-server@jpdanielchristopherserver-virtualbox:~/CN_project$
 ./server
Socket successfully created..
Socket successfully binded..
Server listening..
server accept the client...
From client: upper
From client: madam
From client: silent and listen
From client: qwertyuiopasdfghjklzxcvbnm
```