Name: Preyash Date: 27-01-2022

Registration Number: 20BPS1022

LAB-04

Socket Programming

AIM: To perform the given tasks using the socket programming.

Task 1: Simulate an online English dictionary using sockets. A user searches for meaning of a word in the dictionary which is present in the server. The server responds to the user with the corresponding meaning of the word.

ALGORITHM

Server-side Algorithm

1. Begin by declaring variables.
2. Take the user's input for the port number.
3. Create a TCP socket for the server with socket().
4. Bind the socket to the server address with bind().
5. listen() places the server socket in passive mode, waiting for a client to reach it.
6. accept() establishes a connection between the client and the server in order to send data.
7. Sending a message to the client indicating the connection has been established.
8. Make a structure declaration. The word is stored as a word variable in dictionary, and the variable is stored in mean. Create a total of ten objects and define their variables.

9. Receive the word from the client and see whether it's in any of the objects; if it is, copy the meaning and send it to the msgvariable; if it's not, copy "Meaning not found."
10. Send a message to the client.
11. Stop the programme.

Client-Side Algorithm

1. Begin by declaring variables.
2. Take the user's port as input.
3. Create a TCP socket for the client with socket().
4. Connect the client to the server using connect() to exchange data.
5. Recv() is used to receive data from the server.
6. The data will be restored if the connection is successful.
7. Enter the term you're looking for and submit it to the server.
8. Obtain the message from the server.
9. On the client side, print the definition of the term.
10. Put an end to the programme.

Server Program Source Code:

dict_ser.c
~/Netcom1022/LAB04

```c
#include<stdio.h>
#include<sys/types.h>
#include<netinet/in.h>
#include<string.h>
#include<stdlib.h>

struct dictionary{
    int num;
    char word[100];
    char meaning[200];
}s[10];

int main(){
//    struct dictionary s[10];
    // s[0].num=0;
    strcpy(s[0].word,"A-game");
    strcpy(s[0].meaning,"One’s highest level of performance");

    strcpy(s[1].word,"Anglosphere");
    strcpy(s[1].meaning,"English-speaking countries considered collectively the United Kingdom, the United States, Canada, Australia, and New Zealand, and Ireland");
    strcpy(s[2].word,"Euphoria");
    strcpy(s[2].meaning,"A feeling of intense happiness & excitement.");
    strcpy(s[3].word,"athleisure");
    strcpy(s[3].meaning,"Casual, comfortable clothing or footwear designed to be suitable for both exercise and everyday wear");
    strcpy(s[4].word,"cancel culture");
    strcpy(s[4].meaning,"Call for the withdrawal of support from a public figure, usually in response to an accusation of a socially unacceptable action or comment.");
    strcpy(s[5].word,"Grapple");
    strcpy(s[5].meaning,"wordk hard to come to terms with or deal with something.");
    strcpy(s[6].word,"Peremptory");
    strcpy(s[6].meaning,"Not allowing contradiction or refusal.");
    strcpy(s[7].word,"chipmunky");
    strcpy(s[7].meaning,"Resembling or characteristic of a chipmunk, typically with reference to a person having prominent cheeks or a perky, mischievous character.");
    strcpy(s[8].word,"droning");
    strcpy(s[8].meaning,"The action of using a military drone or a similar commercially available device");
    strcpy(s[9].word,"hench");
    strcpy(s[9].meaning,"Of a person having a powerful, muscular physique; fit, strong");

```

Open | dict_ser.c ~/Netcom1022/LAB04 | Save

```c

    int sd,sd2,nsd,clilen,sport,len;

    char wordd[100];
    char sendmsg[200],rcvmsg[100];
    struct  sockaddr_in servaddr,cliaddr;
    printf("Enter the server port:\n");
    scanf("%d",&sport);
    printf("%d",sport);
    sd=socket(AF_INET,SOCK_STREAM,0);
    if(sd<0)
        printf("Can't create \n");
    else
        printf("Socket is created\n");
    servaddr.sin_family=AF_INET;
    servaddr.sin_addr.s_addr=htonl(INADDR_ANY);
    servaddr.sin_port=htons(sport);

    sd2=bind(sd,(struct  sockaddr*) &servaddr,sizeof(servaddr));
    if(sd2<0)
        printf("Can't bind\n");
    else
        printf(" Binded \n");
    listen(sd,5);
    clilen=sizeof(cliaddr);
    nsd=accept(sd,(struct  sockaddr *)&cliaddr,&clilen);
    if(nsd<0)
        printf("Can't accept\n");
    else
        printf("Accepted\n");

    recv(nsd,wordd,100,0);
    printf("Word Received : %s\n\n",wordd);

    printf("Searching the word in the Dictionary ...........\n");
    int flag=0;
```

```c
    for(int i=0;i<10;i++)
    {
        if(strcmp(wordd,s[i].word)==0)
        {
            // printf("%s\n",s[i].meaning);
            strcpy(sendmsg,s[i].meaning);
            flag=1;
            break;
        }
    }
    if(flag== 0 )
    {
        strcpy(sendmsg,"meaninging not found");
    }
    send(nsd,sendmsg,200,0);

    return 0;
}
```

Client Program Source Code:

dict_cli.c
~/Netcom1022/LAB04

```c
#include<stdio.h>
#include<sys/types.h>
#include<netinet/in.h>
#include<string.h>

int main(){
    int csd,cport,len;
    char wor[100];
    char sendmsg[100],revmsg[200];
    char mean[200];
    struct  sockaddr_in  servaddr;
    printf("Enter the port \n");
    scanf("%d",&cport);
    printf("%d",cport);
    csd=socket(AF_INET,SOCK_STREAM,0);
    if(csd<0)
        printf("Can't create\n");
    else
        printf("Socket is created\n");
    servaddr.sin_family=AF_INET;
    servaddr.sin_addr.s_addr=htonl(INADDR_ANY);
    servaddr.sin_port=htons(cport);

    if(connect(csd,(struct sockaddr *)&servaddr,sizeof(servaddr))<0)
        printf("Can't  connect\n");
    else
        printf("Connected sucessfully\n");
    scanf("%s",wor);
    send(csd,wor,100,0);
    recv(csd,revmsg,200,0);
    printf("\n");
    printf("wor : %s\n",wor);
    printf("Meaning : %s\n",revmsg);
    return 0;
}
```

Output:

Server-side:

preyash-20bps1022@Preyash-20BPS1... preyash-20bps1022@Preyash-20BPS1...

```
preyash-20bps1022@Preyash-20BPS1022:~/Netcom1022/LAB04$ gcc dict_ser.c
preyash-20bps1022@Preyash-20BPS1022:~/Netcom1022/LAB04$ ./a.out
Enter the server port:
1029
1029Socket is created
 Binded
Accepted
Word Received : A-game

Searching the word in the Dictionary ...........
preyash-20bps1022@Preyash-20BPS1022:~/Netcom1022/LAB04$
```

Client-side:

```
preyash-20bps1022@Preyash-20BPS1...   preyash-20bps1022@Preyash-20BPS1...
preyash-20bps1022@Preyash-20BPS1022:~/Netcom1022/LAB04$ gcc dict_cli.c
preyash-20bps1022@Preyash-20BPS1022:~/Netcom1022/LAB04$ ./a.out
Enter the port
1029
1029Socket is created
Connected sucessfully
A-game

wor : A-game
Meaning : One’s highest level of performance
preyash-20bps1022@Preyash-20BPS1022:~/Netcom1022/LAB04$
```

Task 2: Develop a “Remote Calculator” application that works as follows: The client program inputs two integers and an arithmetic operation (‘*’;’/;’+’; ’-’) from the user and sends these three values to the server side. The server does the given operation or the two integers and sends back the result of the operation on the two integers and sends back the result of the operation to the client. Help server to implement this scenario using connection-oriented sockets.

ALGORITHM

Server-side Algorithm

1. Begin by declaring variables.
2. Take the user's input for the port number.
3. Create a TCP socket for the server with socket().
4. Bind the socket to the server address with bind().
5. listen() places the server socket in passive mode, waiting for a client to reach it.
6. accept() establishes a connection between the client and the server in order to send data.
7. Sending a message to the client indicating the connection has been established.
8. From the server, get the two numbers and the operation you wish to execute.
9. Calculate the result as needed.
10. Send a message to the client.
11. Stop the programme.

Client-Side Algorithm

1. Begin by declaring variables.
2. Take the user's port as input.
3. Create a TCP socket for the client with socket().
4. Connect the client to the server using connect() to exchange data.
5. Recv() is used to receive data from the server.
6. Send the values to the server for calculation.
7. Receive the result from the server.
8. Print the output.
9. End the program.

Server Program Source Code:

Open *calc_ser.c ~/Netcom1022/LAB04 Save

```c
#include<stdio.h>
#include<sys/types.h>
#include<netinet/in.h>
#include<string.h>
#include<stdlib.h>
#include<string.h>

void main(){
    int sd,sd2,nsd,clilen,sport,len;

    char a[5];
    char b[5];
    char c[5];
    char sendmsg[100],rcvmsg[100];
    struct  sockaddr_in servaddr,cliaddr;
    printf("Enter the server port:\n");
    scanf("%d",&sport);
    printf("Connected to port %d\n",sport);
    sd=socket(AF_INET,SOCK_STREAM,0);
    if(sd<0)
        printf("Can't create \n");
    else
        printf("Socket is created\n");
    servaddr.sin_family=AF_INET;
    servaddr.sin_addr.s_addr=htonl(INADDR_ANY);
    servaddr.sin_port=htons(sport);

    sd2=bind(sd,(struct  sockaddr*) &servaddr,sizeof(servaddr));
    if(sd2<0)
        printf("Can't bind\n");
    else
        printf("Binded \n");
    listen(sd,5);
    clilen=sizeof(cliaddr);
    nsd=accept(sd,(struct  sockaddr *)&cliaddr,&clilen);
    if(nsd<0)
        printf("Can't accept\n");
    else
        printf("Accepted\n");

```

```c
    recv(nsd,a,5,0);
    recv(nsd,b,5,0);
    recv(nsd,c,5,0);

    int x=atoi(a);
    int y=atoi(b);

    int z=0;
    printf("Data Received\n");
    printf("Performing the operation.....\n");

    if(strcmp(c,"+")==0)
    {
        z=x+y;
    }
    if(strcmp(c,"-")==0)
    {
        z=x-y;
    }if(strcmp(c,"*")==0)
    {
        z=x*y;
    }
    if(strcmp(c,"/")==0)
    {
        z=x/y;
    }

    printf("Result sent : %d\n",z);
    sprintf(sendmsg,"%d",z);
    send(nsd,sendmsg,100,0);
}

```

Client Program Source Code:

Open calc_cli.c ~/Netcom1022/LAB04 Save

```c
#include<stdio.h>
#include<sys/types.h>
#include<netinet/in.h>
#include<string.h>

void main(){
    int csd,cport,len;
    char sendmsg[100],revmsg[100];
    struct  sockaddr_in  servaddr;
    printf("Enter the port \n");
    scanf("%d",&cport);
    printf("%d\n",cport);
    csd=socket(AF_INET,SOCK_STREAM,0);
    if(csd<0)
        printf("Can't create\n");
    else
        printf("Socket is created\n");
    servaddr.sin_family=AF_INET;
    servaddr.sin_addr.s_addr=htonl(INADDR_ANY);
    servaddr.sin_port=htons(cport);

    if(connect(csd,(struct sockaddr *)&servaddr,sizeof(servaddr))<0)
        printf("Can't  connect\n");
    else
        printf("Connected sucessfully\n");
    char aa[5];
    char bb[5];
    char cc[5];
    int a;
    int b;
    printf("Enter the 1st number: ");
    scanf("%d",&a);
    printf("Enter the 2nd number: ");
    scanf("%d",&b);
    printf("Enter the operation : ");
    scanf("%s",cc);
    sprintf(aa,"%d",a);
    sprintf(bb,"%d",b);
    send(csd,aa,5,0);
    send(csd,bb,5,0);
    send(csd,cc,5,0);
```

```
    send(csd,cc,5,0);
    recv(csd,revmsg,100,0);
    printf("The answer received from the server side is: %s\n",revmsg);
}
```

Output:

Server-side:

```
preyash-20bps1022@Preyash-20BPS1...   preyash-20bps1022@Preyash-20BPS1...
preyash-20bps1022@Preyash-20BPS1022:~/Netcom1022/LAB04$ gedit calc_ser.c
preyash-20bps1022@Preyash-20BPS1022:~/Netcom1022/LAB04$ gcc calc_ser.c
preyash-20bps1022@Preyash-20BPS1022:~/Netcom1022/LAB04$ ./a.out
Enter the server port:
1029
Connected to port 1029
Socket is created
Binded
Accepted
Data Received
Performing the operation.....
Result sent : 30
preyash-20bps1022@Preyash-20BPS1022:~/Netcom1022/LAB04$
```

Client-side:

```
preyash-20bps1022@Preyash-20BPS1...   preyash-20bps1022@Preyash-20BPS1...
preyash-20bps1022@Preyash-20BPS1022:~/Netcom1022/LAB04$ gedit calc_cli.c
preyash-20bps1022@Preyash-20BPS1022:~/Netcom1022/LAB04$ gcc calc_cli.c
preyash-20bps1022@Preyash-20BPS1022:~/Netcom1022/LAB04$ ./a.out
Enter the port
1029
1029
Socket is created
Connected sucessfully
Enter the 1st number: 01
Enter the 2nd number: 29
Enter the operation : +
The answer received from the server side is: 30
preyash-20bps1022@Preyash-20BPS1022:~/Netcom1022/LAB04$
```