|  |  |  |  |
| --- | --- | --- | --- |
|  | Bansilal Ramnath Agarwal Charitable Trust's  Vishwakarma Institute of Information Technology  **Department of Artificial Intelligence and Data**  **Science** | | |
| Name: Siddhesh Dilip Khairnar | | | |
| Class: SY | Division: B | | Roll No: 272028 |
| Semester: IV | | Academic Year: 2022-2023 | |
| Subject Name & Code: Fundamentals of Computer Networks: ADUA22203 | | | |
| Title of Assignment: Write a program using TCP socket for wired network for following:  a. Say Hello to Each other b File transfer.  c. Calculator. | | | |
| Date of Performance: 27-04-2023 | | Date of Submission: 27-04-2023 | |

# Assignment No- 7

**Program and Output:**

# Say Hello to each other:

## Client:

#include <stdio.h> #include <sys/socket.h> #include <netinet/in.h> #include <string.h>

int main()

{

int clientSocket; char buffer[1024];

struct sockaddr\_in serverAddr; socklen\_t addr\_size;

/\*—- Create the socket. The three arguments are: —-\*/

/\* 1) Internet domain 2) Stream socket 3) Default protocol (TCP in this case) \*/ clientSocket = socket(PF\_INET, SOCK\_STREAM, 0);

/\*—- Configure settings of the server address struct —-\*/

/\* Address family = Internet \*/ serverAddr.sin\_family = AF\_INET;

/\* Set port number, using htons function to use proper byte order \*/ serverAddr.sin\_port = htons(7891);

/\* Set IP address to localhost \*/ serverAddr.sin\_addr.s\_addr = inet\_addr("127.0.0.1");

/\* Set all bits of the padding field to 0 \*/ memset(serverAddr.sin\_zero, '\0', sizeof serverAddr.sin\_zero);

/\*—- Connect the socket to the server using the address struct —-\*/ addr\_size = sizeof serverAddr;

connect(clientSocket, (struct sockaddr \*) &serverAddr, addr\_size);

/\*—- Read the message from the server into the buffer —-\*/ recv(clientSocket, buffer, 1024, 0);

/\*—- Print the received message —-\*/ printf("Data received: %s",buffer);

return 0;

}

## Server:

#include <stdio.h> #include <sys/socket.h> #include <netinet/in.h> #include <string.h>

int main()

{

int welcomeSocket, newSocket; char buffer[1024];

struct sockaddr\_in serverAddr;

struct sockaddr\_storage serverStorage; socklen\_t addr\_size;

/\*—- Create the socket. The three arguments are: —-\*/

/\* 1) Internet domain 2) Stream socket 3) Default protocol (TCP in this case) \*/ welcomeSocket = socket(PF\_INET, SOCK\_STREAM, 0);

/\*—- Configure settings of the server address struct —-\*/

/\* Address family = Internet \*/ serverAddr.sin\_family = AF\_INET;

/\* Set port number, using htons function to use proper byte order \*/ serverAddr.sin\_port = htons(7891);

/\* Set IP address to localhost \*/ serverAddr.sin\_addr.s\_addr = inet\_addr("127.0.0.1");

/\* Set all bits of the padding field to 0 \*/ memset(serverAddr.sin\_zero, '\0', sizeof serverAddr.sin\_zero);

/\*—- Bind the address struct to the socket —-\*/

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

/\*—- Listen on the socket, with 5 max connection requests queued —-\*/ if(listen(welcomeSocket,5)==0)

printf("Listening\n"); else printf("Error\n");

/\*—- Accept call creates a new socket for the incoming connection —-\*/ addr\_size = sizeof serverStorage;

newSocket = accept(welcomeSocket, (struct sockaddr \*) &serverStorage, &addr\_size);

/\*—- Send message to the socket of the incoming connection —-\*/ strcpy(buffer,"Hello World\n");

send(newSocket,buffer,13,0);

return 0;

}



# File Transfer:

## Client:

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

int main(void)

{

int sockfd = 0;

int bytesReceived = 0; char recvBuff[256];

memset(recvBuff, '0', sizeof(recvBuff)); struct sockaddr\_in serv\_addr;

/\* Create a socket first \*/

if((sockfd = socket(AF\_INET, SOCK\_STREAM, 0))< 0)

{

printf("\n Error : Could not create socket \n"); return 1;

}

/\* Initialize sockaddr\_in data structure \*/ serv\_addr.sin\_family = AF\_INET; serv\_addr.sin\_port = htons(5000); // port

serv\_addr.sin\_addr.s\_addr = inet\_addr("172.16.6.168");

/\* Attempt a connection \*/

if(connect(sockfd, (struct sockaddr \*)&serv\_addr, sizeof(serv\_addr))<0)

{

printf("\n Error : Connect Failed \n"); return 1;

}

/\* Create file where data will be stored \*/ FILE \*fp;

fp = fopen("sample\_file.txt", "ab"); if(NULL == fp)

{

printf("Error opening file"); return 1;

}

/\* Receive data in chunks of 256 bytes \*/ while((bytesReceived = read(sockfd, recvBuff, 256)) > 0)

{

printf("Bytes received %d\n",bytesReceived);

// recvBuff[n] = 0;

fwrite(recvBuff, 1,bytesReceived,fp);

// printf("%s \n", recvBuff);

}

if(bytesReceived < 0)

{

printf("\n Read Error \n");

}

return 0;

}

## Server:

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

int main(void)

{

int listenfd = 0; int connfd = 0;

struct sockaddr\_in serv\_addr; char sendBuff[1024];

int numrv;

listenfd = socket(AF\_INET, SOCK\_STREAM, 0); printf("Socket retrieve success\n");

memset(&serv\_addr, '0', sizeof(serv\_addr));

memset(sendBuff, '0', sizeof(sendBuff));

serv\_addr.sin\_family = AF\_INET; serv\_addr.sin\_addr.s\_addr = htonl(INADDR\_ANY); serv\_addr.sin\_port = htons(5000);

bind(listenfd, (struct sockaddr\*)&serv\_addr,sizeof(serv\_addr)); if(listen(listenfd, 10) == -1)

{

printf("Failed to listen\n"); return -1;

}

while(1)

{

connfd = accept(listenfd, (struct sockaddr\*)NULL ,NULL);

/\* Open the file that we wish to transfer \*/ FILE \*fp = fopen("sample\_file.txt","rb"); if(fp==NULL)

{

printf("File opern error"); return 1;

}

/\* Read data from file and send it \*/ while(1)

{

/\* First read file in chunks of 256 bytes \*/ unsigned char buff[256]={0};

int nread = fread(buff,1,256,fp); printf("Bytes read %d \n", nread);

/\* If read was success, send data. \*/ if(nread > 0)

{

printf("Sending \n");

write(connfd, buff, nread);

}

/\*

* There is something tricky going on with read ..
* Either there was error, or we reached end of file.

\*/

if (nread < 256)

{

if (feof(fp))

printf("End of file\n"); if (ferror(fp))

printf("Error reading\n"); break;

}

}

close(connfd); sleep(1);

}

return 0;

}



