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:

A) 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;
}
```

```
/*OUTPUT CLIENT
iotlab@iotlab-Veriton-M200-B360:~$ cd TCP\ Socket/
iotlab@iotlab-Veriton-M200-B360:~/TCP Socket$ cd Simple\ Hello/
iotlab@iotlab-Veriton-M200-B360:~/TCP Socket/Simple Hello$ gcc client_simple_hello.c -o client
client_simple_hello.c: In function ‘main’:
client_simple_hello.c:23:30: warning: implicit declaration of function ‘inet_addr’; did you mean
‘s6_addr’? [-Wimplicit-function-declaration]
 serverAddr.sin_addr.s_addr = inet_addr("127.0.0.1");
                              ^~~~~~~~~
                              s6_addr
iotlab@iotlab-Veriton-M200-B360:~/TCP Socket/Simple Hello$ ./client
Data received: Hello World


OUTPUT SERVER
iotlab@iotlab-Veriton-M200-B360:~/TCP Socket/Simple Hello$ gcc server_simple_hello.c -o
server
server_simple_hello.c: In function ‘main’:
server_simple_hello.c:24:30: warning: implicit declaration of function ‘inet_addr’; did you mean
‘s6_addr’? [-Wimplicit-function-declaration]
 serverAddr.sin_addr.s_addr = inet_addr("127.0.0.1");
                              ^~~~~~~~~
                              s6_addr
iotlab@iotlab-Veriton-M200-B360:~/TCP Socket/Simple Hello$ ./server
Listening
*/
```

B) 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;
}

/*OUTPUT SERVER
iotlab@iotlab-Veriton-M200-B360:~$ cd TCP\ Socket/
iotlab@iotlab-Veriton-M200-B360:~/TCP Socket$ cd File\ Transfer/
iotlab@iotlab-Veriton-M200-B360:~/TCP Socket/File Transfer$ gcc Server_file.c -o server
iotlab@iotlab-Veriton-M200-B360:~/TCP Socket/File Transfer$ ./server
Socket retrieve success
Bytes read 0
End of file

OUTPUT CLIENT
iotlab@iotlab-Veriton-M200-B360:~/TCP Socket/File Transfer$ gcc Client_file.c -o client
iotlab@iotlab-Veriton-M200-B360:~/TCP Socket/File Transfer$ ./clientiotlab@iotlab-Veriton-M200-
B360:~/TCP Socket/File Transfer$
*/
```

```
/*OUTPUT SERVER
iotlab@iotlab-Veriton-M200-B360:~$ cd TCP\ Socket/
iotlab@iotlab-Veriton-M200-B360:~/TCP Socket$ cd Arithmetic/
iotlab@iotlab-Veriton-M200-B360:~/TCP Socket/Arithmetic$ gcc server_arithmetic.c -o ser
iotlab@iotlab-Veriton-M200-B360:~/TCP Socket/Arithmetic$ ./ser
socket created sucessfully
bind sucessful
listen sucessful
accept sucessful
Result is: 10 + 15 = 25

iotlab@iotlab-Veriton-M200-B360:~/TCP Socket/Arithmetic$

OUTPUT CLIENT
iotlab@iotlab-Veriton-M200-B360:~$ cd TCP\ Socket/
iotlab@iotlab-Veriton-M200-B360:~/TCP Socket$ cd Arithmetic/
iotlab@iotlab-Veriton-M200-B360:~/TCP Socket/Arithmetic$ gcc client_arithmetic.c -o cl
iotlab@iotlab-Veriton-M200-B360:~/TCP Socket/Arithmetic$ ./cl
socket created sucessfully
connect sucessful
Enter operation:
 + Addition
 -: Subtraction
 /: Division
 * Multiplication
+
Enter operands:
10
15
Operation result from server=25

iotlab@iotlab-Veriton-M200-B360:~/TCP Socket/Arithmetic$
*/
```

Conclusion: successfully learn & write the program using TCP socket for wired network for say Hello to each other, file transfer and calculator.

Ashwin
27/4/23