```
case 7: {
    HANDLE hProcess;
    HANDLE hThread;
    STARTUPINFO si;
    PROCESS_INFORMATION pi;
    DWORD dwProcessId = 0;

    ZeroMemory( &si, sizeof(si) );
    ZeroMemory( &pi, sizeof(pi) );

    char pname[20];
    //printf("Enter a name of process: ");
    scanf("%s", pname);
    BOOL bCreateProcess = NULL;
    char p_path[] = "C:\\Users\\Bagaa\\OneDrive\\Documents\\2021-2022-autumn-semister\\TOS\\Sem03\\";
    strcat(p_path, pname);
    bCreateProcess = CreateProcess(
        NULL,
        p_path,
```

```
        strcat(p_path, pname);
        bCreateProcess = CreateProcess(
            NULL,
            p_path,
            NULL,
            NULL,
            FALSE,
            0,
            NULL,
            NULL,
            &si,
            &pi);

        // Start the child process.
        if(bCreateProcess){
            printf("CreateProcess successfully.\n");
        }else{
            printf( "CreateProcess failed (%d).\n", GetLastError() );
        }

        // Wait until child process exits.
        WaitForSingleObject( pi.hProcess, INFINITE );

        // Close process and thread handles.
        CloseHandle( pi.hProcess );
        CloseHandle( pi.hThread );
        break;
    }
```

```
Program is started. Enter command:
>> help
Defined commands:
        >> help
        >> create filename.txt
        >> insert filename.txt sample data
        >> rename newname filename.txt
        >> del filename.txt
        >> dir
        >> createProcess process_name
>> createProcess hello.exe
CreateProcess successfully.
Hello, This is Sem03.
>> createProcess printData.exe
CreateProcess successfully.
Enter a name: Bagaa
Enter a age: 20
Your data:
Name: Bagaa, Age: 20
```

```
>> dir
.
..
client.c
hello.c
hello.exe
hello.o
printData.c
printData.exe
printData.o
process.c
process.exe
process.o
server.c
TOS_Lab03.c
TOS_Lab03.exe
TOS_Lab03.o
```

client.c server.c

```c
#include <winsock2.h>'
#include <windows.h>
#include <stdio.h>
#include <string.h>

#pragma comment(lib,"ws2_32.dll") //Winsock Library

int main(){
    printf("\t\t---------- TCP SERVER ----------\n");
    //local variable
    WSADATA Winsockdata;
    int iWsaStartup;
    int iWsaCleanup;

    SOCKET TCPServerSocket;
    int iCloseSocket;

    struct sockaddr_in TCPServerAdd;
    struct sockaddr_in TCPClientAdd;
    int iTCPClientAdd = sizeof(TCPClientAdd);

    int iBind;

    int iListen;

    SOCKET sAcceptSocket;

    int iSend;
```

client.c server.c

```
28      int iSend;
29      char SenderBuffer[512];
30      printf("Enter a messages: ");
31      gets(SenderBuffer);
32      int iSenderBuffer = strlen(SenderBuffer) + 1;
33
34      int iRecv;
35      char RecvBuffer[512];
36      int iRecvBuffer = strlen(RecvBuffer) + 1;
37
38      //STEP-1 WSAStartup Fun
39      iWsaStartup = WSAStartup(MAKEWORD(2,2), &Winsockdata);
40      if(iWsaStartup != 0){
41          printf("WSAStartUp Failed\n");
42      }else
43          printf("WSAStartUp Success\n");
44
45      //STEP-2 Fill the Structure
46      TCPServerAdd.sin_family = AF_INET;
47      TCPServerAdd.sin_addr.s_addr = inet_addr("127.0.0.1");
48      TCPServerAdd.sin_port = htons(8000);
49
50      //STEP-3 Socket Creation
51      TCPServerSocket = socket(AF_INET, SOCK_STREAM, IPPROTO_TCP);
52      if(TCPServerSocket == INVALID_SOCKET){
53          printf("TCP Server Socket Creation Failed\n");
54      }else
55          printf("TCP Server Socket Creation Success\n");
```

client.c server.c

```
55          printf("TCP Server Socket Creation Success\n");
56
57      //STEP-4 Bind Fun
58      iBind = bind(TCPServerSocket, (SOCKADDR*)&TCPServerAdd, sizeof(TCPServerAdd));
59      if(iBind == SOCKET_ERROR){
60          printf("Binding Failed & Error No -> %d\n", WSAGetLastError());
61      }else
62          printf("Binding Success\n");
63
64      //STEP-5 Listen Fun
65      iListen = listen(TCPServerSocket, 2);
66      if(iListen == SOCKET_ERROR){
67          printf("Listen Fun failed & Error No -> %d\n", WSAGetLastError());
68      }else
69          printf("Listen Fun success\n");
70
71      //STEP-6 Accept
72      sAcceptSocket = accept(TCPServerSocket, (SOCKADDR*)&TCPClientAdd, &iTCPClientAdd);
73      if(sAcceptSocket == INVALID_SOCKET){
74          printf("Accept Failed & Error No -> %d\n", WSAGetLastError());
75      }else
76          printf("Connection Accepted\n");
77
78      //STEP-7 Send Data to Client
79      iSend = send(sAcceptSocket, SenderBuffer, iSenderBuffer, 0);
80      if(iSend == SOCKET_ERROR){
81          printf("Sending Failed & Error No -> %d\n", WSAGetLastError());
82      }else
```

client.c server.c

```
80      if(iSend == SOCKET_ERROR){
81          printf("Sending Failed & Error No -> %d\n", WSAGetLastError());
82      }else
83          printf("Data Sending Success\n");
84
85      //STEP-8 Recv Data from Client
86      iRecv = recv(sAcceptSocket, RecvBuffer, iRecvBuffer, 0);
87      if(iRecv == SOCKET_ERROR){
88          printf("Recieve Data Failed & Error No -> %d\n", WSAGetLastError());
89      }else
90          printf("Data Recieved Success\n");
91
92      //STEP-9 Close Socket
93      iCloseSocket = closesocket(TCPServerSocket);
94      if(iCloseSocket == SOCKET_ERROR){
95          printf("Closing Socket Failed & Error No -> %d\n", WSAGetLastError());
96      }else
97          printf("Closing Socket Success\n");
98
99      //STEP-10 CleanUp from DLL
100     iWsaCleanup = WSACleanup();
101     if(iWsaCleanup == SOCKET_ERROR){
102         printf("CleanUp Fun Failed & Error No -> %d\n", WSAGetLastError());
103     }else
104         printf("CleanUp Fun Success\n");
105     return 0;
106 }
```

client.c server.c

```c
#include <winsock2.h>
#include <windows.h>
#include <stdio.h>
#include <string.h>


#pragma comment(lib,"ws2_32.dll") //Winsock Library

int main(){
    printf("\t\t---------- TCP CLIENT ----------\n");
    //local variable
    WSADATA WinSockData;
    int iWsaStartup;
    int iWsaCleanup;

    SOCKET TCPClientSocket;
    int iCloseSocket;

    struct sockaddr_in TCPServerAdd;

    int iConnect;

    int iSend;
    char SenderBuffer[512];
    printf("Enter a messages: ");
    gets(SenderBuffer);
    int iSenderBuffer = strlen(SenderBuffer) + 1;
```

client.c server.c

```
28
29      int iRecv;
30      char RecvBuffer[512];
31      int iRecvBuffer = strlen(RecvBuffer) + 1;
32
33      //STEP-1 WSAStartup Fun
34      iWsaStartup = WSAStartup(MAKEWORD(2,2), &WinSockData);
35      if(iWsaStartup != 0){
36          printf("WSAStartUp Failed\n");
37      }else
38          printf("WSAStartUp Success\n");
39
40      //STEP-2 Fill the Structure
41      TCPServerAdd.sin_family = AF_INET;
42      TCPServerAdd.sin_addr.s_addr = inet_addr("127.0.0.1");
43      TCPServerAdd.sin_port = htons(8000);
44
45      //STEP-3 Socket Creation
46      TCPClientSocket = socket(AF_INET, SOCK_STREAM, IPPROTO_TCP);
47      if(TCPClientSocket == INVALID_SOCKET){
48          printf("TCP Server Socket Creation Failed\n");
49      }else
50          printf("TCP Server Socket Creation Success\n");
51
52      //STEP-4 Connect Fun
53      iConnect = connect(TCPClientSocket, (SOCKADDR*)&TCPServerAdd, sizeof(TCPServerAdd));
54      if(iConnect == SOCKET_ERROR){
55          printf("Connection Failed & Error No -> %d\n", WSAGetLastError());
```

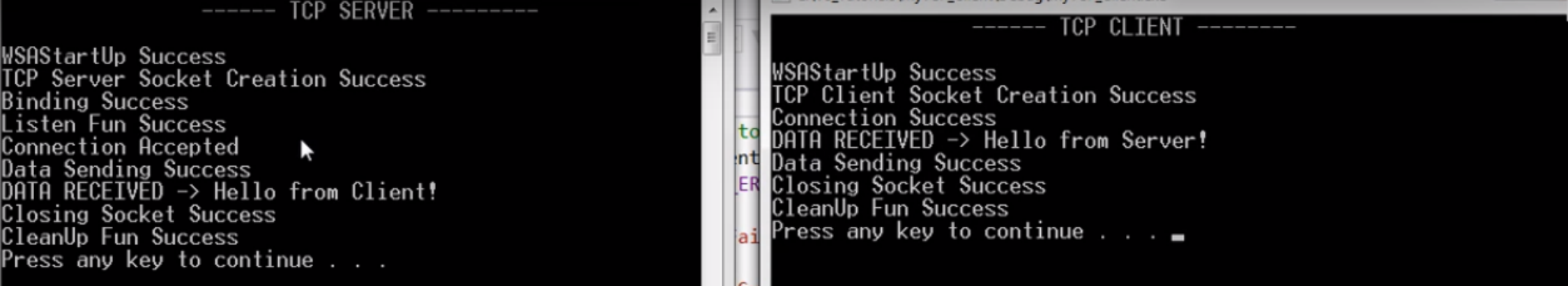
client.c server.c

```
55          printf("Connection Failed & Error No -> %d\n", WSAGetLastError());
56      }else
57          printf("Connection Success\n");
58
59      //STEP-5 Send Data to Client
60      iSend = send(TCPClientSocket, SenderBuffer, iSenderBuffer, 0);
61      if(iSend == SOCKET_ERROR){
62          printf("Sending Failed & Error No -> %d\n", WSAGetLastError());
63      }else
64          printf("Data Sending Success\n");
65
66      //STEP-6 Recv Data from Client
67      iRecv = recv(TCPClientSocket, RecvBuffer, iRecvBuffer, 0);
68      if(iRecv == SOCKET_ERROR){
69          printf("Recieve Data Failed & Error No -> %d\n", WSAGetLastError());
70      }else
71          printf("Data Recieved Success\n");
72
73
74      //STEP-7 Close Socket
75      iCloseSocket = closesocket(TCPClientSocket);
76      if(iCloseSocket == SOCKET_ERROR){
77          printf("Closing Socket Failed & Error No -> %d\n", WSAGetLastError());
78      }else
79          printf("Closing Socket Success\n");
80
81      //STEP-8 CleanUp from DLL
82      iWsaCleanup = WSACleanup();
```

```
82      iWsaCleanup = WSACleanup();
83      if(iWsaCleanup == SOCKET_ERROR){
84          printf("CleanUp Fun Failed & Error No -> %d\n", WSAGetLastError());
85      }else
86          printf("CleanUp Fun Success\n");
87      return 0;
88  }
```

```
Program is started. Enter command:
>> help
Defined commands:
        >> help
        >> create filename.txt
        >> insert filename.txt sample data
        >> rename newname filename.txt
        >> del filename.txt
        >> dir
        >> createProcess process_name
>> createProcess server.exe
CreateProcess successfully.
               ---------- TCP SERVER ----------
Enter a messages: Hello from Server!
```

```
Program is started. Enter command:
>> help
Defined commands:
        >> help
        >> create filename.txt
        >> insert filename.txt sample data
        >> rename newname filename.txt
        >> del filename.txt
        >> dir
        >> createProcess process_name
>> createProcess client.exe
CreateProcess successfully.
              ---------- TCP CLIENT ----------
Enter a messages: Hello from Client!
```


------ TCP SERVER ---------
WSAStartUp Success
TCP Server Socket Creation Success
Binding Success
Listen Fun Success
Connection Accepted
Data Sending Success
DATA RECEIVED -> Hello from Client!
Closing Socket Success
CleanUp Fun Success
Press any key to continue . . .
------ TCP CLIENT --------
WSAStartUp Success
TCP Client Socket Creation Success
Connection Success
DATA RECEIVED -> Hello from Server!
Data Sending Success
Closing Socket Success
CleanUp Fun Success
Press any key to continue . . .

```
>> dir
.
..
client.c
client.exe
client.o
hello.c
hello.exe
hello.o
printData.c
printData.exe
printData.o
process.c
process.exe
process.o
server.c
server.exe
server.o
TOS_Lab03.c
TOS_Lab03.exe
TOS_Lab03.o
```