4. Homework

12201922 이규민

0) Open two Cygwin/terminal windows and type cli.c and serv.c at each window (the code is in Section 1).

[cli.c]

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

#define SERV_TCP_PORT 34357
#define SERV_ADDR "165.246.222.68"

int main(){
    int x,y;s
    struct sockaddr_in  serv_addr;
    char buf[50];

    printf("Hi, I am the client\n");

    bzero((char *) &serv_addr, sizeof(serv_addr));
    serv_addr.sin_family = PF_INET;
    serv_addr.sin_addr.s_addr = inet_addr(SERV_ADDR);
    serv_addr.sin_port = htons(SERV_TCP_PORT);
```

```c
    /* open a tcp socket*/
    if ( (x =socket(PF_INET, SOCK_STREAM,0)) < 0){
        printf("socket creation error\n");
        exit(1);
        }
    printf(" socket opened successfully. socket num is %d\n", x);

    /* connect to  the server */
    if (connect(x, (struct sockaddr *) &serv_addr, sizeof(serv_addr))<0){
        printf("can't connect to the server\n");
        exit(1);
    }

    /* send input str to the server */
    printf("now i am connected to the erver. enter a string to send\n");
    scanf("%s", buf);
    write(x,buf,strlen(buf));

    printf("now let's read from the server\n");
    y=read(x,buf,50);
    buf[y]=0;
    printf("what echoed from the server is %s\n",buf);
    close(x);    // disconnect
}
```

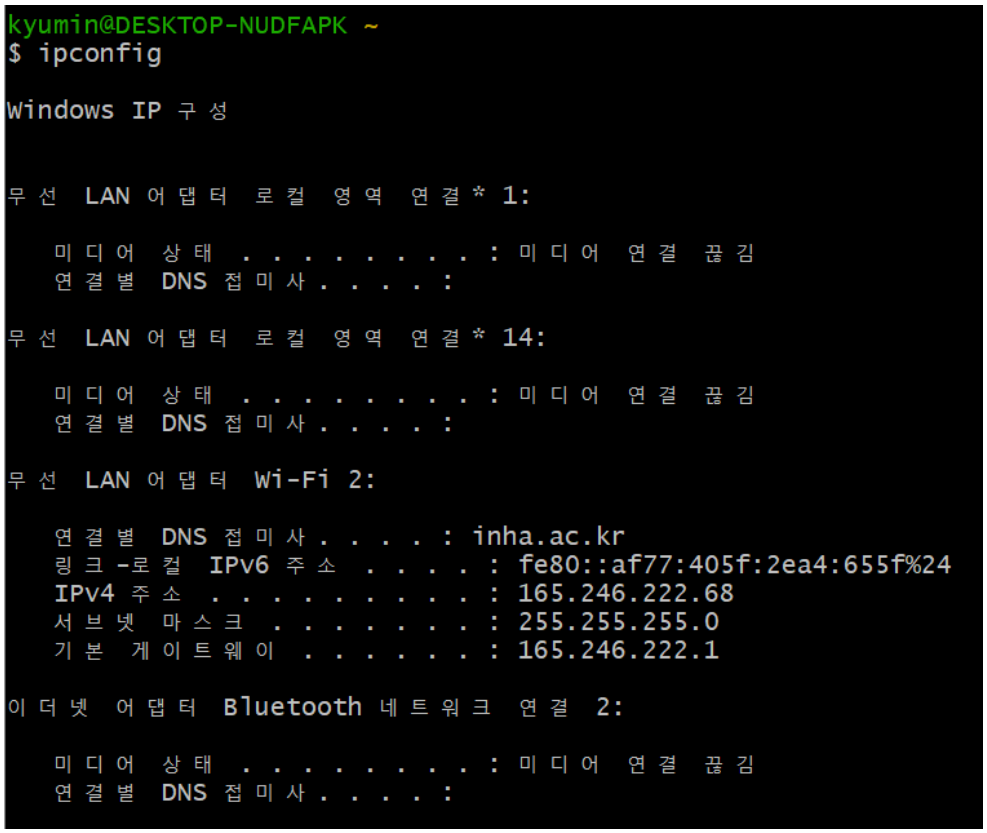
[serv.c]

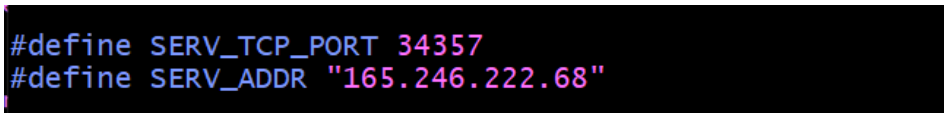
```c
#include <stdio.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <arpa/inet.h>
#include <string.h>
#include <stdlib.h>
#include <sys/time.h>
#include <unistd.h>

#define SERV_TCP_PORT 34357
#define SERV_ADDR "165.246.222.68"

int main(){
    int s1,s2,x,y;
    struct sockaddr_in serv_addr, cli_addr;
    char buf[50];
    socklen_t xx;

    printf("Hi, I am the server\n");
    bzero((char *)&serv_addr, sizeof(serv_addr));
    serv_addr.sin_family=PF_INET;
    serv_addr.sin_addr.s_addr=inet_addr(SERV_ADDR);
    serv_addr.sin_port=htons(SERV_TCP_PORT);
```

```c
    if((s1=socket(PF_INET, SOCK_STREAM, 0))<0){
        printf("socket creation error\n");
        exit(1);
    }
    printf("socket created successfully. socket num is %d\n", s1);

    x=bind(s1, (struct sockaddr *)&serv_addr, sizeof(serv_addr));
    if (x < 0){
        printf("binding failed\n");
        exit(1);
    }
    printf("binding passed\n");

    listen(s1, 5);
    xx = sizeof(cli_addr);
    s2 = accept(s1,(struct sockaddr *)&cli_addr,&xx);
    printf("we passed accept. new socket num is %d\n", s2);

    y=read(s2,buf,50);
    buf[y]=0;
    printf("we got %s from cli\n",buf);
    printf("what do you want to send to cli? enter your string\n");
    scanf("%s", buf);
    write(s2,buf,strlen(buf));
    close(s2);      // disconnect the connection
    close(s1);      // close the original socket
}
```

1) Adjust port and IP address for both cli.c and serv.c. SERV_ADDR in cli.c and serv.c should be the IP address of the linux server you are using. Use "ifconfig"(in ubuntu) or "ipconfig"(in cygwin) to find out the IP address of your virtual machine or PC. cli.c will use this SERV_ADDR to access the server while serv.c will use this SERV_ADDR to set its own IP address. Pick a port number in the range of [10000..65535]. You need two terminals: one for the server and the other for the client. Compile both and run the server first and then the client.

(* Sometimes, you have binding failure when running the server. It happens because the server port number is blocked temporarily. Wait for 10 seconds and retry or use a different port number.)





cygwing에서 ipconfig 명령어를 통해 나의 ip를 확인한 후 cli.c와 serv.c의 ip를 수정해줬다. 나의 pc가 보내고, 나의 pc가 받아야하므로 두 ip를 동일하게 입력해줬다. port번호의 경우 10000 이하는 시스템의 포트이므로 10000~60000 중 임의로 숫자를 입력했다.

```
kyumin@DESKTOP-NUDFAPK ~
$ cli
Hi, I am the client
 socket opened successfully. socket num is 3
now i am connected to the erver. enter a string to send
hello
now let's read from the server
what echoed from the server is hi
```

```
kyumin@DESKTOP-NUDFAPK ~
$ serv
Hi, I am the server
socket created successfully. socket num is 3
binding passed
we passed accept. new socket num is 4
we got hello from cli
what do you want to send to cli? enter your string
hi
```

서버가 수신을 하고 있을 때 클라이언트가 정보를 보내야하므로 serv를 먼저 실행한 후 cli를 실행했다. cli에서 hello라는 메시지를 입력했고, serv 창을 보니 hello 메시지가 정상적으로 수신되었다. 그리고 hi라는 메시지를 입력하니 cli에서도 정상적으로 입력을 받았다.

1-1) Modify cli.c and serv.c such that they can talk in sentence (not just in word as in the current implementation).

[cli.c]

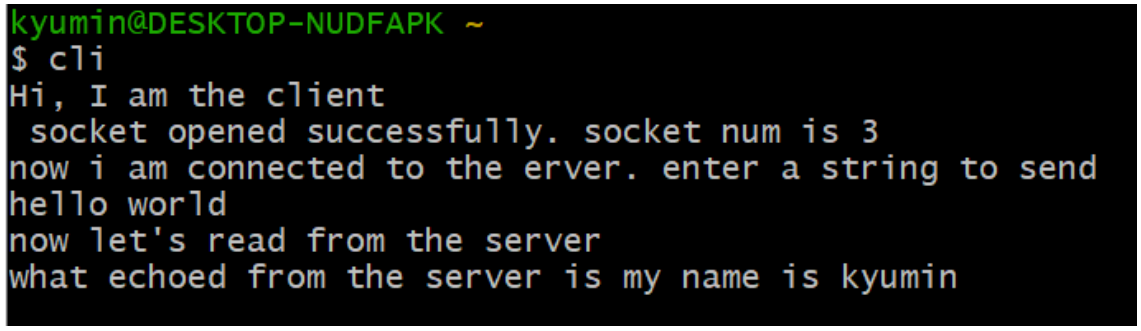
```
    /* send input str to the server */
    printf("now i am connected to the erver. enter a string to send\n");
    fgets(buf, 50, stdin);
    buf[strlen(buf) - 1] = '\0';
    write(x,buf,strlen(buf));

    printf("now let's read from the server\n");
    y=read(x,buf,50);
    buf[y]=0;
    printf("what echoed from the server is %s\n",buf);
    close(x);    // disconnect
}
```

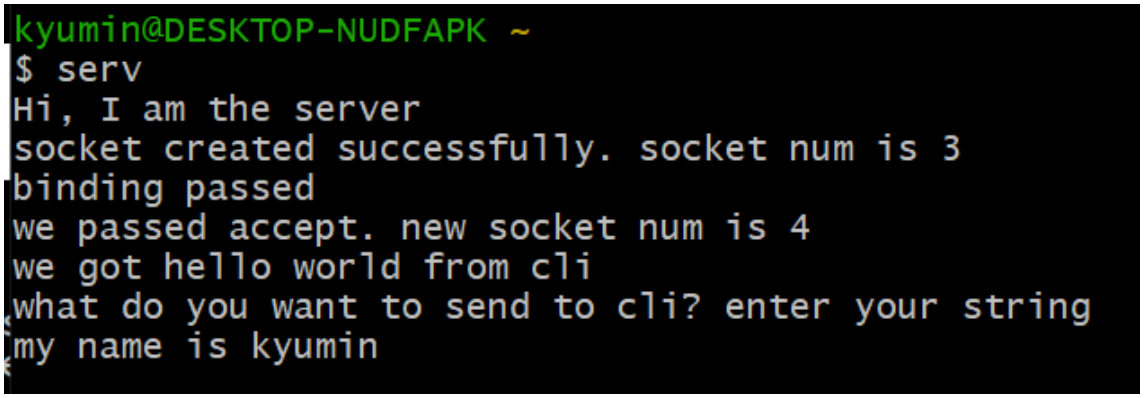
[serv.c]

```
    //socket을 통해 들어오는 연결을 듣는다.
    listen(s1, 5);
    xx = sizeof(cli_addr);
    s2 = accept(s1,(struct sockaddr *)&cli_addr,&xx);
    printf("we passed accept. new socket num is %d\n", s2);

    y=read(s2,buf,50);
    buf[y]=0;
    printf("we got %s from cli\n",buf);
    printf("what do you want to send to cli? enter your string\n");
    fgets(buf, 50, stdin);
    buf[strlen(buf) - 1] = '\0';
    write(s2,buf,strlen(buf));
    close(s2);       // disconnect the connection
    close(s1);       // close the original socket
}

"serv.c" 57L, 1542B                                          52,2-5
```

위 두 코드에서 사용자의 입력을 받는 함수인 scanf를 제거하고, fgets를 이용하여 입력을 받았다. Scanf는 문장을 입력해도 띄어쓰기까지만(단어만)입력을 받는다. 그러나 fgets의 경우 사용자가 엔터를 눌러야 입력이 끝난다. 마지막에 개행문자가 buf에 들어가므로 '₩0'으로 수정을해줬다.

[결과]





Serv를 실행한 후 cli를 실행했다. 그리고 cli에서 hello world를 입력하니 serv에서 정상적으로 수신했다는 메시지가 나타난다. 그리고 my name is kyumin을 입력하니 cli에서 정상적으로 수신했다는 메시지가 나타난다.

2) Modify cli.c and serv.c such that they can keep talking until the client sends "bye". Use a finite loop.

[cli.c]

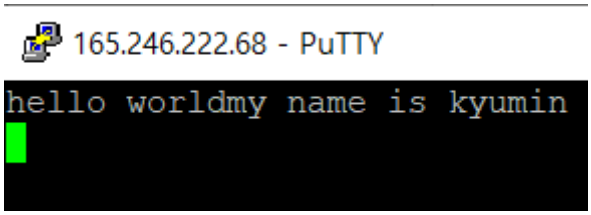
```
    for(int i = 0; i < 10; i++)
    {
        /* send input str to the server */
        printf("now i am connected to the erver. enter a string to send\n");
        fgets(buf, 50, stdin);
        buf[strlen(buf) - 1] = '\0';
        if(strcmp(buf, "bye") == 0) break;
        write(x,buf,strlen(buf));

        //socket의 메시지 출력
        printf("now let's read from the server\n");
        y=read(x,buf,50);
        buf[y]=0;
        printf("what echoed from the server is %s\n",buf);
    }
    close(x);    // disconnect
}
```

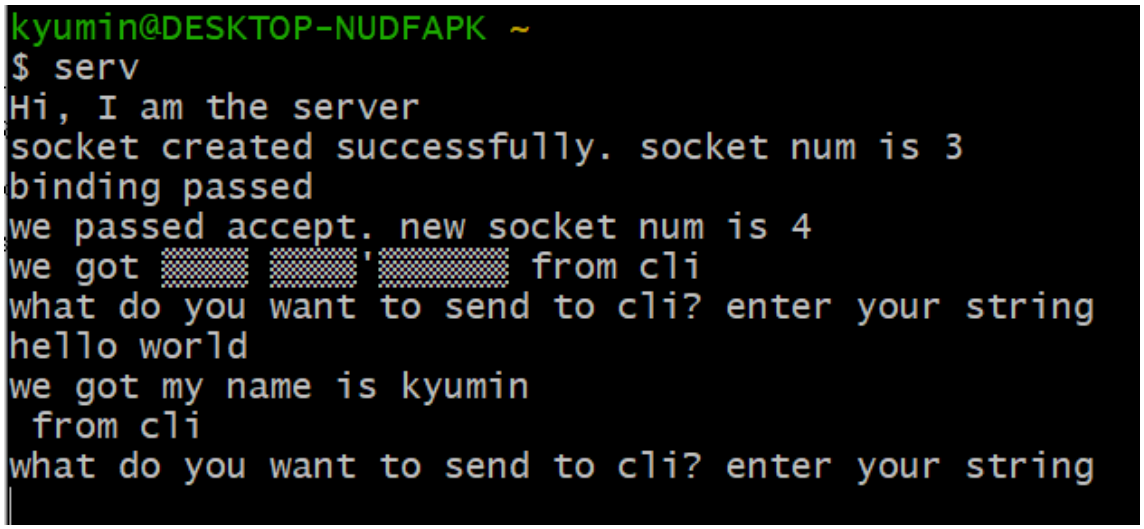
[serv 결과]

```
kyumin@DESKTOP-NUDFAPK ~
$ serv
Hi, I am the server
socket created successfully. socket num is 3
binding passed
we passed accept. new socket num is 4
we got hello from cli
what do you want to send to cli? enter your string
hi
we got  from cli
what do you want to send to cli? enter your string
```

if(strcmp(buf, "bye") == 0) break;코드를 추가하였다. Bye가 입력되면 cli는 for문을 탈출하고, x를 close 한다. 이 때 serv를 보면 화면에 공백이 출력된다.

2-1) Try to talk with the other student. Note that one of you would be the client and the other the server. SERV_ADDR and SERV_TCP_PORT in cli.c should match to those in serv.c of the other student. (If you prefer to work alone, run the client in ubuntu and run the server in cygwin terminal. Make sure the client specify the IP and port number set in the cygwin server.)





Putty를 실행하고 ip와 port를 serv.c와 동일하게 입력해줬다. 그리고 putty에서 Connection type은 Telnet으로 설정 후 실행했다. 그리고 위 사진과 같이 서로 입력한 내용이 출력 되는 것을 볼 수 있다.

2-2) Modify cli.c such that it connects to Inha web server and read the web page. Inha web server domain name is www.inha.ac.kr and port number is 80. You can find the IP address for www.inha.ac.kr with ping command. To receive the web page from a web server, use GET command (your code should send below string automatically using write() function – do not type it by hand):

GET / HTTP/1.1₩rW₩nHOST:www.inha.ac.krW₩rW₩nW₩rW₩n

where ₩ is a backslash character.

```
    for(int i = 0; i < 10; i++)
    {
        /* send input str to the server */
        /*
        printf("now i am connected to the server. enter a string to send\n");
        fgets(buf, 50, stdin);
        buf[strlen(buf) - 1] = '\0';
        if(strcmp(buf, "bye") == 0) break;
        write(x,buf,strlen(buf));
        write(x,"GET / HTTP/1.1\r\nHOST:www.inha.ac.kr\r\n\r\n",45);

        //socket의 메시지 출력
        printf("\nnow let's read from the server\n");
        y=read(x,buf,50);
        buf[y]=0;
        printf("what echoed from the server is %s\n",buf);
        */


        write(x,"GET / HTTP/1.1\r\nHOST:www.inha.ac.kr\r\n\r\n",45);
        y=read(x,buf,50);
        buf[y]=0;
        printf("%s\n", buf);
    }
    close(x);    // disconnect
```

```
kyumin@DESKTOP-NUDFAPK ~
$ cli
Hi, I am the client
 socket opened successfully. socket num is 3
HTTP/1.1 200 OK
Date: Mon, 13 May 2024 01:46:06 G
MT
Content-Type: text/html;charset=utf-8
Content
-Length: 1875
Connection: keep-alive
Server: Apa
che
Set-Cookie: JSESSIONID=E80184FDCA4EB39FE3ADF0
A9D46CE757; Path=/; HttpOnly
Cache-Control: no-ca
che
Pragma: no-cache
Expires: Thu, 01 Jan 1970 0
0:00:00 GMT
```

GET 명령어를 write 함수를 통해서 보내고, 받은 내용을 출력하는 과정을 10번 반복하였다. 그 결과는 위와 같다.

3) Modify the code such that the client and the server can talk in any order. Use a finite loop to avoid infinite number of processes.

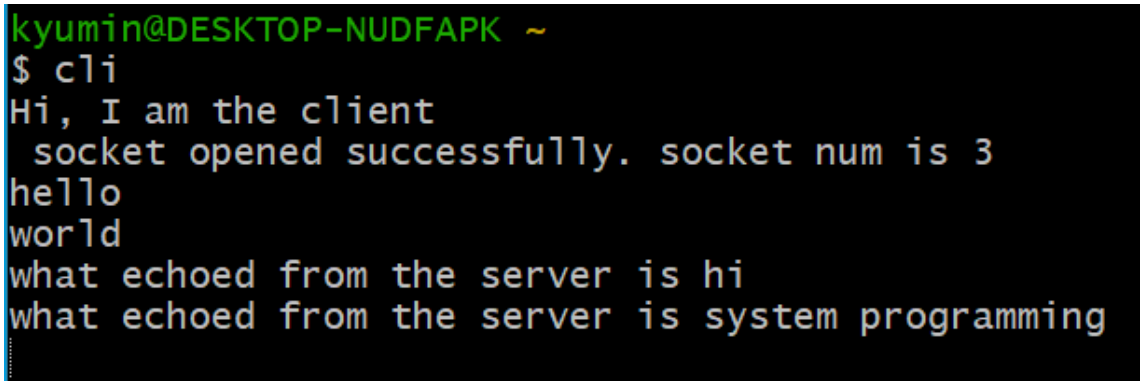
[cli.c]

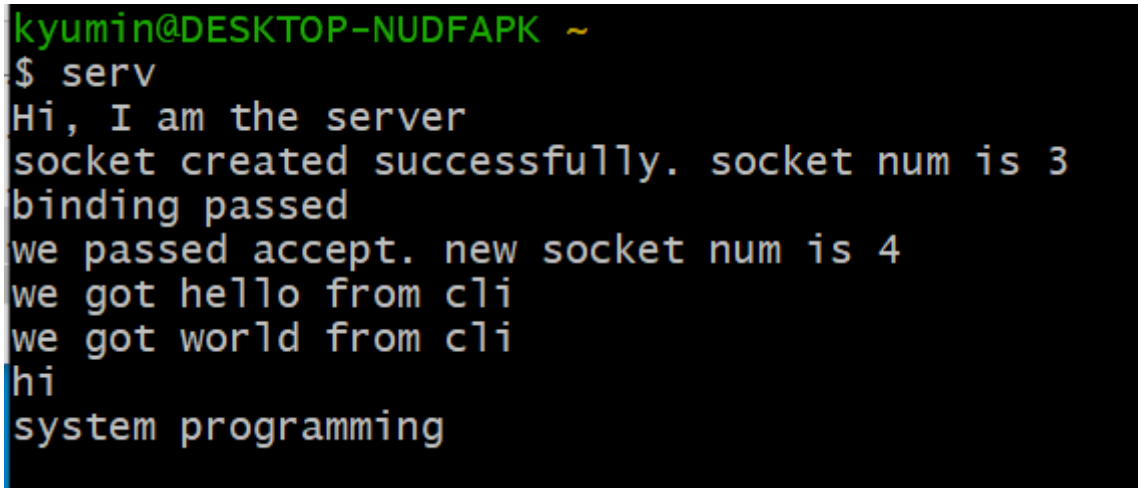
```
    //프로세스 복제
    int fp = fork();

    for(int i = 0; i < 10; i++)
    {
        if(fp != 0)
        {
            /* send input str to the server */
            //printf("now i am connected to the server. enter a string to send\n
");
            fgets(buf, 50, stdin);
            buf[strlen(buf) - 1] = '\0';
            if(strcmp(buf, "bye") == 0) break;
            write(x, buf, strlen(buf));
        }

        else
        {
            //socket의 메시지 출력
            //printf("now let's read from the server\n");
            y = read(x,buf,50);
            buf[y] = 0;
            printf("what echoed from the server is %s\n", buf);
        }
    }
    close(x);    // disconnect
                                                          44,3-9        94%
```

[serv.c]

```
    //프로세스 복제
    int fp = fork();

    //socket의 메시지를 읽고 출력
    for(int i = 0; i < 10; i++)
    {
        if(fp == 0)
        {
            y=read(s2,buf,50);
            buf[y]=0;
            printf("we got %s from cli\n",buf);
        }

        else
        {
            //printf("what do you want to send to cli? enter your string\n");
            fgets(buf, 50, stdin);
            buf[strlen(buf) - 1] = '\0';
            write(s2,buf,strlen(buf));
        }
    }
    close(s2);      // disconnect the connection
    close(s1);      // close the original socket
                                                          47,20-18      95
```

[결과]





3-1) Write a server that speaks only with registered user in "login.txt" file. The server should open login.txt file before chatting and remember the names of legal users.

cli -> serv: hello

serv -> cli: name?

cli -> serv: kim

serv -> cli: ok. now we can talk

.... cli and serv chats ....

cli -> serv: hello

serv -> cli: name?

cli -> serv: hong

serv -> cli: not legal user. disconnecting.

4) Implement simple ftp server and client. If the client doesn't follow the protocol, the server should stop the communication. This is not chatting program. Do not type "hello", "what file do you want?", etc. The server and client should automatically send or receive the ftp protocol messages. The user will only provide the file name to download. Do not use the code from Problem 3, which will make the coding very hard. Modify the code from Problem 1. Make sure your ftp server and client can handle any file size.

Simple ftp protocol

client => server: hello
server => client: what file do you want?
client => server: file name
server => client: file contents

4-1) Modify your ftp client so that the client saves the file content under the same file name. For this purpose, run the client in a different directory if you run the server and client in the same machine.

5) Modify your ftp server such that it can handle multiple clients at the same time.

6) Write a client in your PC as follows and let it talk to the server program in the lab server. To compile the client program:

- Make an empty c++ project and copy the code given below.
- Adjust the server IP and port number
- Select
  "project->project property->manifest tools->input and output->include manifest" and set "No"
- add ws2_32.lib in project>project property>link>input>additional dependencies
- Select build->Solution Build
- You should see "Success 1" at the bottom of the compile window.
- Sometimes, the platform of the program (the one shown in the main interface screen:release or debug) is different from the platform in properties configuration. They should match.)
- If you have an error related with "pre-compiled header", go to
  "project->project property->c/c++>pre-compiled header > pre-compiled header>" and set "No"

(* If you use MAC, download cli.c from the lab server and use it as your local client program. You may have to include "unistd.h" to avoid compile errors.)

```
#define _CRT_SECURE_NO_WARNINGS
#include "winsock2.h"
#include "ws2tcpip.h"
#include "stdio.h"

#define SERVER_PORT 9924   // server port number
#define BUF_SIZE 4096 // block transfer size
#define QUEUE_SIZE 10
#define IPAddress "165.246.38.152" // server IP address

int main()
{
        WORD                    wVersionRequested;
        WSADATA                 wsaData;
        SOCKADDR_IN             target; //Socket address information
        SOCKET                  s;
        int                     err;
        int                     bytesSent;
        char                    buf[100];

            //--- INITIALIZATION -----------------------------------
            wVersionRequested = MAKEWORD( 1, 1 );
            err = WSAStartup( wVersionRequested, &wsaData );

            if ( err != 0 ) {
                printf("WSAStartup error %ld", WSAGetLastError() );
                WSACleanup();
                return false;
            }
            //-----------------------------------------------------

            //---- Build address structure to bind to socket.--------
            target.sin_family = AF_INET; // address family Internet
            target.sin_port = htons (SERVER_PORT); //Port to connect on
            inet_pton(AF_INET, IPAddress, &(target.sin_addr.s_addr)); // target IP
            //-----------------------------------------------------------


            // ---- create SOCKET-------------------------------------
            s = socket (AF_INET, SOCK_STREAM, IPPROTO_TCP); //Create socket
            if (s == INVALID_SOCKET)
```

```
        {
                printf("socket error %ld" , WSAGetLastError() );
                WSACleanup();
                return false; //Couldn't create the socket
        }
        //---------------------------------------------------------


        //---- try CONNECT ------------------------------------------
        if (connect(s, (SOCKADDR *)&target, sizeof(target)) == SOCKET_ERROR)
        {
                printf("connect error %ld", WSAGetLastError() );
                WSACleanup();
                return false; //Couldn't connect
        }
        //---------------------------------------------------------

        //---- SEND bytes -------------------------------------------
        printf("enter a string to send to server₩n");
        gets_s(buf, 99);
        bytesSent = send( s, buf, strlen(buf), 0 ); // use "send" in windows
        printf( "Bytes Sent: %ld ₩n", bytesSent );


        // now receive
        int n;
        n=recv(s, buf, 50, 0); // read max 50 bytes
        buf[n]=0; // make a string
        printf("received: %s₩n", buf);

        //---------------------------------------------------------
        closesocket( s );
        WSACleanup();

        return 0;
}
```

7) Write an ftp client in your PC and let it talk to the ftp server you made in problem 5). Use this client to download a file from the lab server and save under the same file name.