

REPORT



과 목 :	네트워크프로그래밍
제출일자 :	2021. 11. 03
담당교수 :	황성호
학 과 :	컴퓨터공학과
학 번 :	201720970
이 름 :	권대한

1. TCPServer 소스코드

```
#define _WINSOCK_DEPRECATED_NO_WARNINGS

#pragma comment(lib, "ws2_32")

#include <stdio.h>

#include <stdlib.h>

#include <WinSock2.h>

#define SERVER_PORT 9000

#define BUFFER_SIZE 512

// 소켓 함수 오류 출력 후 종료
//general error message output Func
void err_quit(char* msg)
{
    // "far" + pointer + void
    LPVOID lpMsgBuf;

    FormatMessage
    (
        //flag
        FORMAT_MESSAGE_ALLOCATE_BUFFER | FORMAT_MESSAGE_FROM_SYSTEM,

        //source
        NULL,

        //message id
```

```
WSAGetLastError(),  
  
//language id  
  
MAKELANGID(LANG_NEUTRAL, SUBLANG_DEFAULT),  
  
//Long Pointer t_string, buffer location  
  
(LPTSTR)&lpMsgBuf,  
  
//size  
  
0,  
  
//arguments  
  
NULL  
  
);
```

MessageBox

```
(  
  
//handle  
  
NULL,  
  
//text  
  
(LPCTSTR)lpMsgBuf,  
  
//caption  
  
msg,  
  
//type  
  
MB_ICONERROR  
  
);
```

```
//pointer memory free Func
```

```
LocalFree(lpMsgBuf);
```

```
        //error code

        exit(1);
    }

void err_display(char* msg)
{
    LPVOID lpMsgBuf;

    FormatMessage
    (
        //flag
        FORMAT_MESSAGE_ALLOCATE_BUFFER | FORMAT_MESSAGE_FROM_SYSTEM,

        //source
        NULL,

        //message id
        WSAGetLastError(),

        //language id
        MAKELANGID(LANG_NEUTRAL, SUBLANG_DEFAULT),

        //Long Pointer t_string, buffer location
        (LPTSTR)&lpMsgBuf,

        //size
        0,

        //arguments
        NULL

    );
}
```

```
printf("[%s] %s", msg, (char*)lpMsgBuf);
```

```
LocalFree(lpMsgBuf);
```

```
}
```

```
int main(int argc, char* argv[])
```

```
{
```

```
    int retval;
```

```
    //winsock init
```

```
    WSADATA wsa;
```

```
    if (WSAStartup(MAKEWORD(2, 2), &wsa) != 0)
```

```
    {
```

```
        return 1;
```

```
    }
```

```
    //socket()
```

```
    SOCKET listen_sock = socket(AF_INET, SOCK_STREAM, 0);
```

```
    if (listen_sock == INVALID_SOCKET)
```

```
    {
```

```
        err_quit("socket()");
```

```
    }
```

```
    //bind()
```

```
    SOCKADDR_IN serveraddr;
```

```

ZeroMemory(&serveraddr, sizeof(serveraddr));

serveraddr.sin_family = AF_INET;

serveraddr.sin_addr.s_addr = htonl(INADDR_ANY);

//serveraddr.sin_addr.s_addr = inet_addr(INADDR_ANY);

serveraddr.sin_port = htons(SERVER_PORT);

retval = bind(listen_sock, (SOCKADDR*)&serveraddr, sizeof(serveraddr));

if (retval == SOCKET_ERROR)
{
    err_quit("bind()");
}

//listen()

retval = listen(listen_sock, SOMAXCONN);

if (retval == SOCKET_ERROR)
{
    err_quit("listen()");
}

//데이터 통신 변수

SOCKET client_sock;

SOCKADDR_IN clientaddr;

int addrlen;

char buf[BUFFER_SIZE + 1];

while (1)

```

```

{

    //accept()

    addrlen = sizeof(clientaddr);

    client_sock = accept(listen_sock, (SOCKADDR*)&clientaddr, &addrlen);

    if (client_sock == INVALID_SOCKET)

    {

        err_display("accept()");

        break;

    }

    printf("\n[TCP 서버] 클라이언트 접속: IP 주소 = %s, 포트 번호 = %d\n",

        inet_ntoa(clientaddr.sin_addr), ntohs(clientaddr.sin_port));

    while (1)

    {

        retval = recv(client_sock, buf, BUFFER_SIZE, 0);

        if (retval == SOCKET_ERROR)

        {

            err_display("recv()");

            break;

        }

        else if (retval == 0)

        {

            break;

        }

        //받은 데이터 출력
    }

```

```

        buf[retval] = 0;

        printf("[TCP/%s : %d] %s\\n",

               inet_ntoa(clientaddr.sin_addr), ntohs(clientaddr.sin_port), buf);

        //데이터 보내기

        retval = send(client_sock, buf, retval, 0);

        if (retval == SOCKET_ERROR)
        {

            err_display("send()");

            break;

        }

    }

    closesocket(client_sock);

    printf("[TCP 서버] 클라이언트 종료: IP 주소 = %s, 포트 번호 = %d\\n",

           inet_ntoa(clientaddr.sin_addr), ntohs(clientaddr.sin_port));

}

//closesocket()

closesocket(listen_sock);

//winsock 종료

WSACleanup();

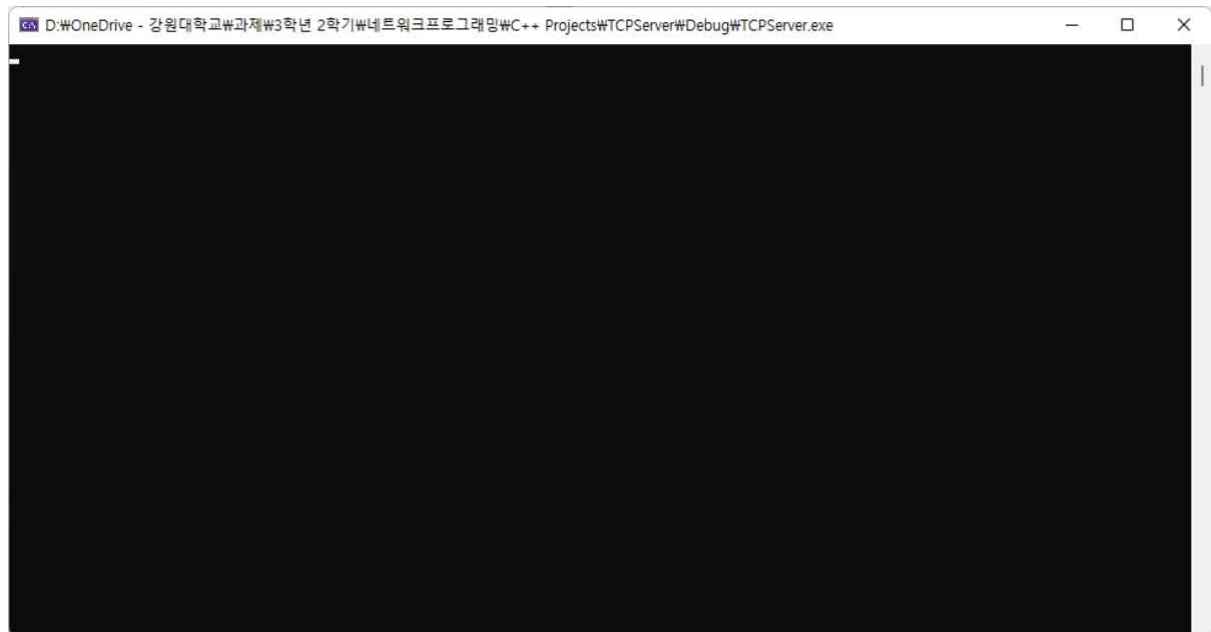
return 0;

}

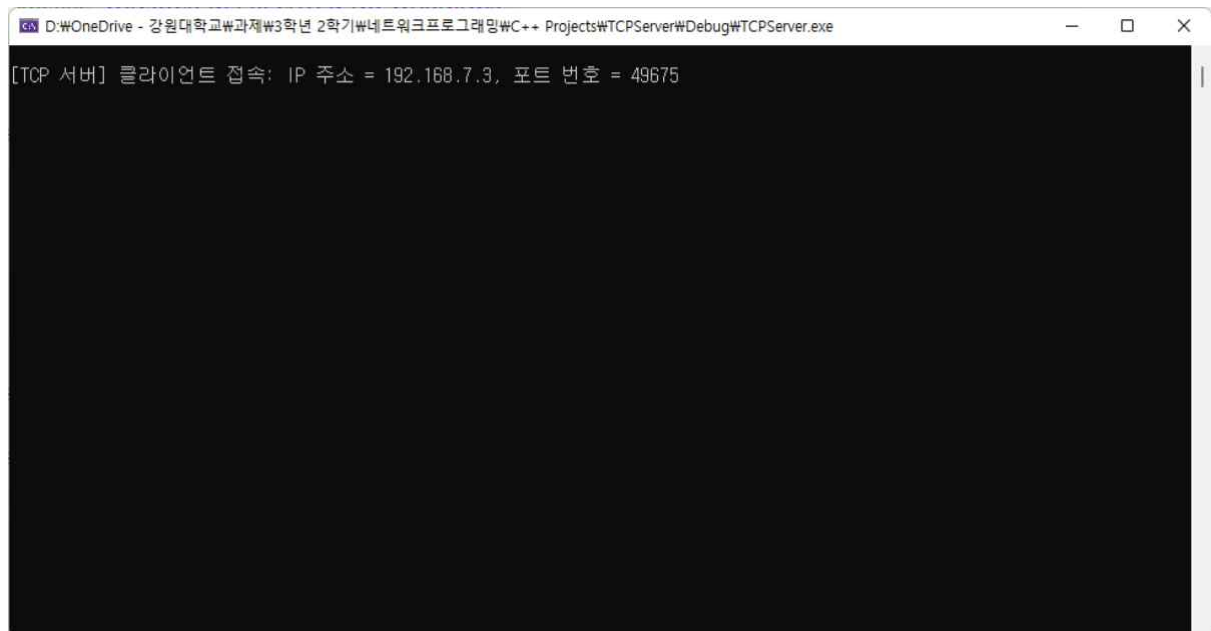
```


TCPServer 출력결과

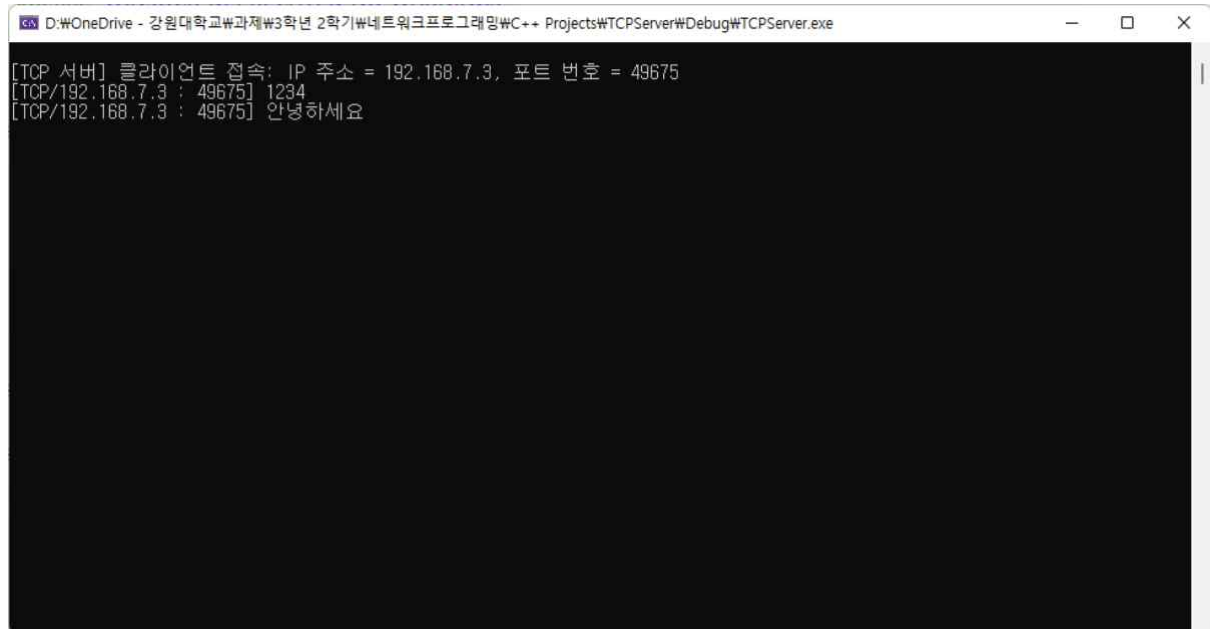
Case 1. TCPServer 단독 실행했을 경우



Case 2. 실습 컴퓨터에서 TCPClient를 통해 접속하였을 경우



Case 3. TCPClient로부터 데이터를 수신했을 경우



```
D:\OneDrive - 강원대학교\과제\3학년 2학기\네트워크프로그래밍\C++ Projects\TCPServer\Debug\TCPServer.exe
[TCP 서버] 클라이언트 접속: IP 주소 = 192.168.7.3, 포트 번호 = 49675
[TCP/192.168.7.3 : 49675] 1234
[TCP/192.168.7.3 : 49675] 안녕하세요
```

The image shows a Windows command prompt window titled "D:\OneDrive - 강원대학교\과제\3학년 2학기\네트워크프로그래밍\C++ Projects\TCPServer\Debug\TCPServer.exe". The window contains three lines of text: "[TCP 서버] 클라이언트 접속: IP 주소 = 192.168.7.3, 포트 번호 = 49675", "[TCP/192.168.7.3 : 49675] 1234", and "[TCP/192.168.7.3 : 49675] 안녕하세요".

2. TCPClient 소스코드

```
#define _WINSOCK_DEPRECATED_NO_WARNINGS

#pragma comment(lib, "ws2_32")

#include <stdio.h>

#include <stdlib.h>

#include <WinSock2.h>

#define SERVERIP "192.168.7.3"

#define SERVER_PORT 9000

#define BUFFER_SIZE 512

// 소켓 함수 오류 출력 후 종료
//general error message output Func
void err_quit(char* msg)
{
    // "far" + pointer + void
    LPVOID lpMsgBuf;

    FormatMessage
    (
        //flag
        FORMAT_MESSAGE_ALLOCATE_BUFFER | FORMAT_MESSAGE_FROM_SYSTEM,

        //source
        NULL,
```

```
    //message id
    WSAGetLastError(),

    //language id
    MAKELANGID(LANG_NEUTRAL, SUBLANG_DEFAULT),

    //Long Pointer t_string, buffer location
    (LPTSTR)&lpMsgBuf,

    //size
    0,

    //arguments
    NULL
);
```

MessageBox

```
(
    //handle
    NULL,

    //text
    (LPCTSTR)lpMsgBuf,

    //caption
    msg,

    //type
    MB_ICONERROR
);
```

```
//pointer memory free Func
```

```
LocalFree(lpMsgBuf);
```

```

        //error code

        exit(1);
    }

//소켓 함수 오류 출력
void err_display(char* msg)
{
    LPVOID lpMsgBuf;

    FormatMessage
    (
        //flag
        FORMAT_MESSAGE_ALLOCATE_BUFFER | FORMAT_MESSAGE_FROM_SYSTEM,

        //source
        NULL,

        //message id
        WSAGetLastError(),

        //language id
        MAKELANGID(LANG_NEUTRAL, SUBLANG_DEFAULT),

        //Long Pointer t_string, buffer location
        (LPTSTR)&lpMsgBuf,

        //size
        0,

        //arguments
        NULL
    );
}

```

```
);
```

```
printf("[%s] %s", msg, (char*)lpMsgBuf);
```

```
LocalFree(lpMsgBuf);
```

```
}
```

//사용자 정의 데이터 수신 함수

```
int recvn(SOCKET s, char* buf, int len, int flags)
```

```
{
```

```
    int received;
```

```
    char* ptr = buf;
```

```
    int left = len;
```

```
    while (left > 0)
```

```
    {
```

```
        received = recv(
```

```
            //socket struct, object
```

```
            s,
```

```
            //buffer
```

```
            ptr,
```

```
            //left length
```

```
            left,
```

```
            //flags, default == 0
```

```
flags);
```

```
if (received == SOCKET_ERROR)
```

```
{
```

```
    return SOCKET_ERROR;
```

```
}
```

```
//Transfer end
```

```
else if (received == 0)
```

```
{
```

```
    break;
```

```
}
```

```
left -= received;
```

```
ptr += received;
```

```
}
```

```
return (len - left);
```

```
}
```

```
int main(int argc, char *argv[])
```

```
{
```

```
    int retval;
```

```
    //winsock init
```

```
    WSADATA wsa;
```

```
    if (WSAStartup(MAKEWORD(2, 2), &wsa) != 0)
```

```
{
```

```
    return 1;
```

```
}
```

```
//socket()
```

```
SOCKET sock = socket(AF_INET, SOCK_STREAM, 0);
```

```
if (sock == INVALID_SOCKET)
```

```
{
```

```
    err_quit("socket()");
```

```
}
```

```
//connect()
```

```
SOCKADDR_IN serveraddr;
```

```
ZeroMemory(&serveraddr, sizeof(serveraddr));
```

```
serveraddr.sin_family = AF_INET;
```

```
serveraddr.sin_addr.s_addr = inet_addr(SERVERIP);
```

```
serveraddr.sin_port = htons(SERVER_PORT);
```

```
retval = connect(sock, (SOCKADDR*)&serveraddr, sizeof(serveraddr));
```

```
if (retval == SOCKET_ERROR)
```

```
{
```

```
    err_quit("connect()");
```

```
}
```

```
//데이터 통신에 사용할 변수
```

```
char buf[BUFFER_SIZE + 1];
```

```
int len;
```

```
//서버와 데이터 통신
```

```
while (1)
```

```
{
```



```
//데이터 입력

printf("\n[보낼 데이터] ");

if (fgets(buf, BUFFER_SIZE + 1, stdin) == NULL)

{

    break;

}

// '\n' 문자 제거 <- 사용자가 전달하는 텍스트에 개행문자가 포함되어 있으므로,,!

len = strlen(buf);

if (buf[len - 1] == '\n')

{

    buf[len - 1] = '\0';

}

if (strlen(buf) == 0)

{

    break;

}

//데이터 보내기

retval = send(sock, buf, strlen(buf), 0);

if (retval == SOCKET_ERROR)

{

    err_display("send()");

    break;

}

printf("[TCP 클라이언트] %d바이트를 보냈습니다.\n", retval);
```

```
//데이터 받기

retval = recvn(sock, buf, retval, 0);

if (retval == SOCKET_ERROR)

{

    err_display("recv()");

    break;

}

else if (retval == 0)

{

    break;

}

//받은 데이터 출력

buf[retval] = '\0';

printf("[TCP 클라이언트] %d바이트를 받았습니다.\n", retval);

printf("[받은 데이터] %s\n", buf);

}

//closesocket()

closesocket(sock);

//winsock 종료

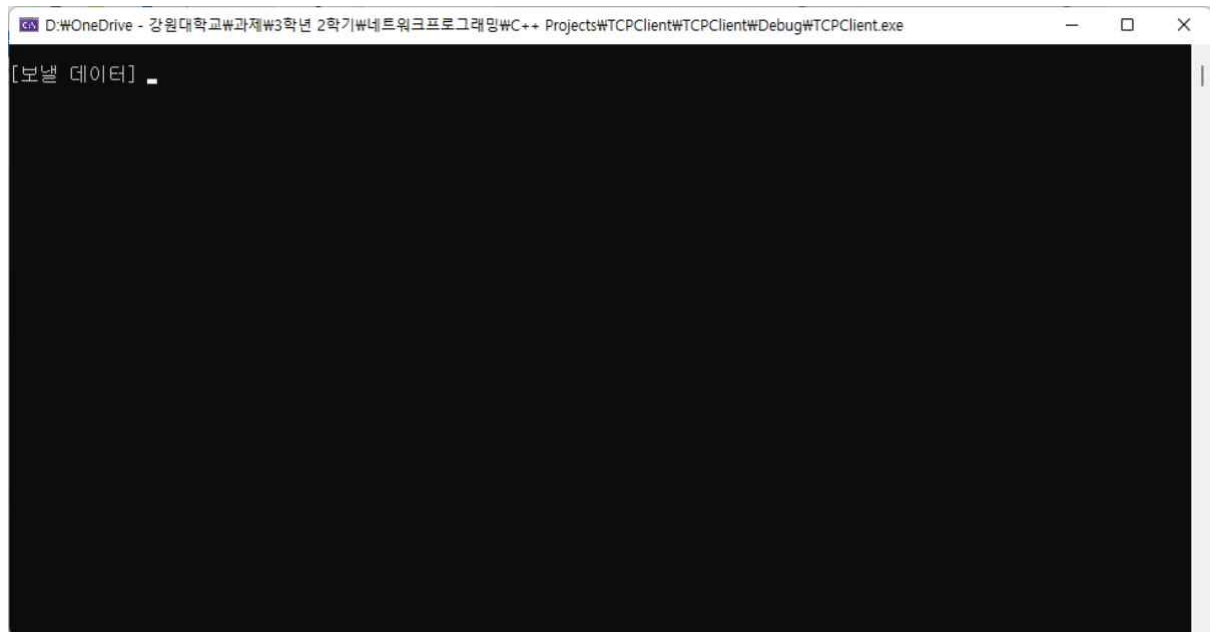
WSACleanup();

return 0;

}
```

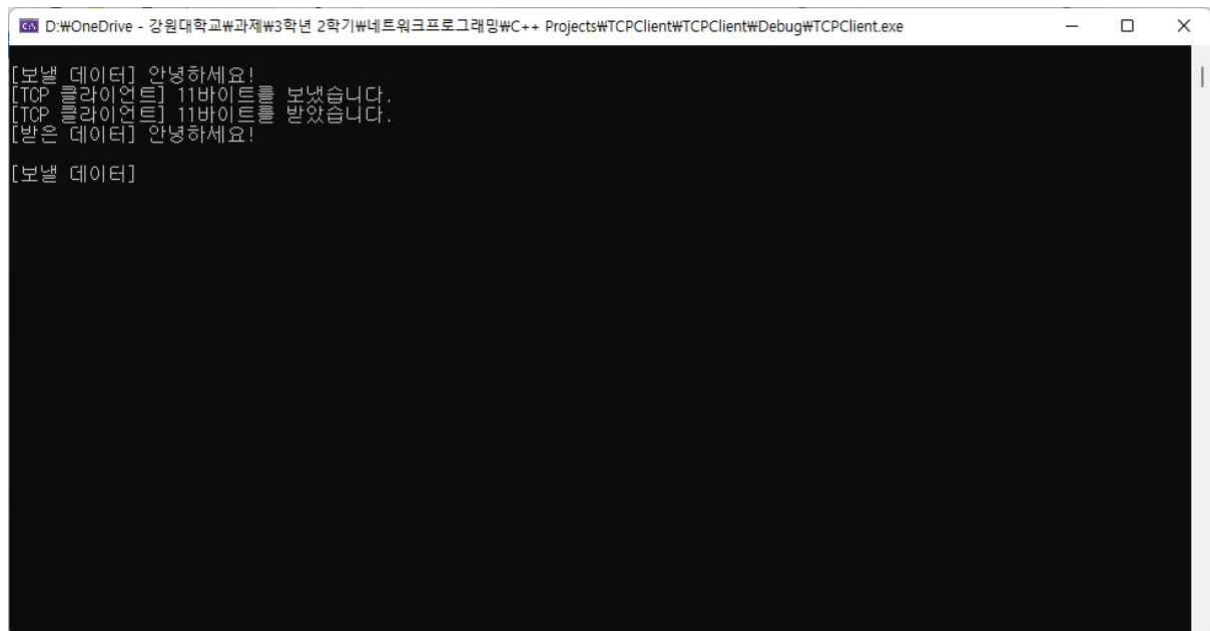
TCPClient 출력결과

Case 1. TCPServer(192.168.7.3)에 연결되었을 경우



```
D:\OneDrive - 강원대학교\과제\3학년 2학기\네트워크프로그래밍\C++ Projects\TCPClient\TCPClient\Debug\TCPClient.exe
[보낼 데이터] _
```

Case 2. TCPServer에 데이터를 보냈을 경우



```
D:\OneDrive - 강원대학교\과제\3학년 2학기\네트워크프로그래밍\C++ Projects\TCPClient\TCPClient\Debug\TCPClient.exe
[보낼 데이터] 안녕하세요!
[TCP 클라이언트] 11바이트를 보냈습니다.
[TCP 클라이언트] 11바이트를 받았습니다.
[받은 데이터] 안녕하세요!
[보낼 데이터] _
```

3. PC에서의 고정 IP address 설정

7조이므로 192.168.7.X 대역의 IP를 설정하였다.

인터넷 프로토콜 버전 4(TCP/IPv4) 속성

일반

네트워크가 IP 자동 설정 기능을 지원하면 IP 설정이 자동으로 할당되도록 할 수 있습니다. 지원하지 않으면, 네트워크 관리자에게 적절한 IP 설정값을 문의해야 합니다.

☐ 자동으로 IP 주소 받기(O)

☒ 다음 IP 주소 사용(S):

IP 주소(I): 192 . 168 . 7 . 1

서브넷 마스크(U): 255 . 255 . 255 . 0

기본 게이트웨이(D): . . .

☐ 자동으로 DNS 서버 주소 받기(B)

☒ 다음 DNS 서버 주소 사용(E):

기본 설정 DNS 서버(P): . . .

보조 DNS 서버(A): . . .

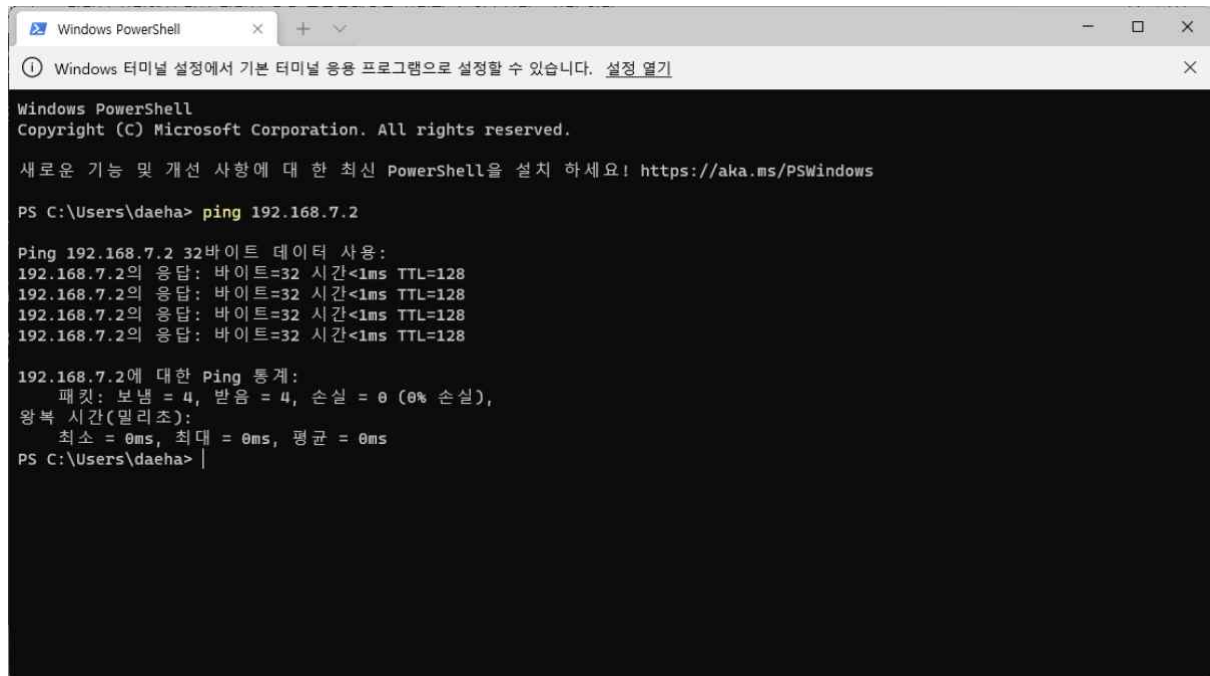
☐ 끝낼 때 설정 유효성 검사(L)

고급(V)...

확인 취소

4. Ping 테스트 결과

192.168.7.1이 본인 IP Address 이므로, 192.168.7.2, 192.168.7.3의 Ping 테스트 결과를 첨부하겠다.



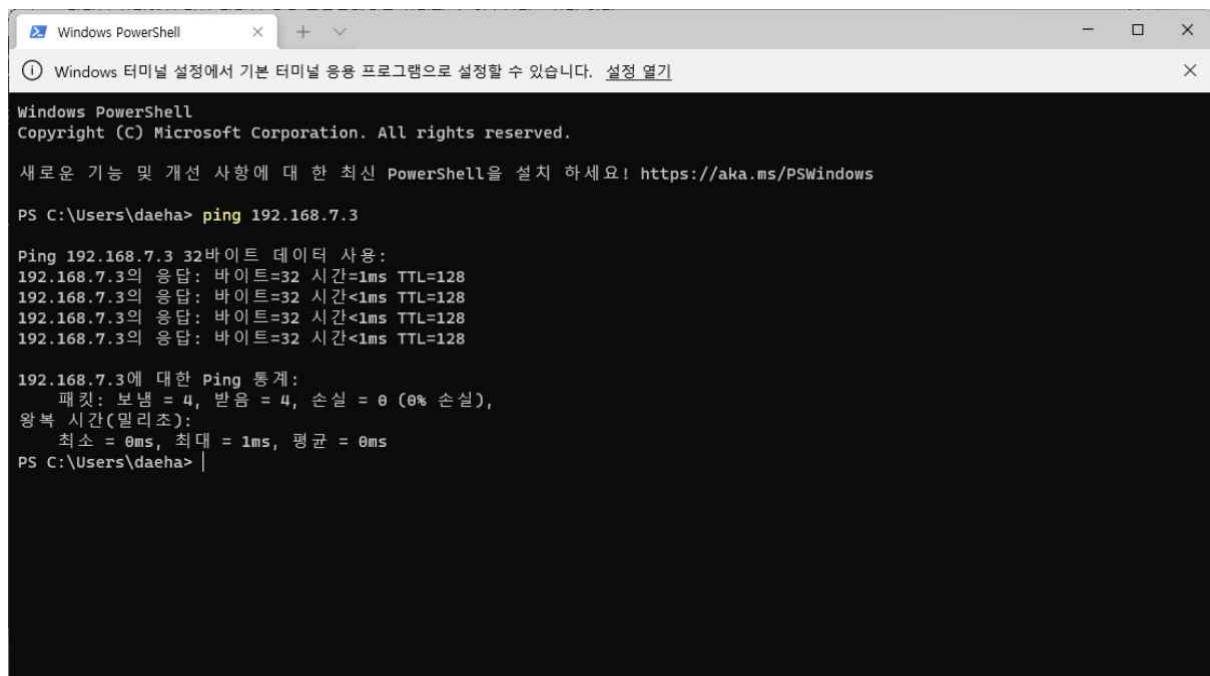
```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

새로운 기능 및 개선 사항에 대한 최신 PowerShell을 설치하세요! https://aka.ms/PSWindows

PS C:\Users\daeha> ping 192.168.7.2

Ping 192.168.7.2 32바이트 데이터 사용:
192.168.7.2의 응답: 바이트=32 시간<1ms TTL=128
192.168.7.2의 응답: 바이트=32 시간<1ms TTL=128
192.168.7.2의 응답: 바이트=32 시간<1ms TTL=128
192.168.7.2의 응답: 바이트=32 시간<1ms TTL=128

192.168.7.2에 대한 Ping 통계:
    패킷: 보냄 = 4, 받음 = 4, 손실 = 0 (0% 손실),
    왕복 시간(밀리초):
        최소 = 0ms, 최대 = 0ms, 평균 = 0ms
PS C:\Users\daeha> |
```



```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

새로운 기능 및 개선 사항에 대한 최신 PowerShell을 설치하세요! https://aka.ms/PSWindows

PS C:\Users\daeha> ping 192.168.7.3

Ping 192.168.7.3 32바이트 데이터 사용:
192.168.7.3의 응답: 바이트=32 시간=1ms TTL=128
192.168.7.3의 응답: 바이트=32 시간<1ms TTL=128
192.168.7.3의 응답: 바이트=32 시간<1ms TTL=128
192.168.7.3의 응답: 바이트=32 시간<1ms TTL=128

192.168.7.3에 대한 Ping 통계:
    패킷: 보냄 = 4, 받음 = 4, 손실 = 0 (0% 손실),
    왕복 시간(밀리초):
        최소 = 0ms, 최대 = 1ms, 평균 = 0ms
PS C:\Users\daeha> |
```