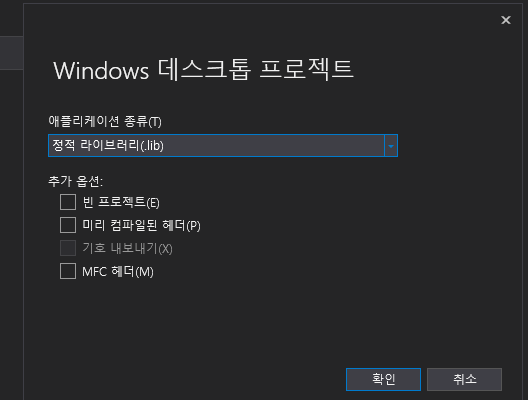
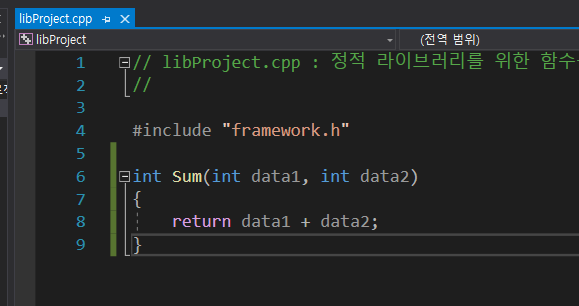
**Lib 만들기**

<https://m.blog.naver.com/PostView.naver?isHttpsRedirect=true&blogId=ruvendix&logNo=220908681017>

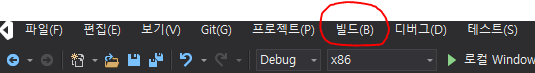
1.윈도우 데스크톱 마법사에서 애플리케이션 종류를 lib로 바꾸고 생성한다.



2.구현하고자 하는 함수 선언한다.



3.빌드 -> 솔루션빌드를 수행한다.

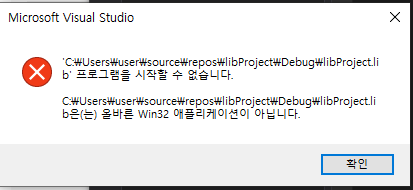


평소처럼 ctrl + f5 할 시 아래의 오류가 발생하는데

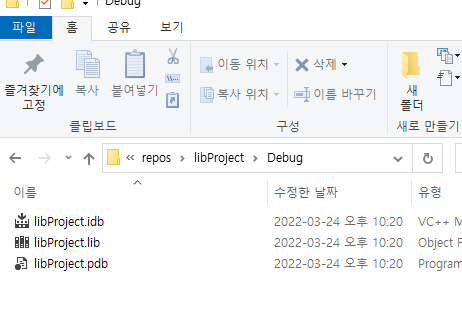
이는 정적라이브러리는 실행파일이 아니기 때문에 링커가 관여하지 않는다.

컴파일만 관여하기 때문에 소스파일을 컴파일만 한다.

따라서 그냥 빌드->솔루션 빌드 해준다.



4. 솔루션이 있는 파일로 가서 debug 폴더를 보면 lib pdb가 생성되어있다.

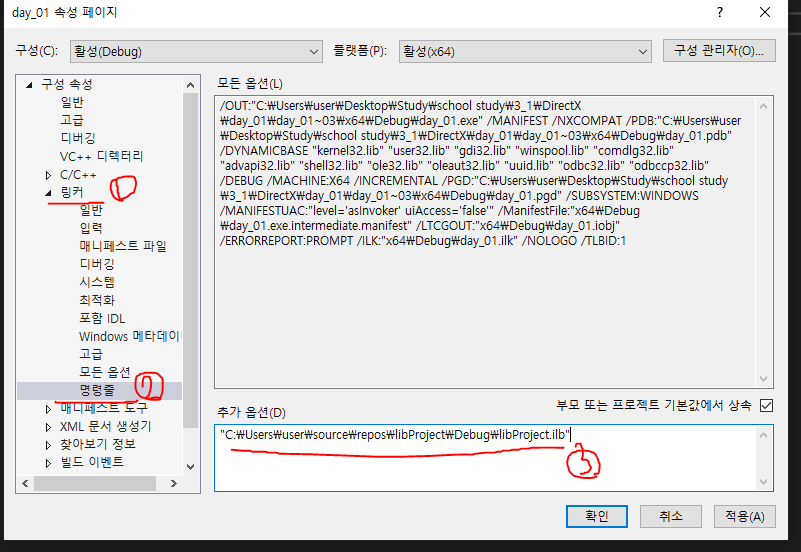


**Lib 사용하기**

방법은 2가지

1. **프로젝트 속성을 이용하기**

프로젝트 속성 -> 링커 -> 명령줄 ->추가옵션에 lib가 있는 파일 경로명을 “ “ 안에 넣고 적용, 확인을 누릅니다.

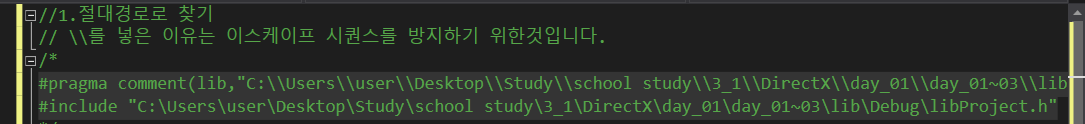


1. **소스 코드에서 사용하기**

2-1) 절대 경로로 찾기

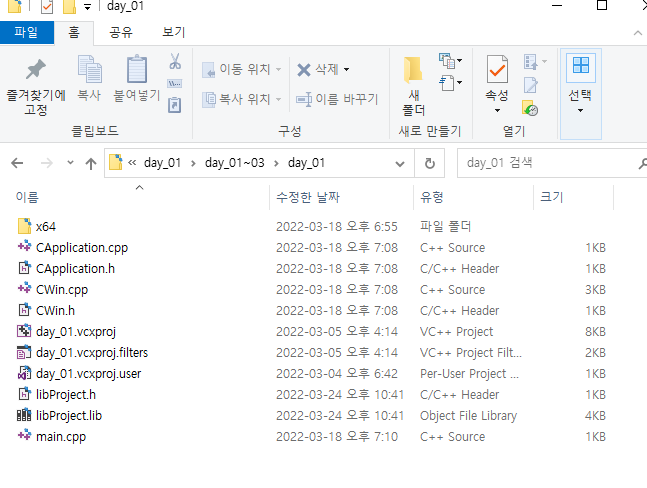
#pragma comment(lib,”lib가 위치한 경로 주소”)

#include “헤더파일이 위치한 경로주소”



2-2)상대 경로로 찾기

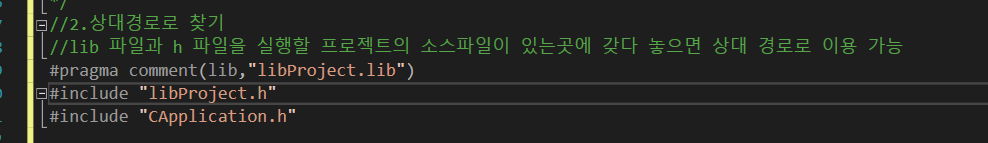
Lib의 헤더파일과 lib파일을 사용할 솔루션의 파일에 넣습니다.



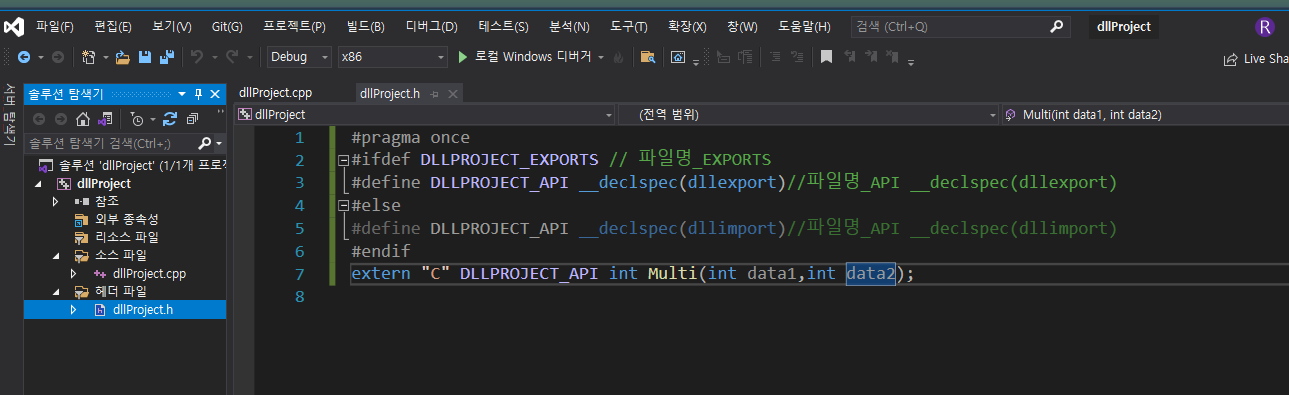
그리고 소스에 밑의 명령문을 작성합니다.

#pragma comment(lib,”라이브러리명.lib”)

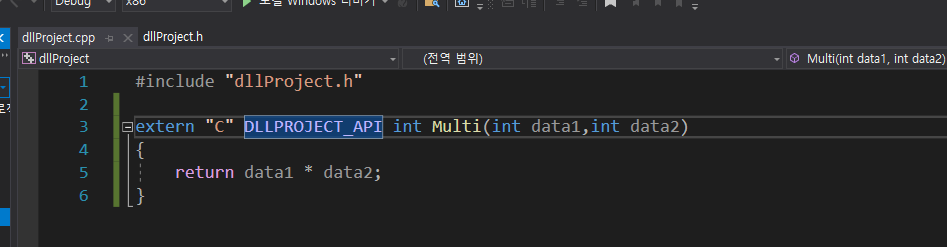
#include “라이브러리헤더파일명.h”



**DLL 생성**



2.



1. 빌드

**DLL 사용**

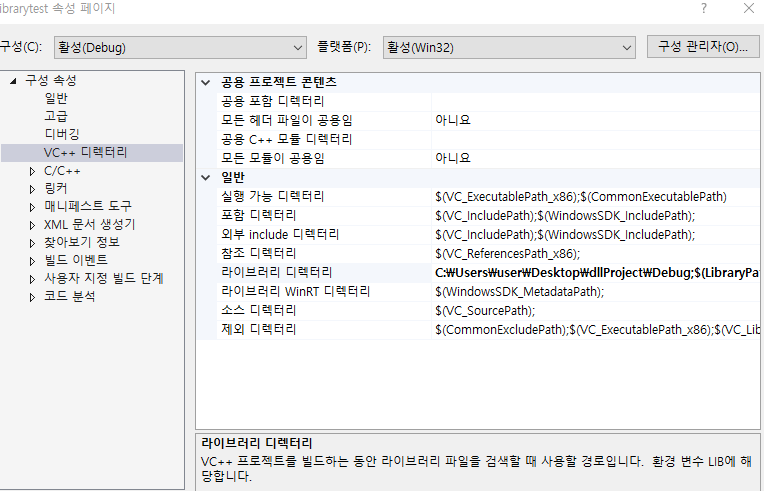
**-1 묵시적 링킹**

<https://gentlesark.tistory.com/11>

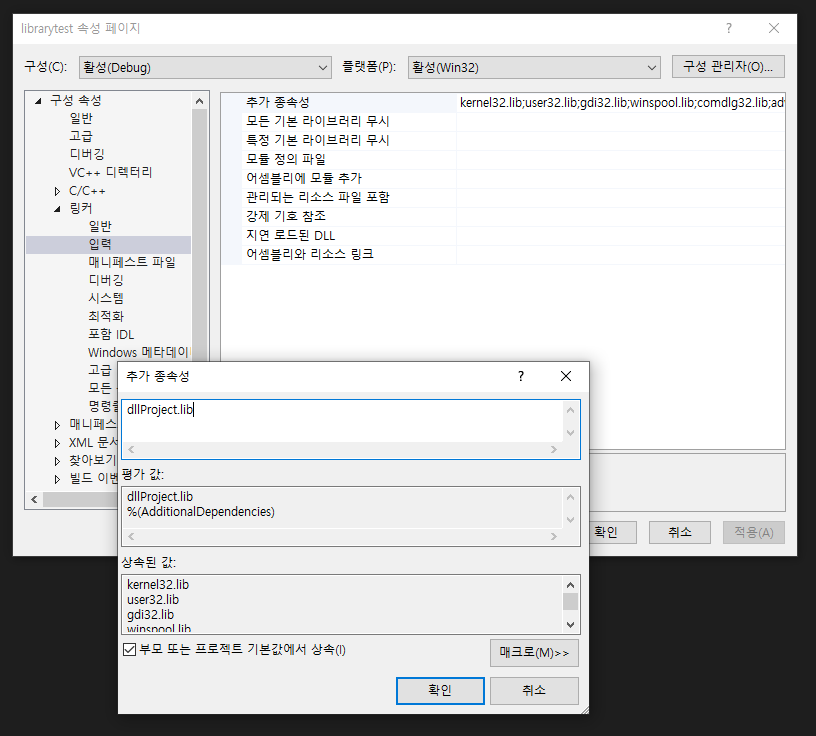
1. extern "C" \_\_declspec(dllimport) int 함수명(int, int);
2. 프로젝트 속성 -> vc++디렉터리 -> 라이브러리 디렉터리에 가서

Dll이 있는 폴더 경로주소를 붙여넣는다.

Lib를 불러들일 때 경로지정없이도 알아서 저 디렉터리를 검사한다.



1. 구성속성 – 링커-입력-추가종속성에서 lib파일을 추가한다.



1. dll 파일을 솔루션 파일이 있는곳에 둔다.
2. 실행.

Extern “c”는?

컴파일러가 소스를 컴파일 할 때 함수명은 네임맹글링 과정을 거쳐 변형됨.

컴파일러마다 규칙이 달라 이름을 통해 함수호출하는 것은 문제발생 할 수 있음.

Extern “C”는 symbol을 만들어 달라는 뜻으로 맹글링이 발생하지 않아 다른 컴파일러여도 함수호출에 문제 없이 사용할 수 있다.

\_\_declspec(dllexport)?

이 키워드가 붙은 함수는 dll 외부로 노출됨.

Dll 사용하여 위 키워드가 붙은 함수를 호출할 수 있다.

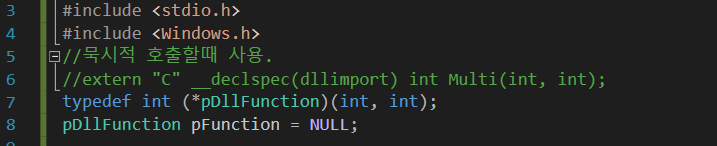
해당 키워드가 안붙으면 외부에서 호출 불가

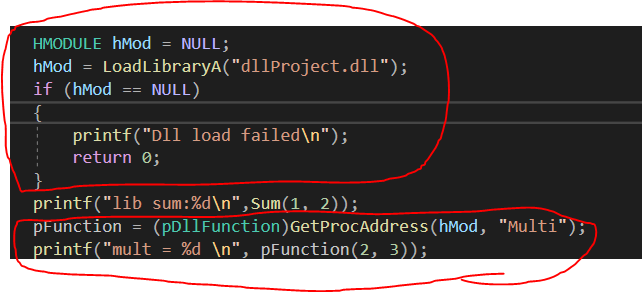
\_\_declspec(dllimport)

Dll에 정의된 함수를 호출한다.

Export로 정의된 함수를 import로 호출.

**-2 명시적 링킹**





LoadLibrary()?

특정경로의 dll을 불러오고, 불러와진 dll 핸들은 HMODULE로 반환된다.

정상적으로 불러오지 못하면 NULL

GetProcAddress()?

특정함수이름의 함수포인터 반환