

어셈블리(Asm + C -> 라이브러리 -> DLL)

먼저 정적 라이브러리를 만들어 줍니다. (C#AsmStatic)

새 프로젝트 구성

정적 라이브러리 C++ Windows 라이브러리

프로젝트 이름(I)
AsmStatic

위치(L)
C:\#

솔루션 이름(M) ⓘ
AsmStatic

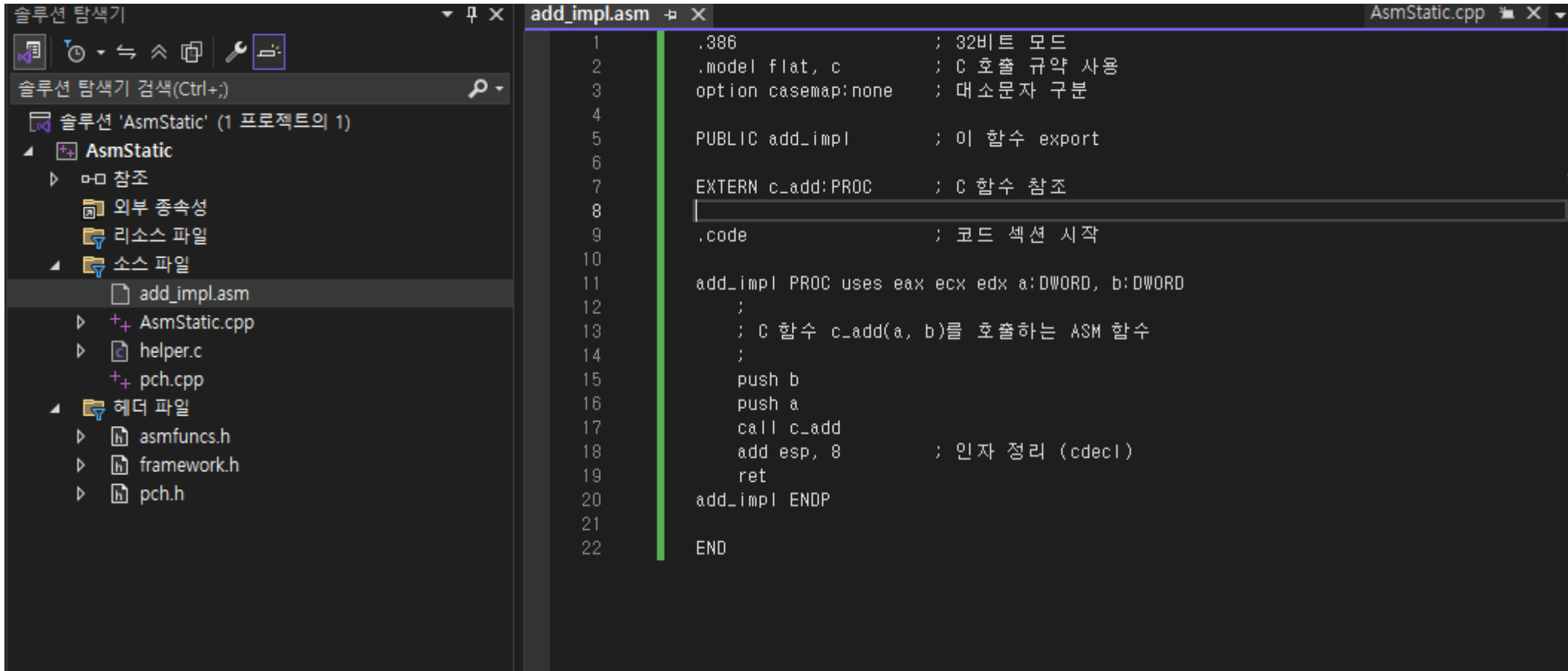
☒ 솔루션 및 프로젝트를 같은 디렉터리에 배치(D)

"C:\#AsmStatic\#"에 프로젝트이(가) 만들어집니다.

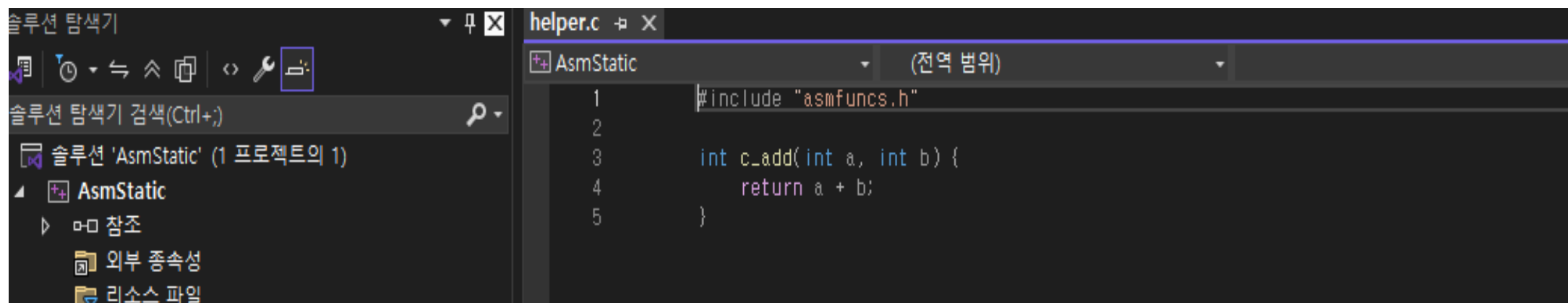
뒤로(B) 만들기(C)

AsmStatic의 프로젝트에 .Asm과 .C 그리고 .h를 추가해줍니다

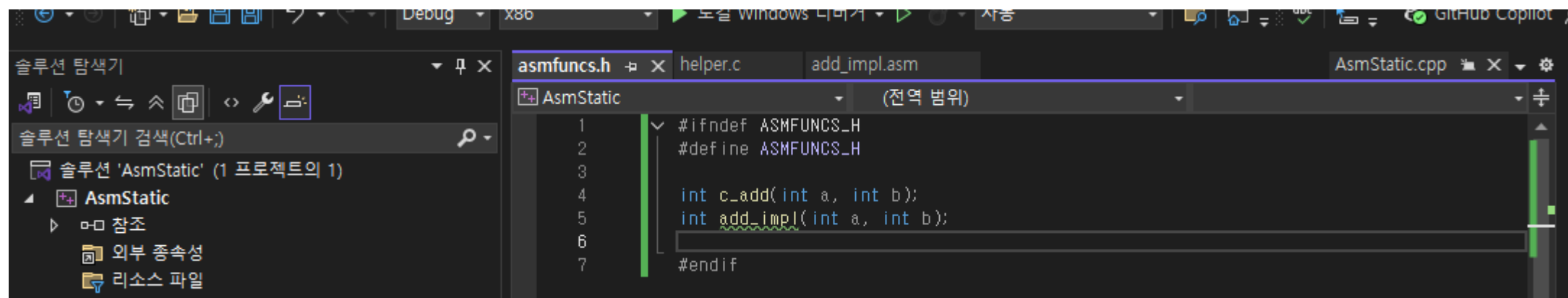
add_impl.asm(Asm파일)



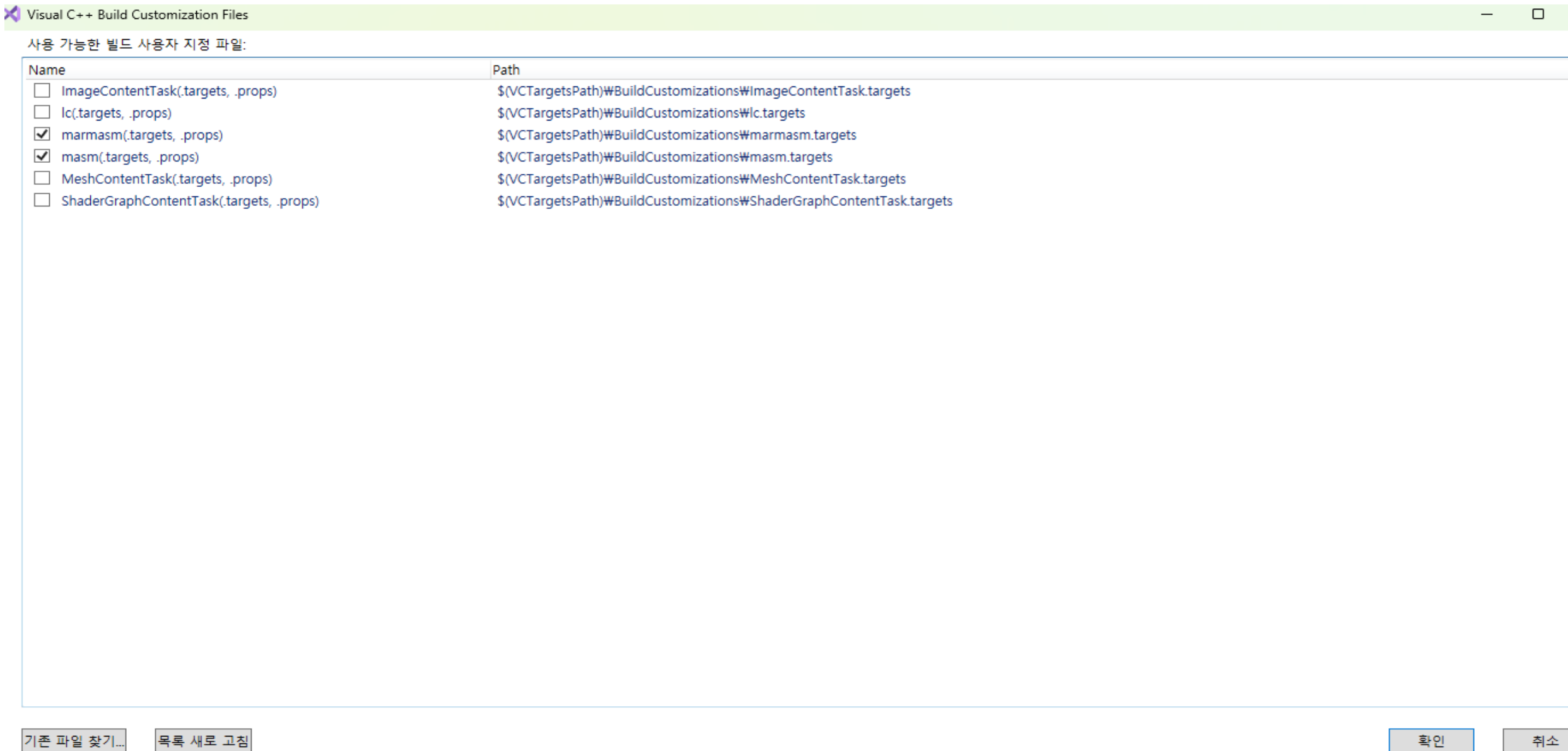
helper.c(C파일)



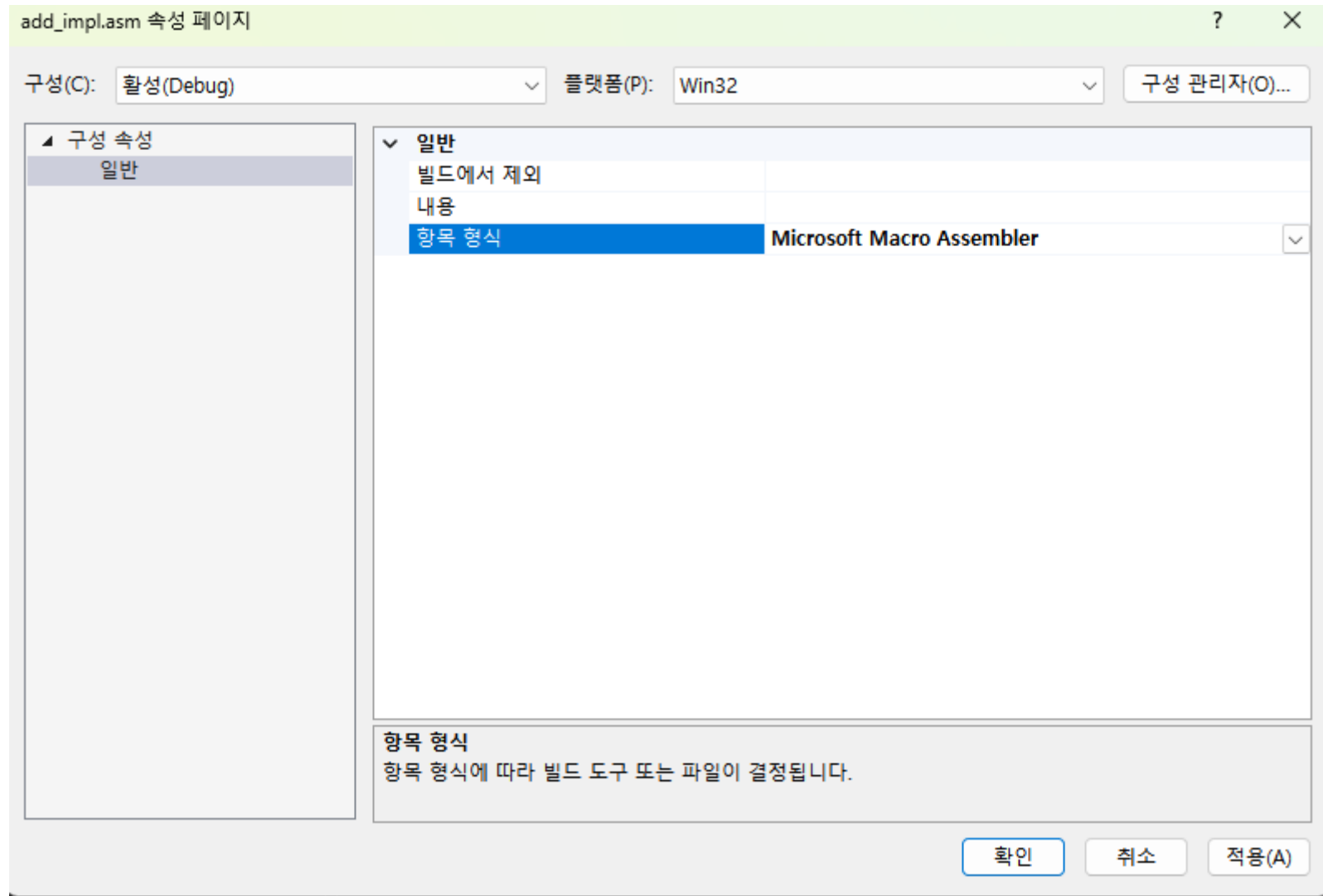
asmfuncs.h(헤더파일)



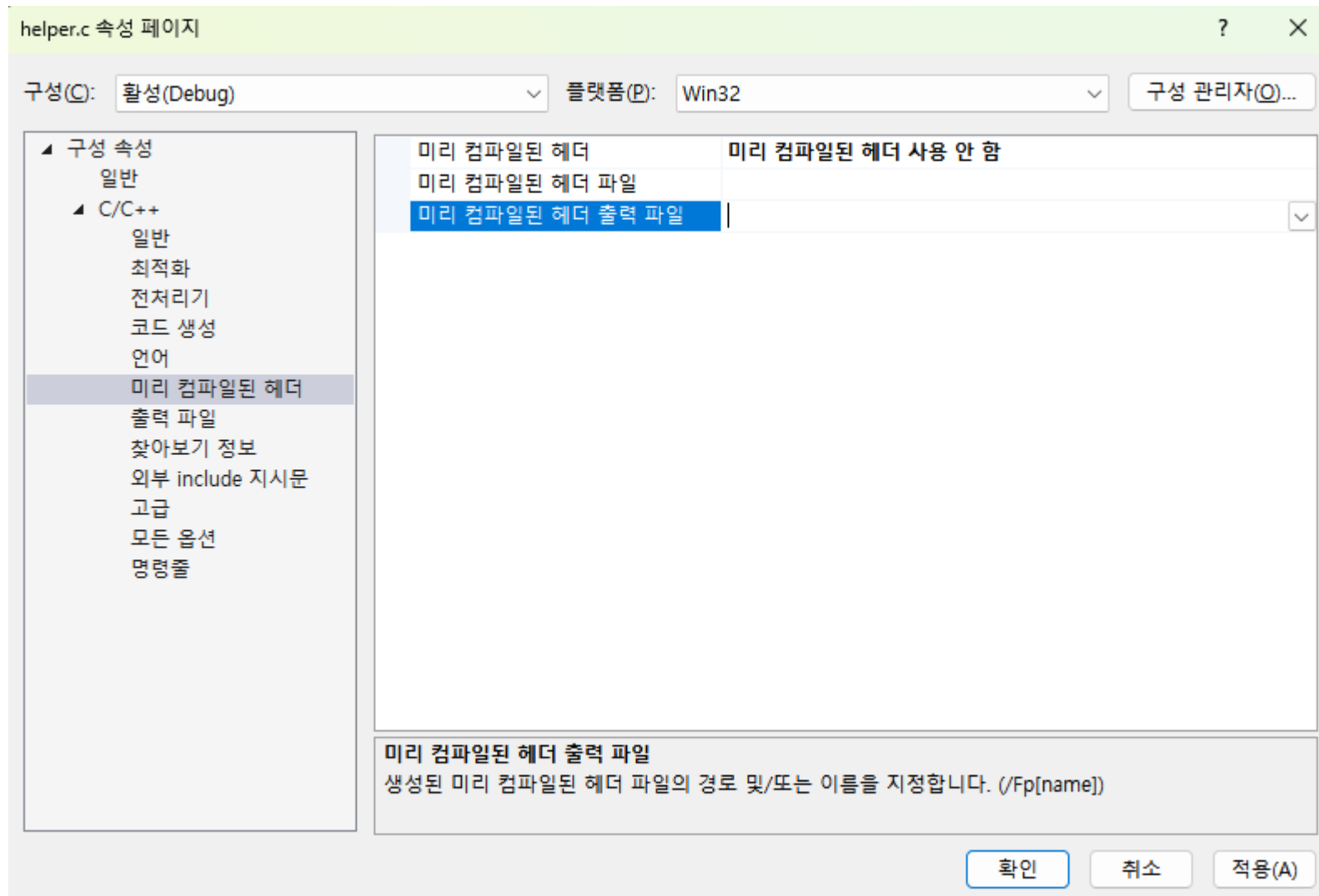
.asm을 실행시키기 위해서 체크박스에 `masm(target, props)`를 체크 해줍니다



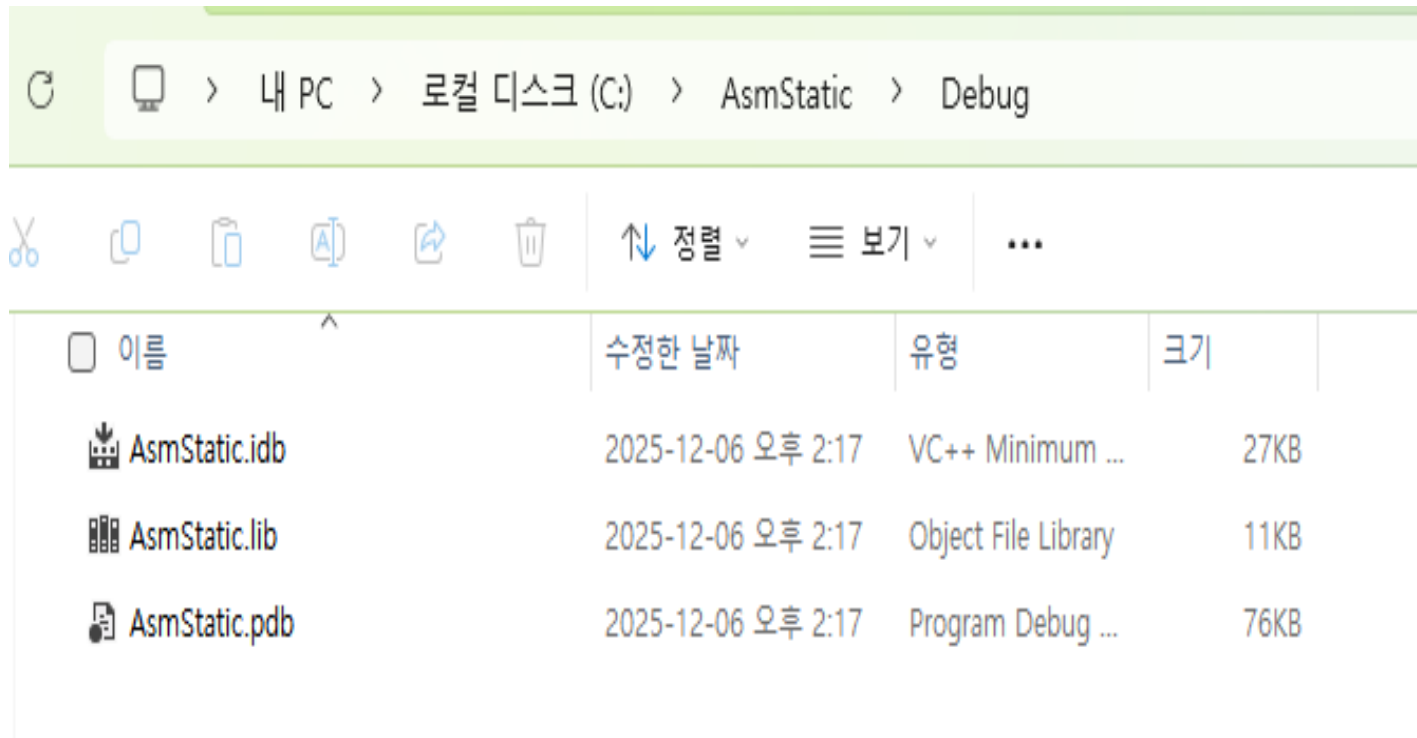
체크 후 add_impl.asm 우클릭 -> 속성 을 하여 활성화된 Microsoft Assembler를 선택



빌드를 실행 시키기 위해 미리 컴파일된 헤더를 "미리 컴파일된 헤더 사용 안함" 선택



솔루션 빌드 후 AsmStatic폴더안의 Debug폴더에 .lib파일(AsmStatic.lib) 생성



.lib파일 생성이 끝났으니 동적 라이브러리를 만들어 줍니다. (C\MyDll)

새 프로젝트 구성

DLL(동적 연결 라이브러리)

C++

Windows

라이브러리

프로젝트 이름(I)

MyDll

위치(L)

C:\

...

솔루션 이름(M) ⓘ

MyDll

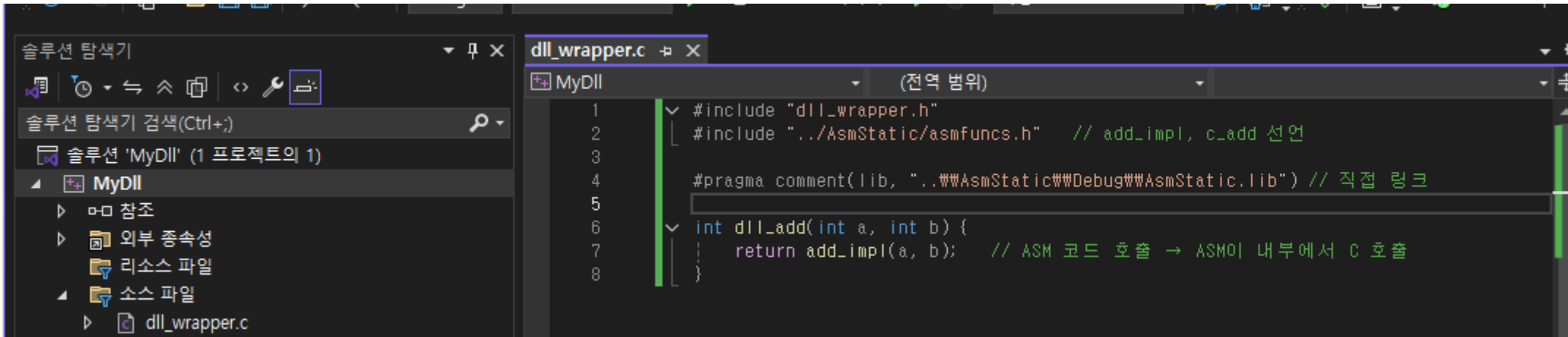
☒ 솔루션 및 프로젝트를 같은 디렉터리에 배치(D)

"C:\MyDll"에 프로젝트이(가) 만들어집니다.

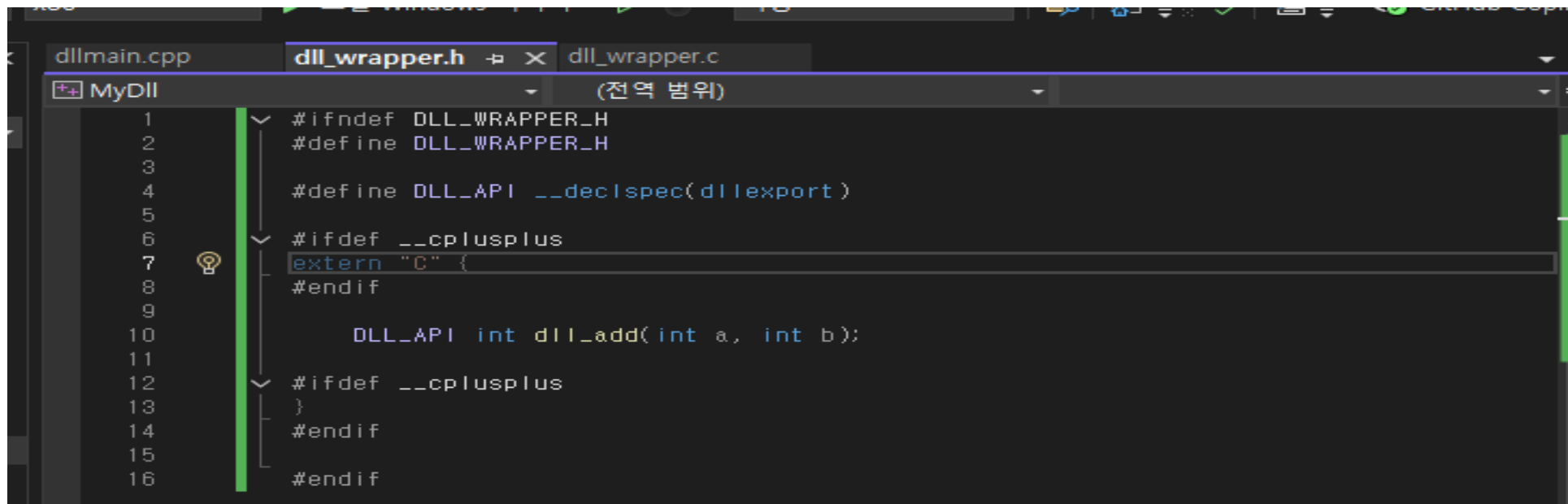
뒤로(B)

만들기(C)

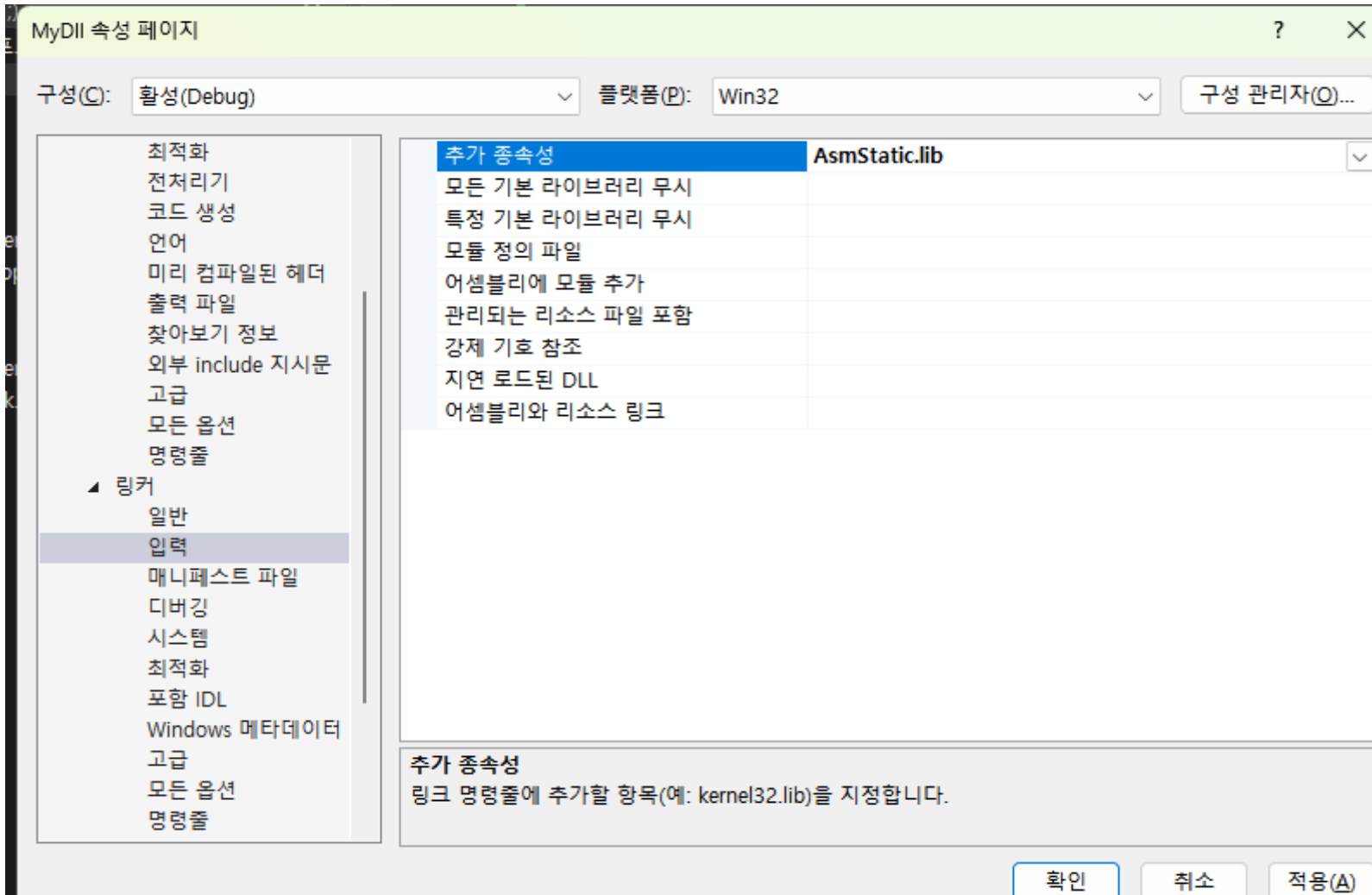
dll_wrapper.c(C파일)



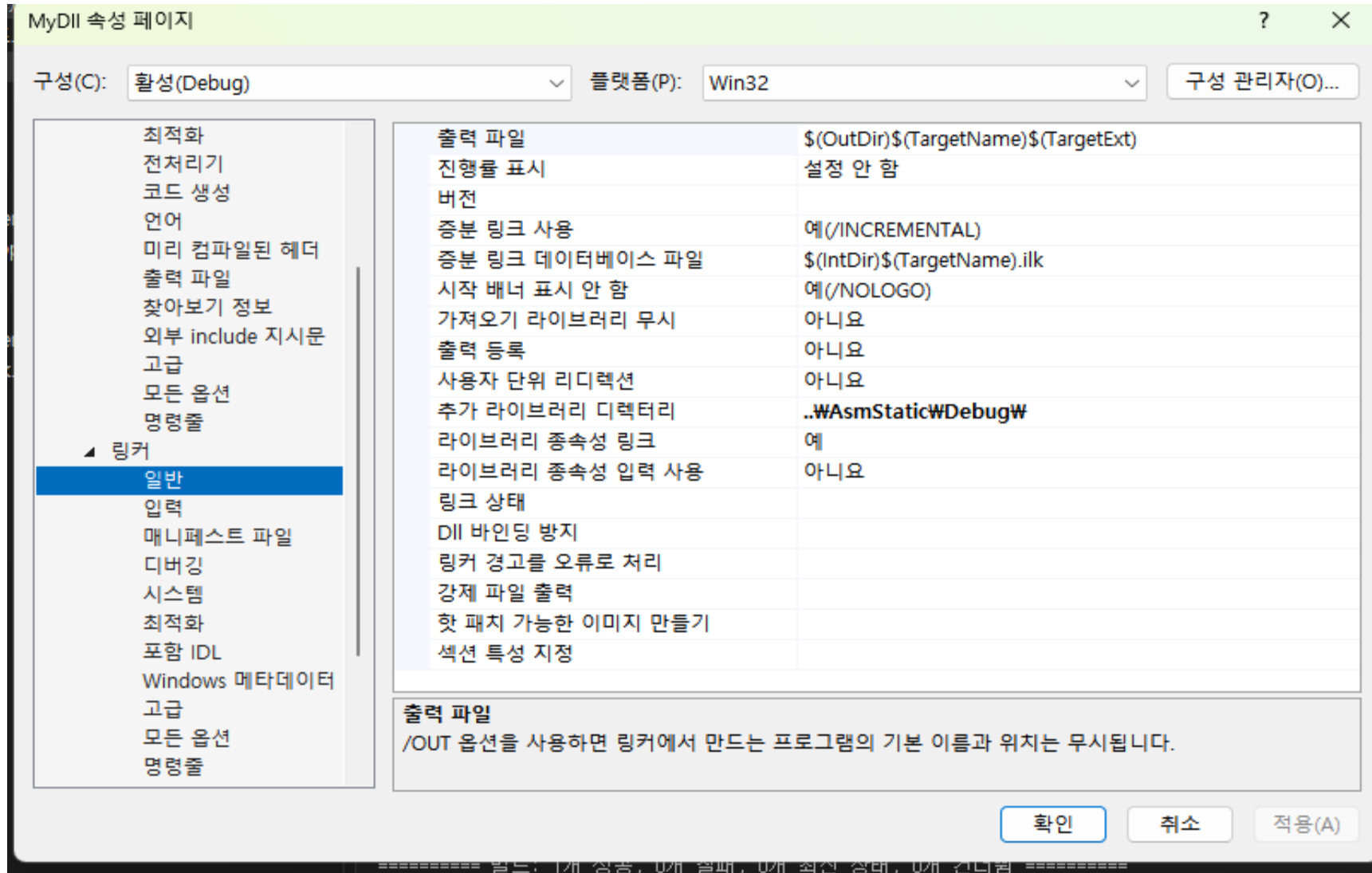
dll_wrapper.h(헤더파일)



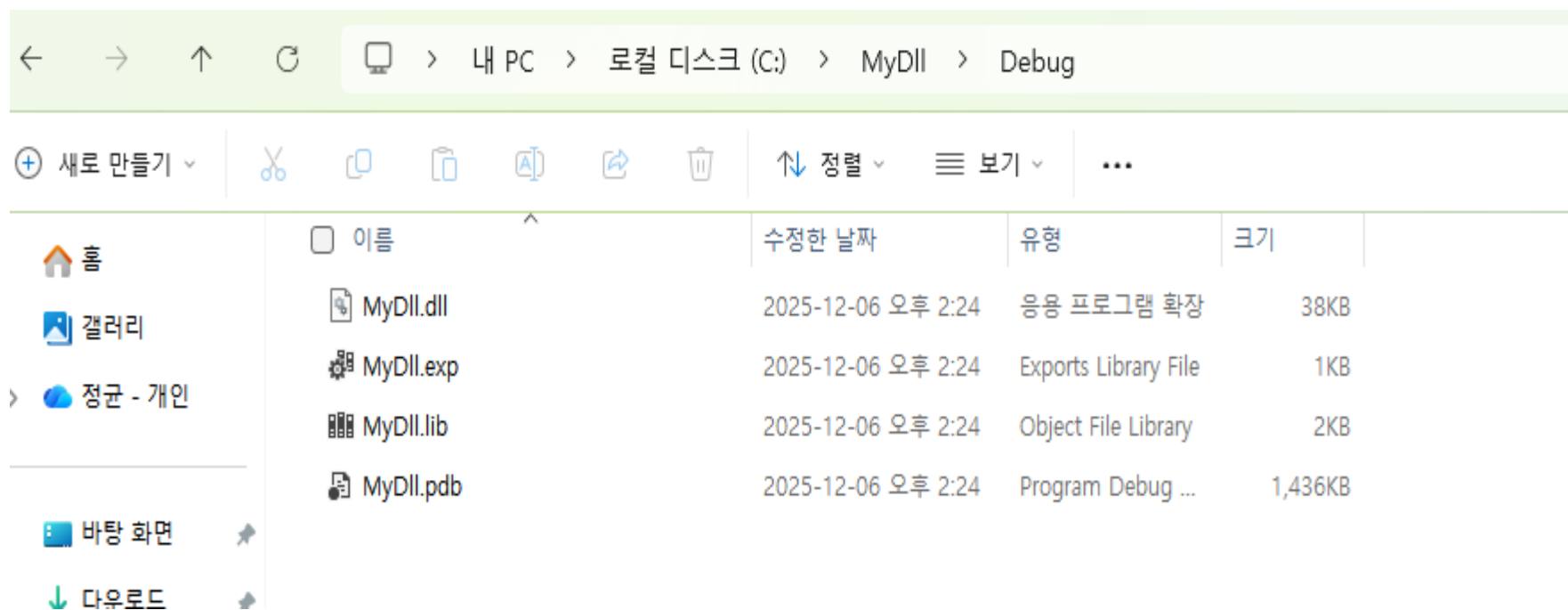
아까 만들어 둔 AsmStatic.lib을 MyDll 프로젝트에 링킹 해줍니다



솔루션 빌드를 할 때 .lib의 파일 위치를 알아야 솔루션 빌드가 성공합니다. AsmStatic.lib의 위치는 C\WasmStatic\Debug 폴더 안에 위치하여 "추가 라이브러리 디렉터리" 에 위치 추가



이전 설정 그대로 솔루션 빌드를 하면 C\MyDll\Debug 폴더에 .exp, .lib, .dll파일이 생성이 됩니다.



마지막으로 exe파일을 만들 프로젝트를 만듭니다. (C\#Consol)

새 프로젝트 구성

콘솔 앱

C++

Windows

콘솔

프로젝트 이름(I)
Consol

위치(L)
C:\#

솔루션(S)
새 솔루션 만들기

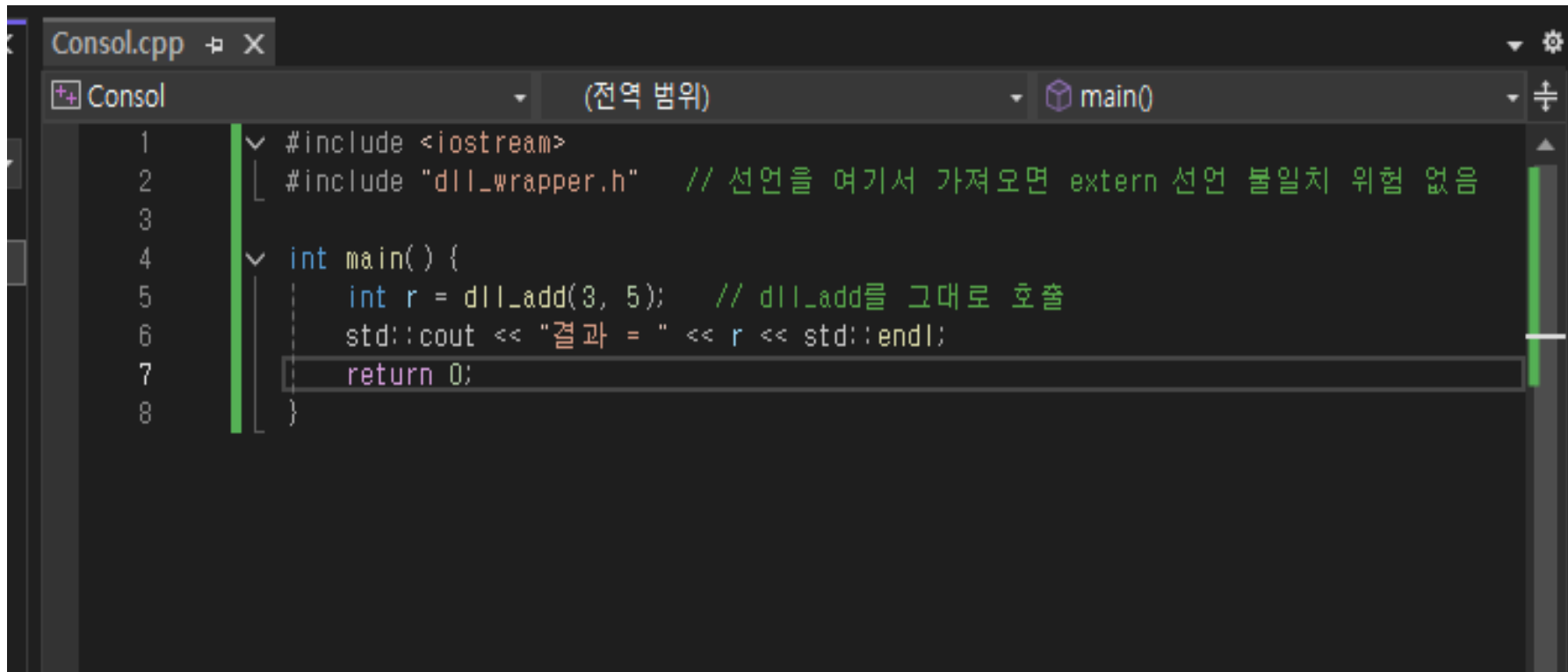
솔루션 이름(M) ⓘ
Consol

☒ 솔루션 및 프로젝트를 같은 디렉터리에 배치(D)

"C:\#Consol\#"에 프로젝트이(가) 만들어집니다.

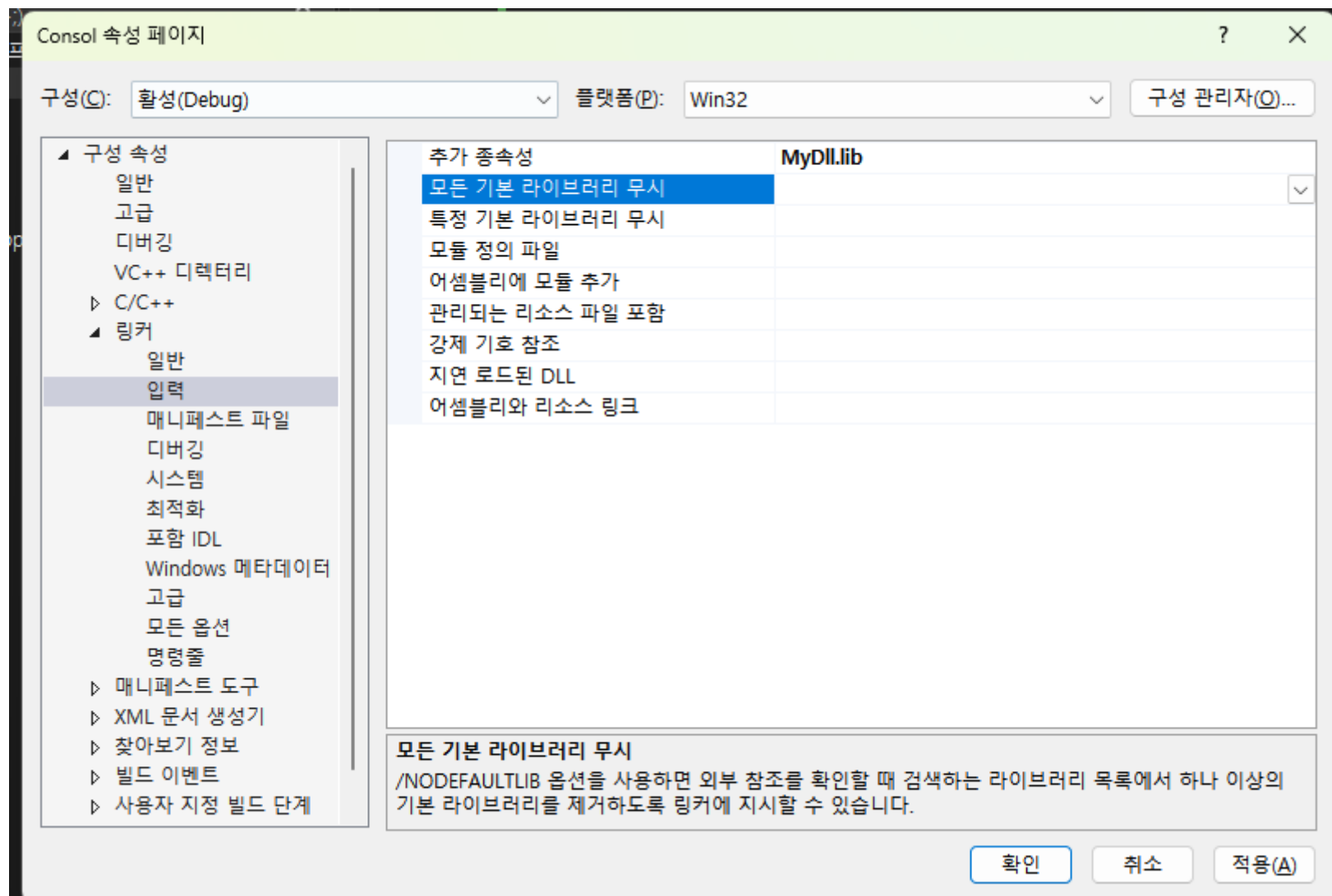
뒤로(B) 만들기(C)

실행할 코드를 Consol.cpp에 넣습니다.

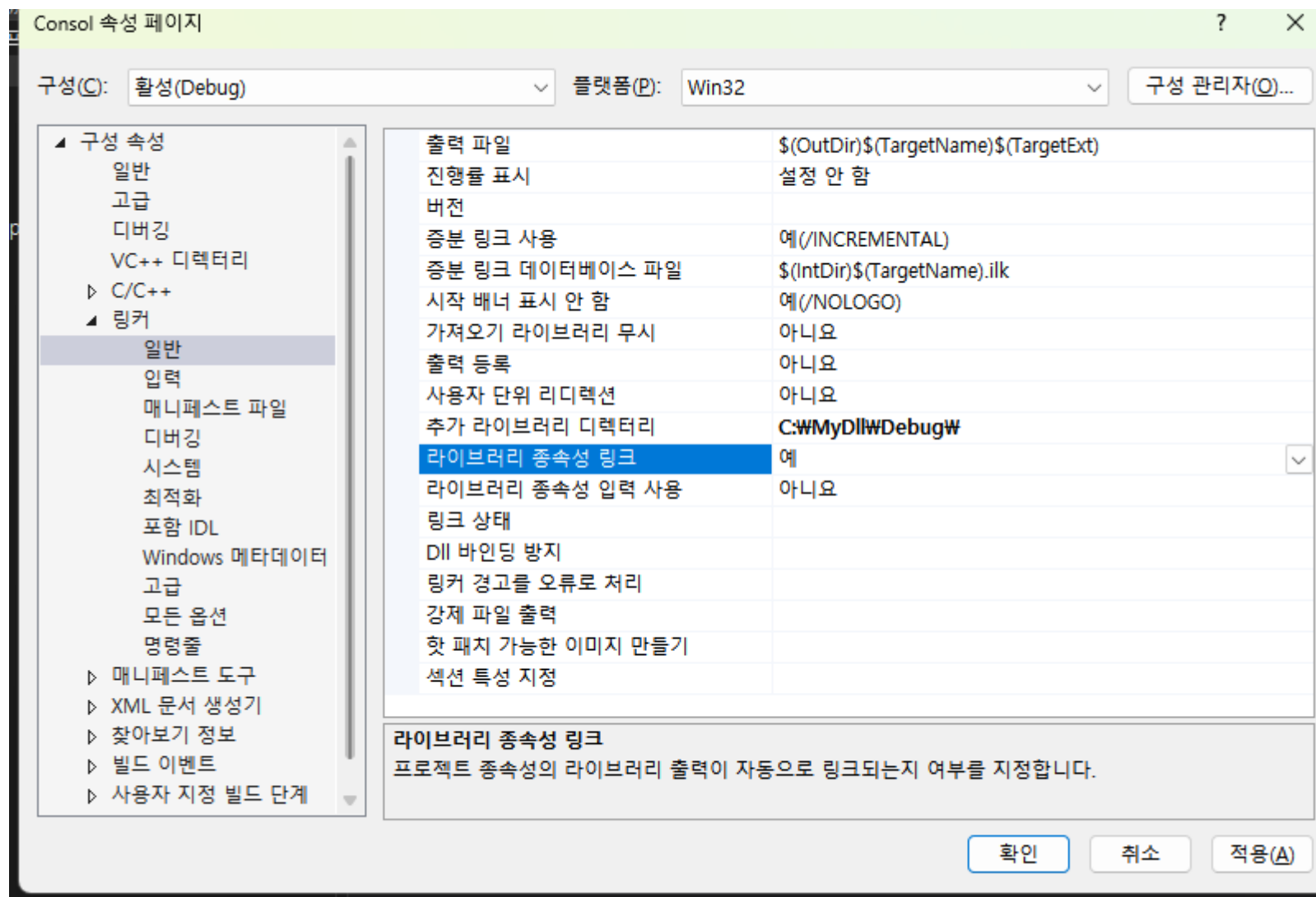


```
1  #include <iostream>
2  #include "dll_wrapper.h" // 선언을 여기서 가져오면 extern 선언 불일치 위험 없음
3
4  int main() {
5      int r = dll_add(3, 5); // dll_add를 그대로 호출
6      std::cout << "결과 = " << r << std::endl;
7      return 0;
8  }
```

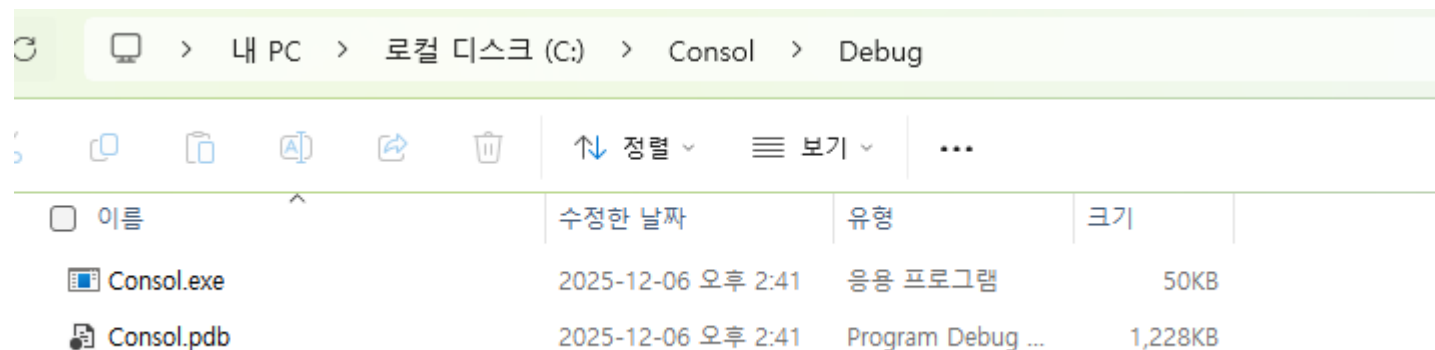
아까 MyDll 프로젝트에 만든 MyDll.lib을 링킹 합니다



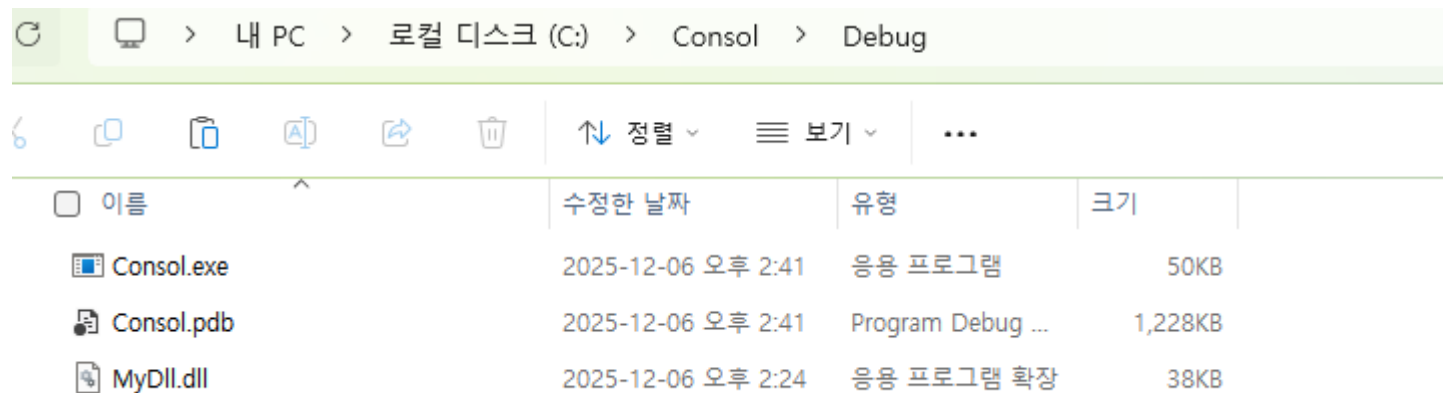
MyDll.lib의 주소를 "추가 라이브러리 디렉터리"에 넣습니다.



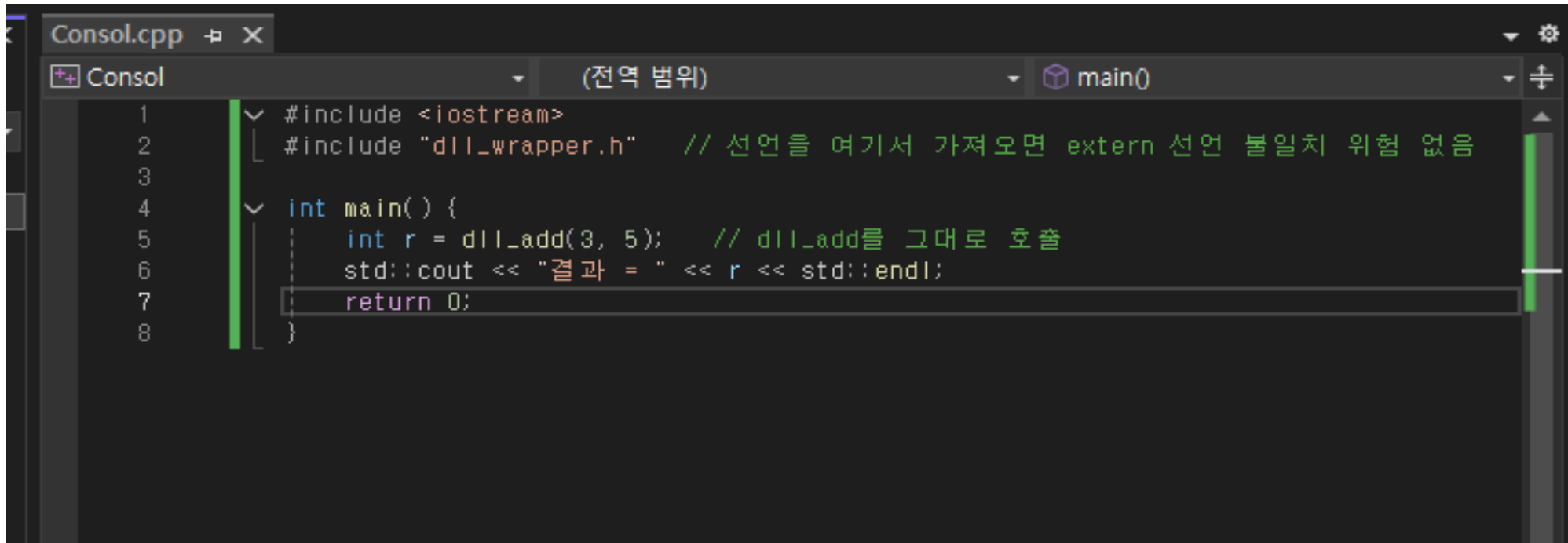
그 후 솔루션 빌드를 하면 CWConsolWDebug 폴더에 .exe가 생성이 됩니다.



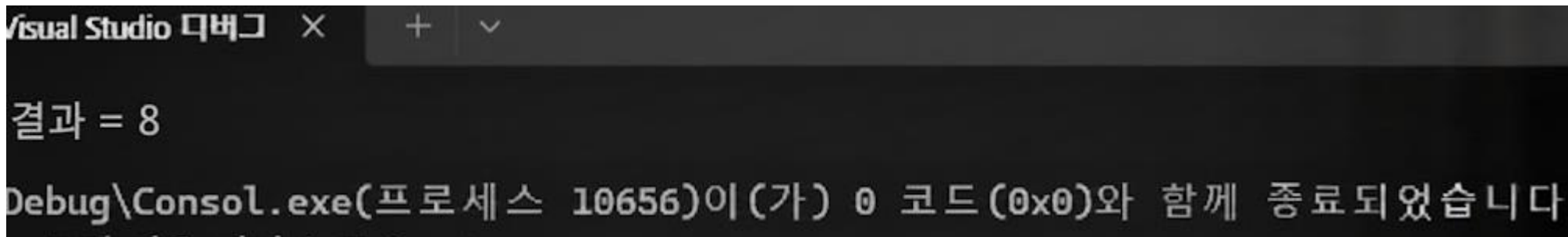
하지만 .exe만으로 실행을 못하기에 CWMyDllWDebug 폴더에 있는 MyDll.dll 을 복사를 하여 .exe파일과 같은 폴더에 위치 시킵니다.



그 후 visual studio 2022에 들어가 실행을 시키면 코드에 작성한 dll_add(3,5)가 작동하여 r에 전달이 되고, 결과값 r을 콘솔에 나타내 줍니다.



```
1  #include <iostream>
2  #include "dll_wrapper.h" // 선언을 여기서 가져오면 extern 선언 불일치 위험 없음
3
4  int main() {
5      int r = dll_add(3, 5); // dll_add를 그대로 호출
6      std::cout << "결과 = " << r << std::endl;
7      return 0;
8  }
```



Visual Studio 디버그

결과 = 8

Debug\Consol.exe(프로세스 10656)이(가) 0 코드(0x0)와 함께 종료되었습니다