



11. 프로젝트(project) 만들기

- 프로그램 한 개를 만들기 위해 여러 개의 소스가 필요한 경우, 프로젝트 파일을 만들어야 합니다.
- 우리가 예로 만들어 볼 프로젝트는 다음 3개의 파일과 관련이 있습니다. 첫 번째 파일은 sub.cpp인데 전역 변수와 함수가 각각 한 개씩 있습니다.

```
//sub.cpp
```

```
int global=100;
```

```
void swap(int *i,int *j) {
```

```
    int t=*i;
```

```
    *i=*j;
```

```
    *j=t;
```

```
}//swap
```

-
- 두 번째 파일은 sub.h입니다■.

```
//sub.h
```

```
extern■ int global;
```

```
void swap(int *i,int *j);
```

- 마지막으로 sub.cpp를 이용하는 main.cpp입니다.

```
//main.cpp
```

```
#include <stdio.h>
```

```
#include "sub.h"
```

```
void main() {
```

```
    int i=2,j=3;
```

```
    swap(&i,&j);
```

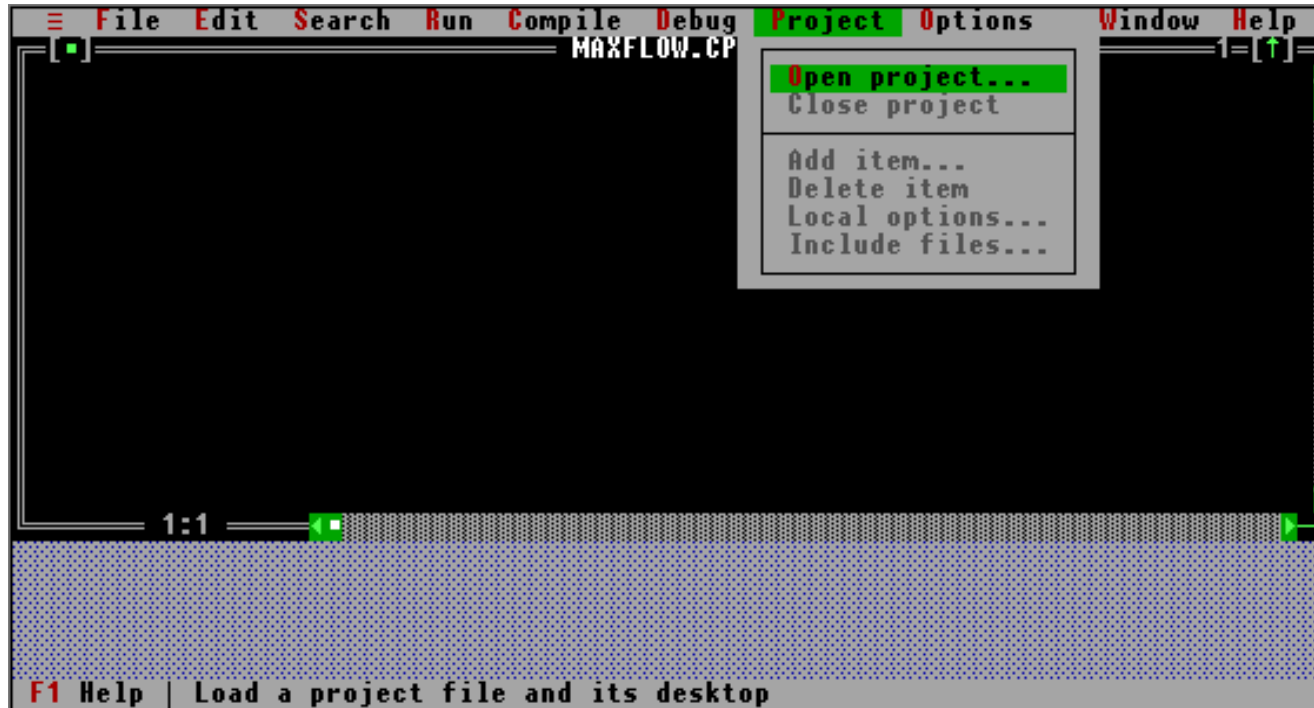
```
    printf("%d,%d,%d\n",i,j,global);
```

```
    //        3 2 100
```

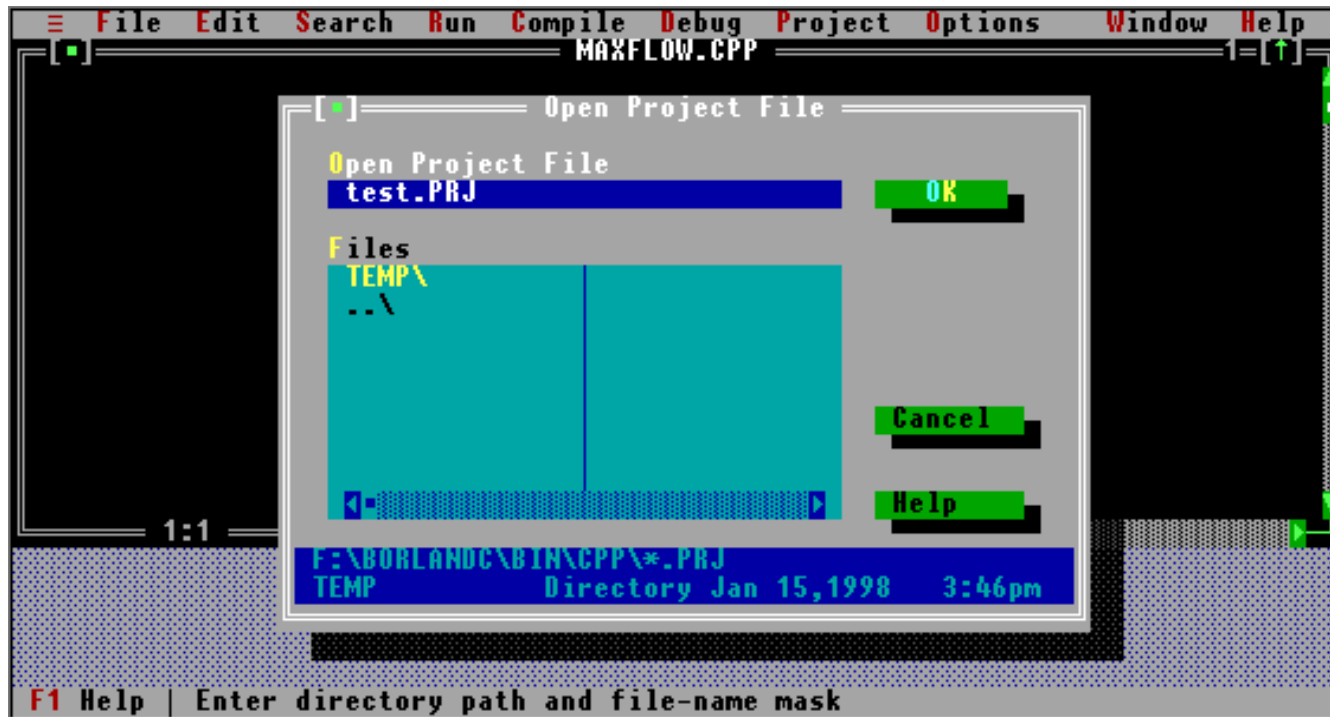
```
}//main
```



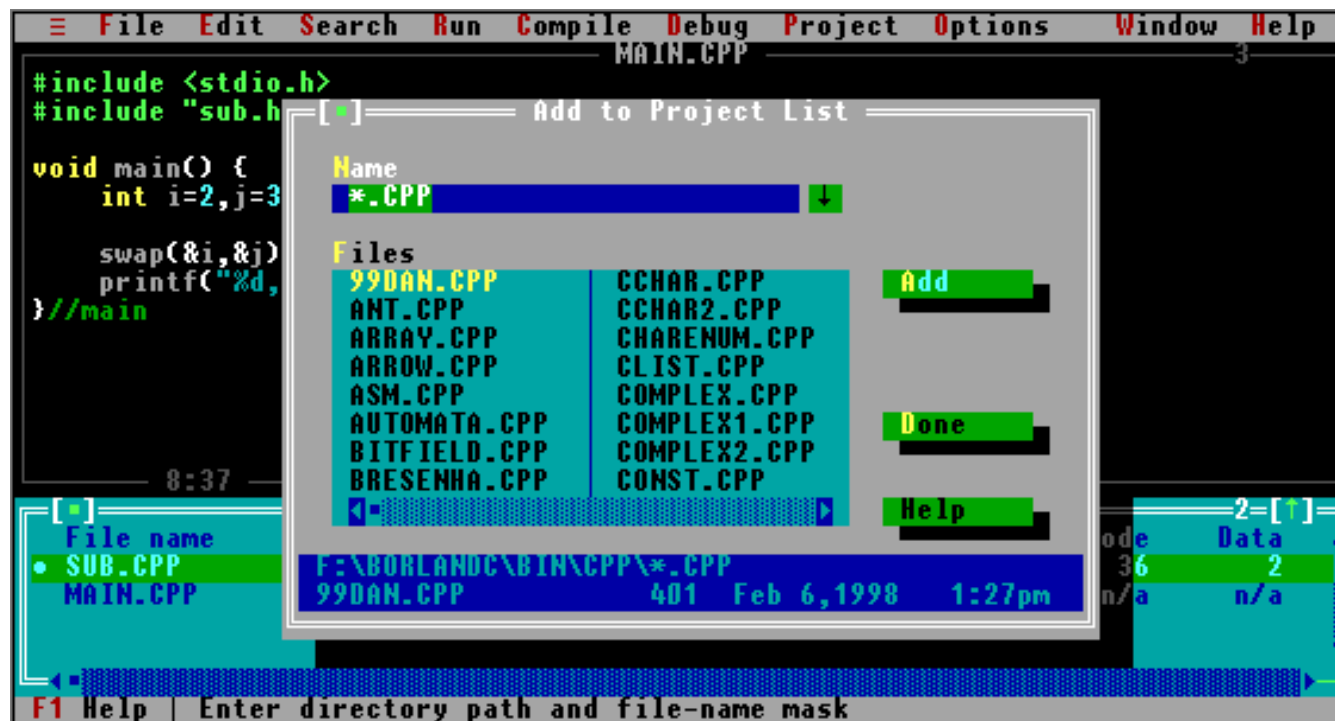
볼런드(Borland) C++ 3.1인 경우



Project->Open Project의 선택

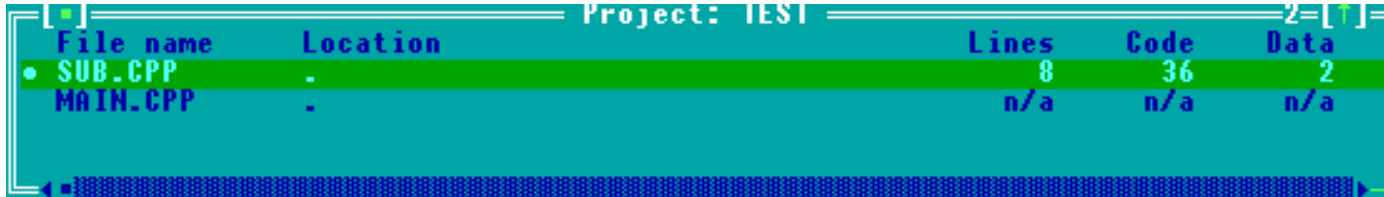


test.prj 프로젝트 이름의 지정



프로젝트에 sub.cpp와 main.cpp를 Add합니다.

- 프로젝트 윈도우가 나타나면서 항목을 추가하라는 대화상자가 나타납니다. 이 때 sub.cpp와 main.cpp를 추가합니다.



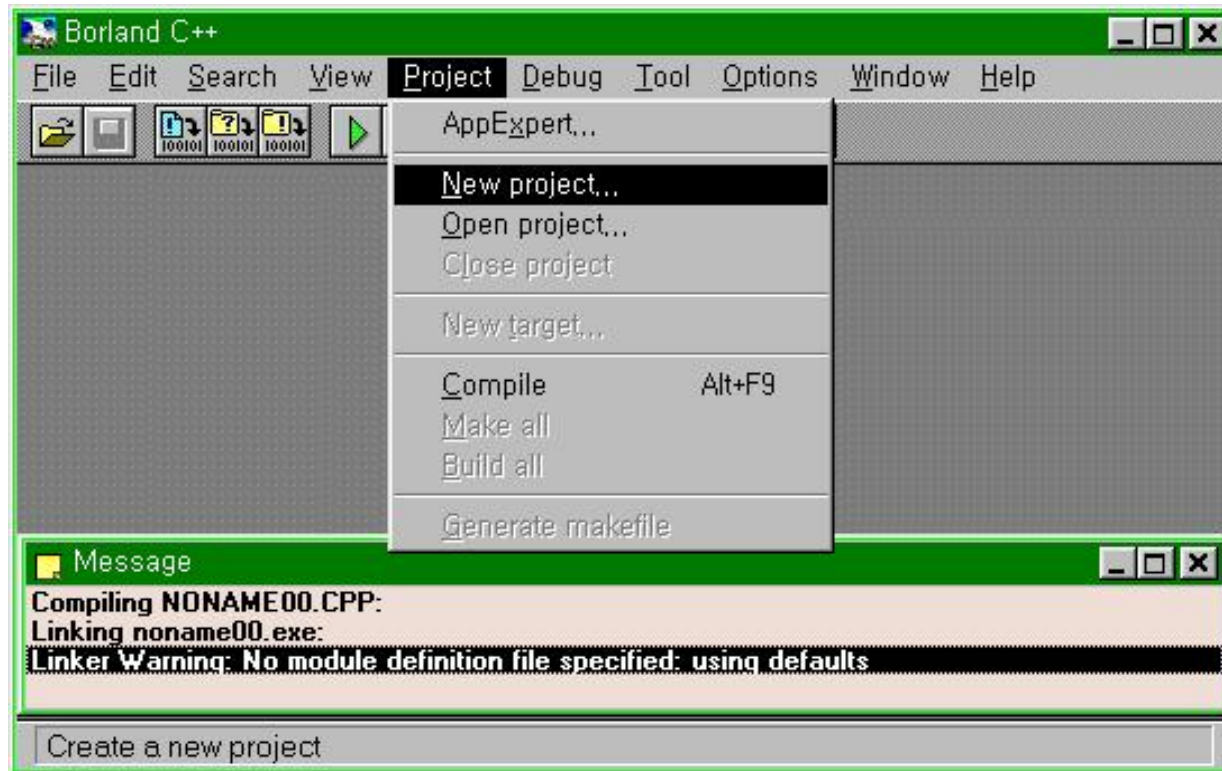
| File name | Location | Lines | Code | Data |
|-----------|----------|-------|------|------|
| • SUB.CPP | - | 8 | 36 | 2 |
| MAIN.CPP | - | n/a | n/a | n/a |



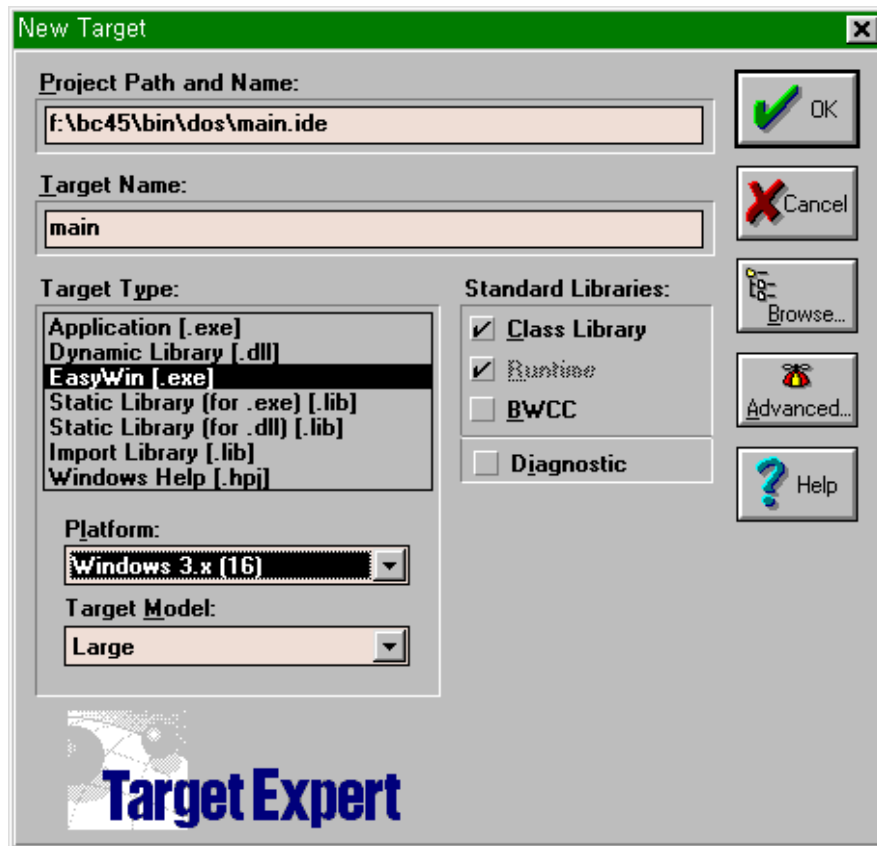
모든 파일이 추가된 후의 Project 윈도우: 항목을 추가/삭제하기 위해 Insert/Delete 키를 사용합니다.



볼랜드 C++ 4.5인 경우



Project->New project의 선택

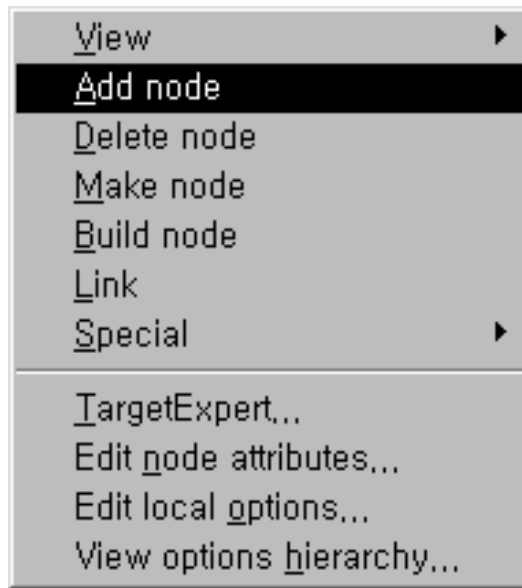


경로와 프로젝트 이름 main.ide의 지정

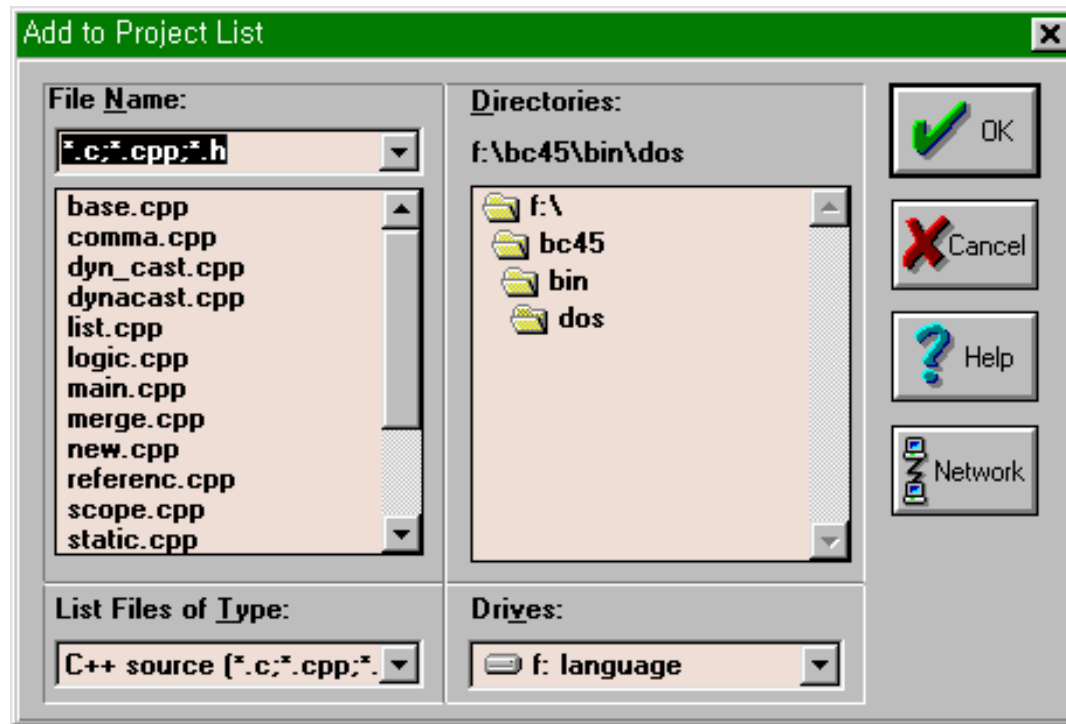
- 경로와 프로젝트 이름 main.ide를 지정합니다. 볼런드 C++ 4.5의 프로젝트 파일의 확장자는 .IDE(Integrated Development Environment)입니다.



생성된 Project 윈도우



메뉴에서 Add node의 선택

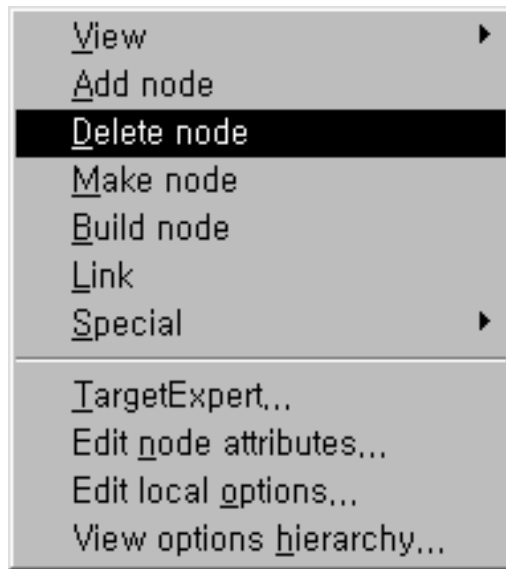


sub.cpp와 main.cpp의 Add

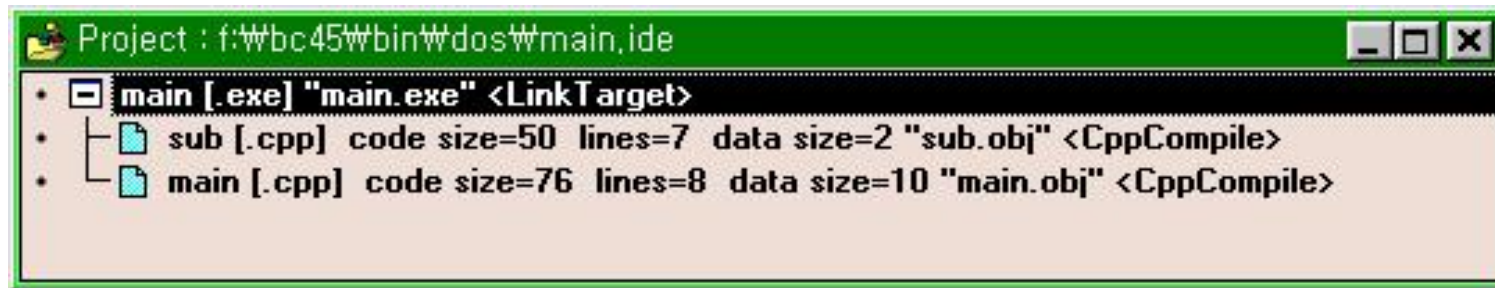
- 대화상자에서 sub.cpp를 추가합니다. main.cpp는 이미 지정되어 있으므로, 추가할 필요가 없습니다.



sub.cpp와 main.cpp가 추가된 Project 윈도우



main.def와 main.rc의 삭제



완성된 Project 윈도우

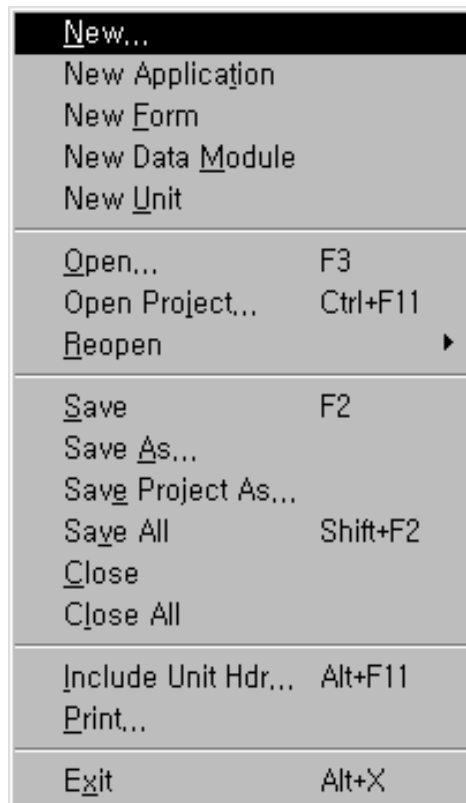


볼런드 C++ 빌더(Builder)인 경우

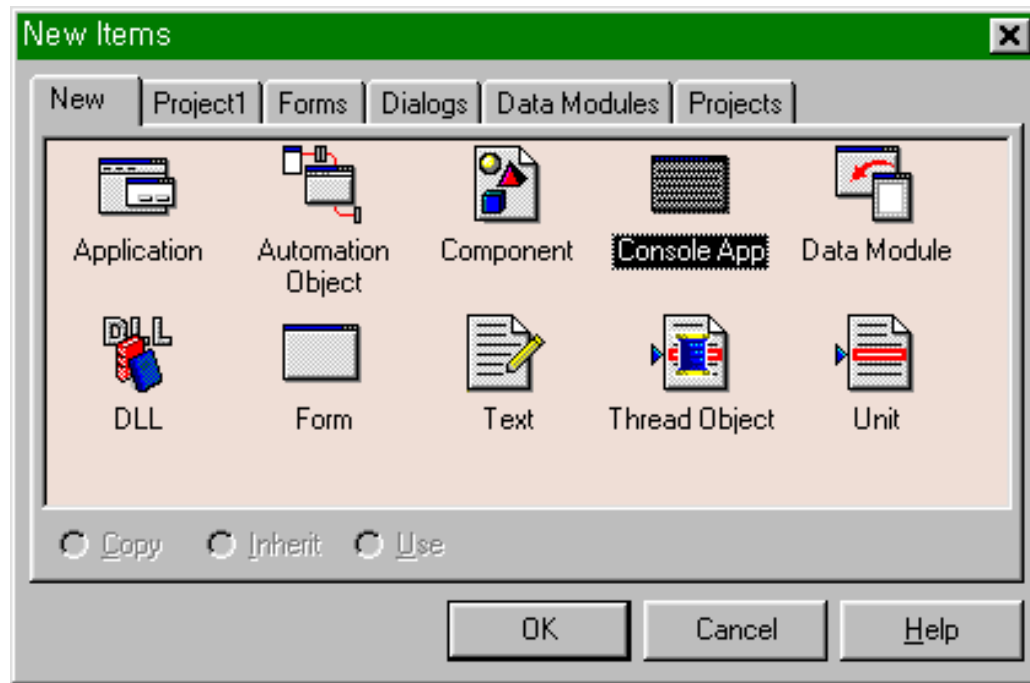


C++ 빌더의 메인 윈도우

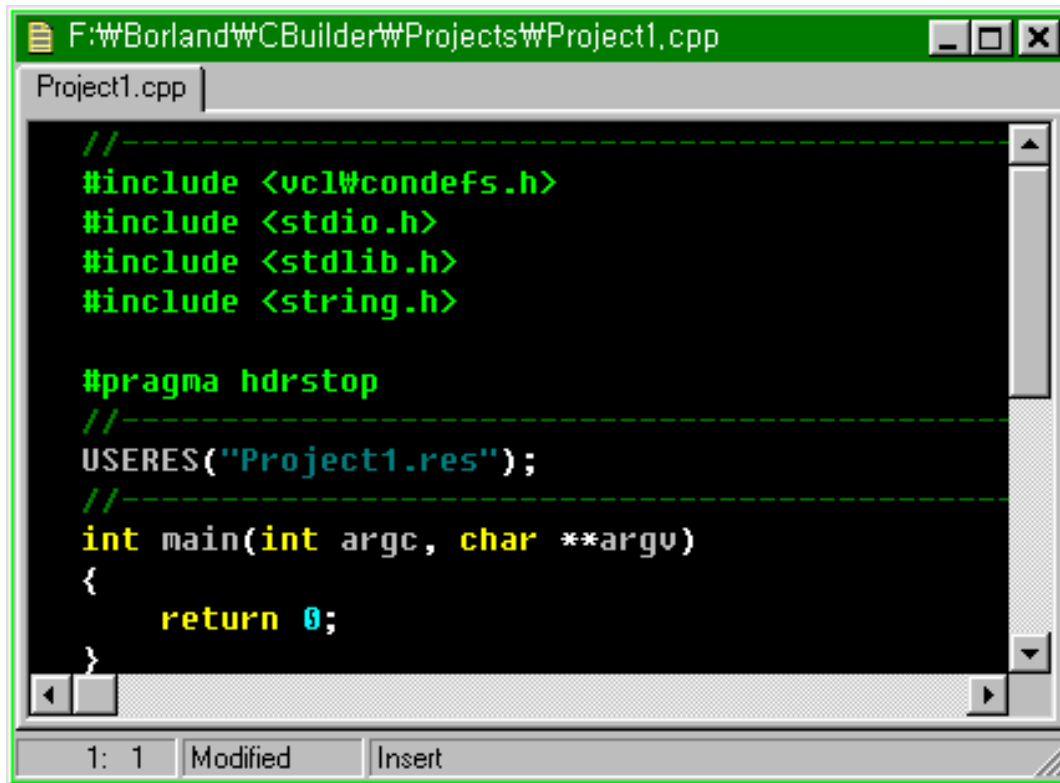
- C++ 빌더의 메인 윈도우는 위 그림과 같습니다.



File->New의 선택



Console App의 선택



```
F:\Borland\WCBUILDER\Projects\Project1.cpp
Project1.cpp
//-----
#include <vcl\condefs.h>
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

#pragma hdrstop
//-----
USERES("Project1.res");
//-----
int main(int argc, char **argv)
{
    return 0;
}
```

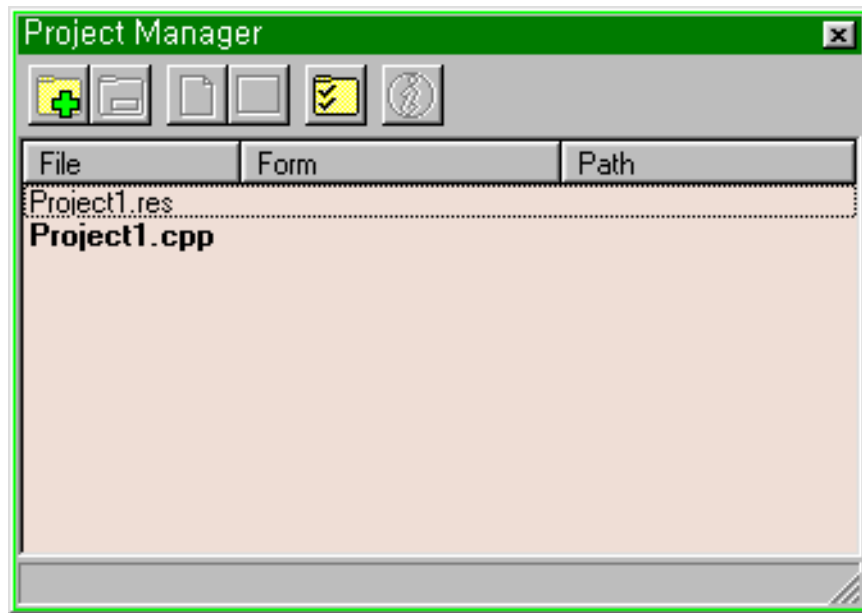


자동으로 생성된 project1.cpp

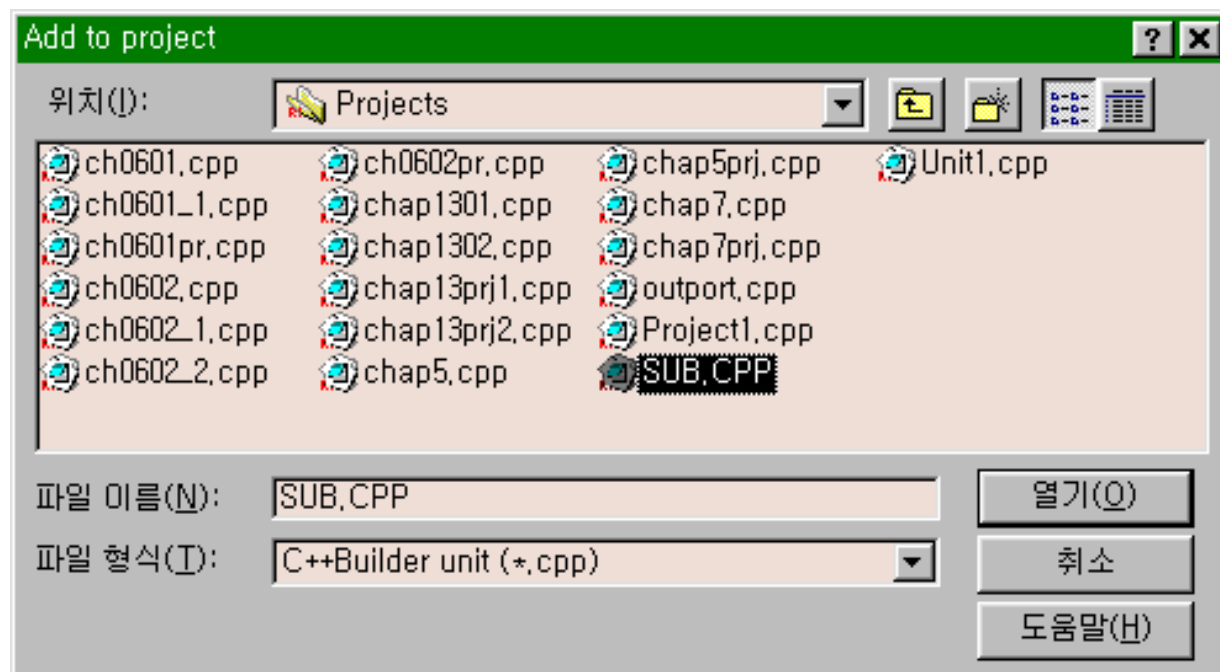
| | |
|-----------------------------|--------------|
| <u>P</u> roject Manager | Ctrl+Alt+F11 |
| Project Source | |
| Project <u>M</u> akefile | |
| <u>O</u> bject Inspector | F11 |
| <u>A</u> lignment Palette | |
| Component <u>L</u> ist | |
| Window List,.. | Alt+0 |
| Call <u>S</u> tack | Ctrl+F3 |
| <u>T</u> hreads | |
| <u>C</u> PU | Alt+F2 |
| Breakpoints | |
| <u>W</u> atches | |
| <u>T</u> oggle Form/Unit | F12 |
| <u>U</u> nits... | Ctrl+F12 |
| <u>F</u> orms... | Shift+F12 |
| New <u>E</u> dit Window | |
| ✓ Tool <u>b</u> ar | |
| ✓ Component <u>P</u> alette | |



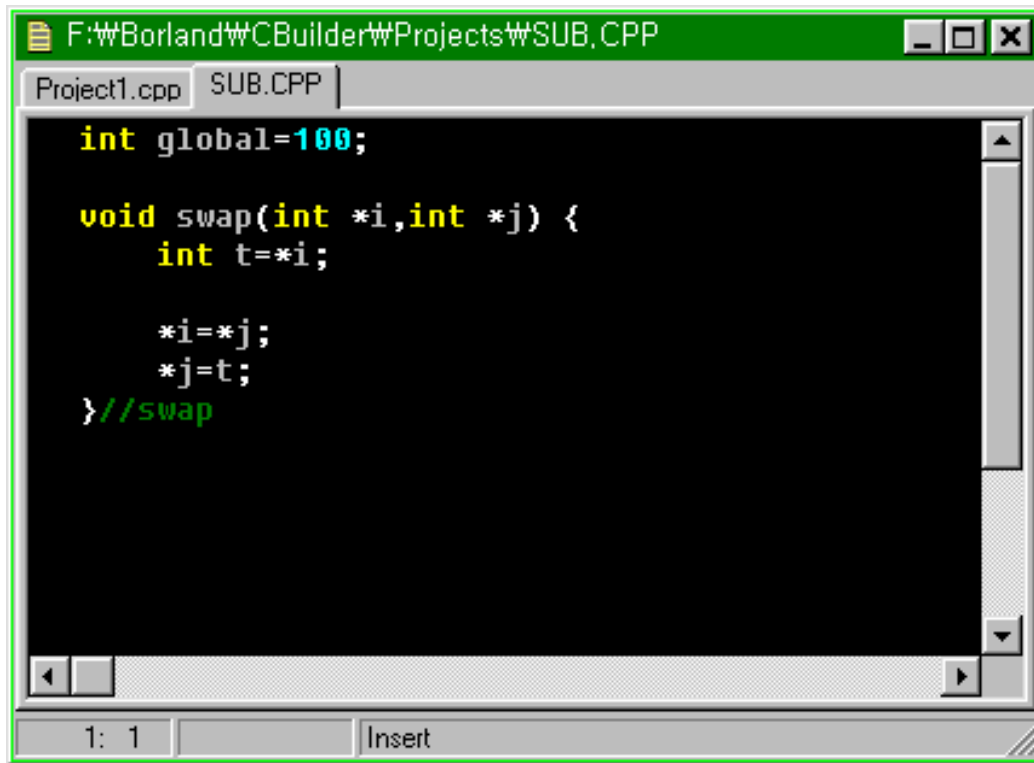
View->Project Manager의 선택



Project Manager의 모양



sub.cpp를 추가



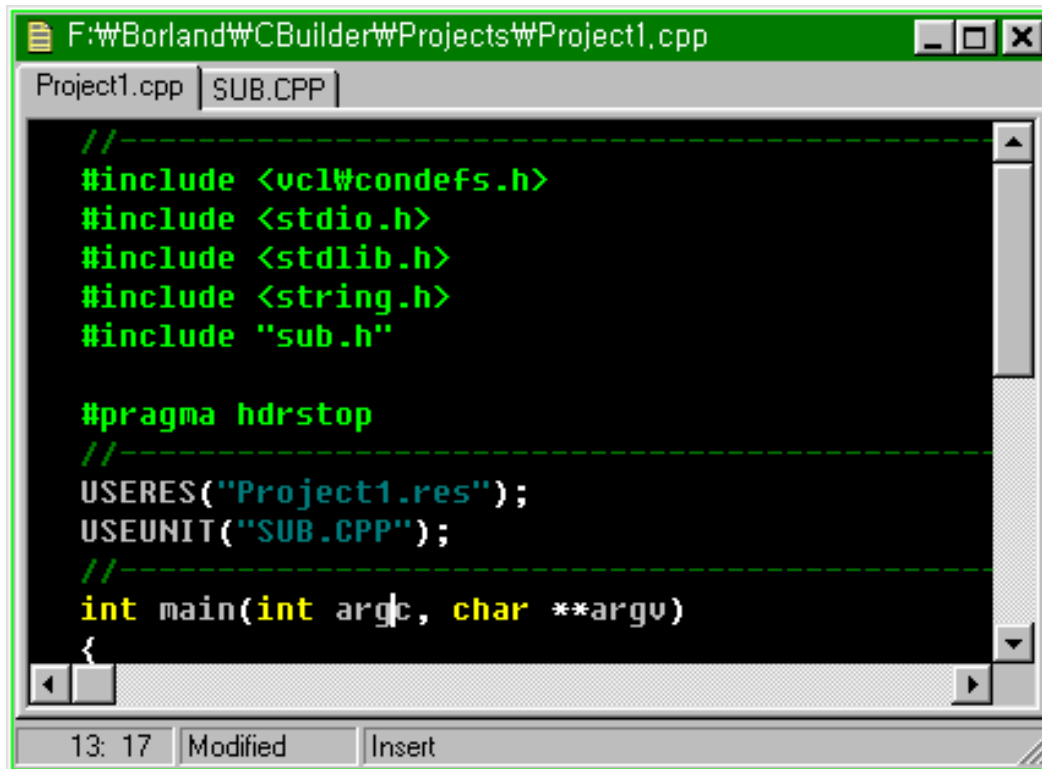
```
int global=100;

void swap(int *i,int *j) {
    int t=*i;

    *i=*j;
    *j=t;
} //swap
```



추가된 sub.cpp가 코드 윈도우에 나타난 모습



```
F:\Borland\CBUILDER\Projects\Project1.cpp
Project1.cpp | SUB.CPP |
//-----
#include <vcl\condefs.h>
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "sub.h"

#pragma hdrstop
//-----
USERES("Project1.res");
USEUNIT("SUB.CPP");
//-----
int main(int argc, char **argv)
{
}
```

13: 17 Modified Insert

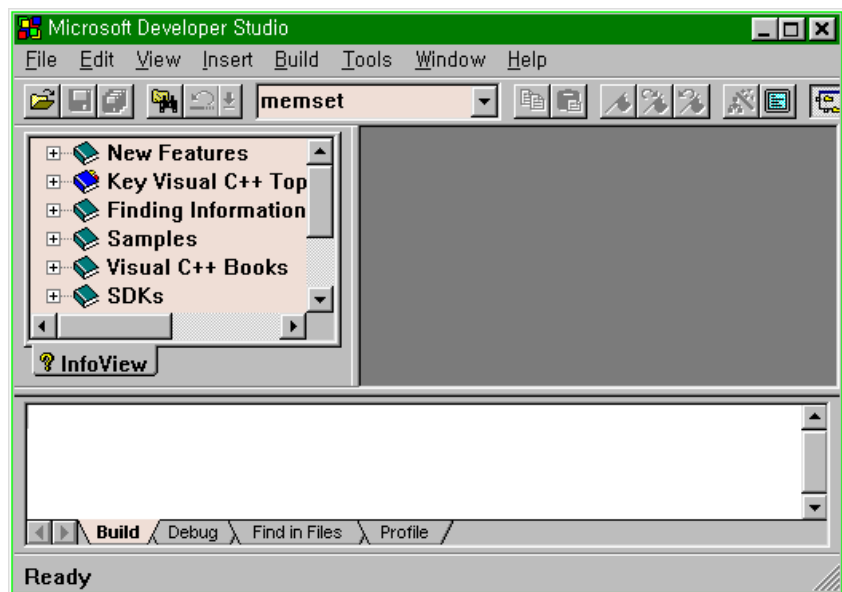


project1.cpp를 수정; #include "sub.h"의 추가

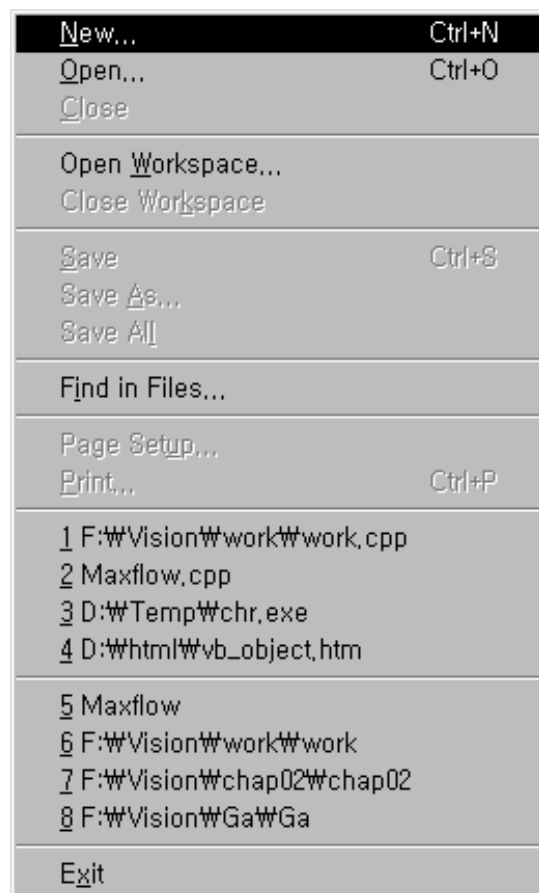


마이크로소프트 비주얼(Visual) C++ 4.2인 경우

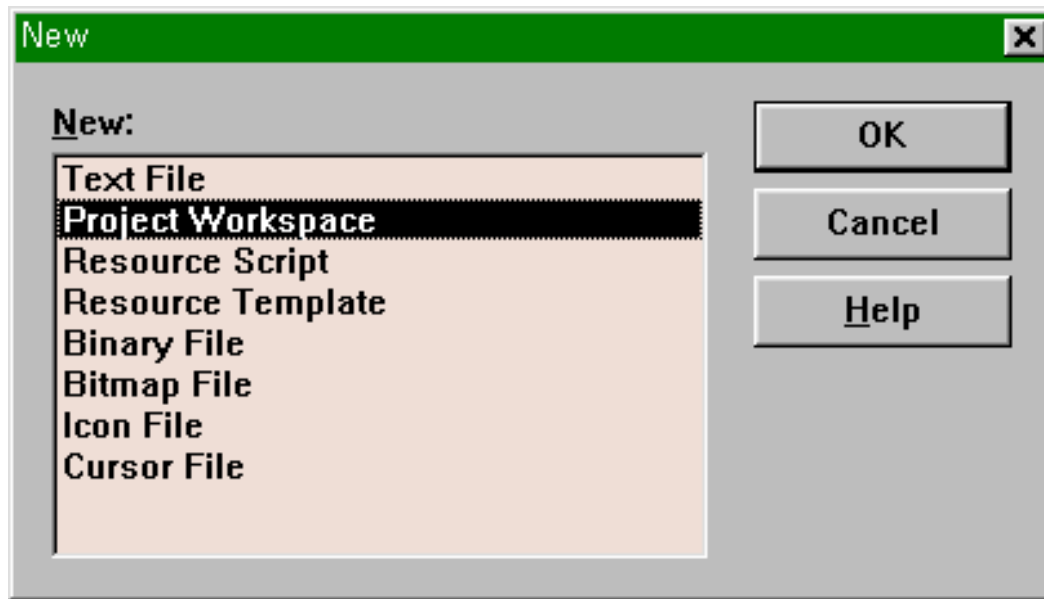
- Visual C++는 이 글을 쓰는 시점에 2017 버전까지 발매되어 있습니다. Visual C++ 4.2의 개발 환경은 아래 그림과 같습니다.



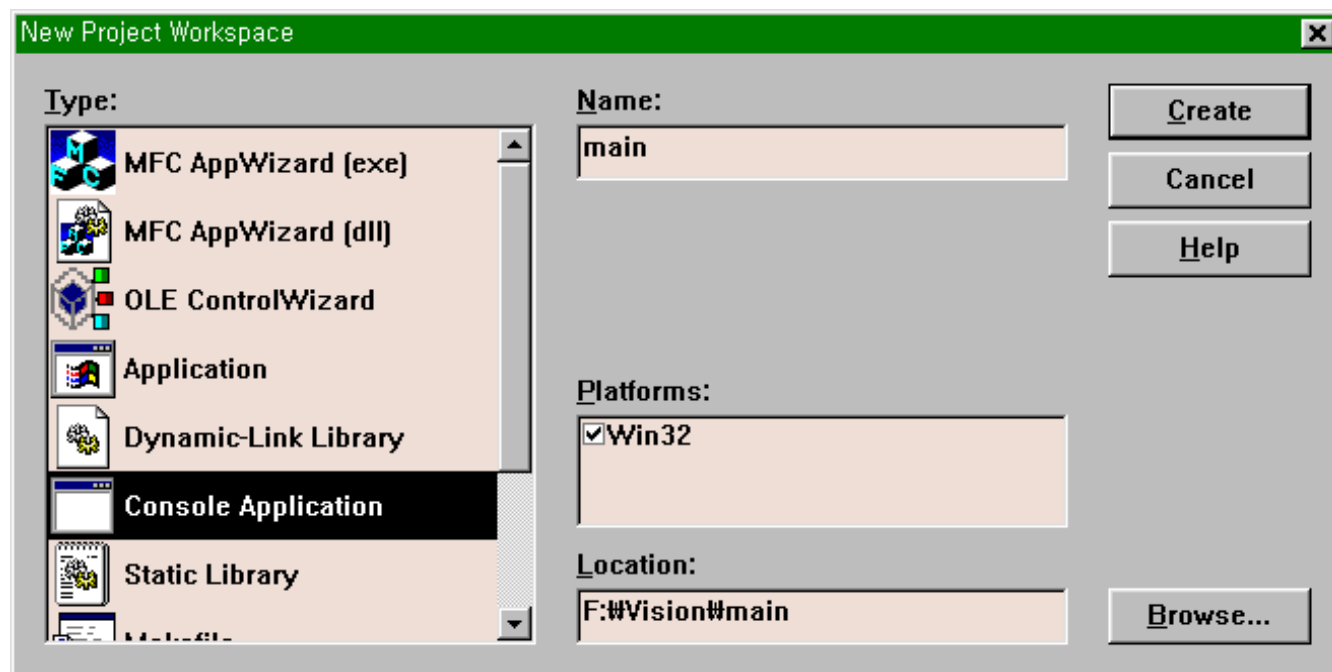
VC++ 4.2의 IDE 윈도우



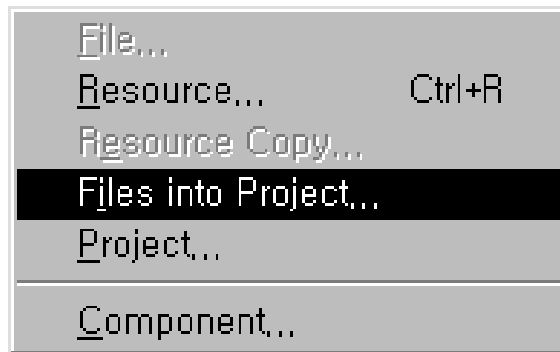
File->New의 선택



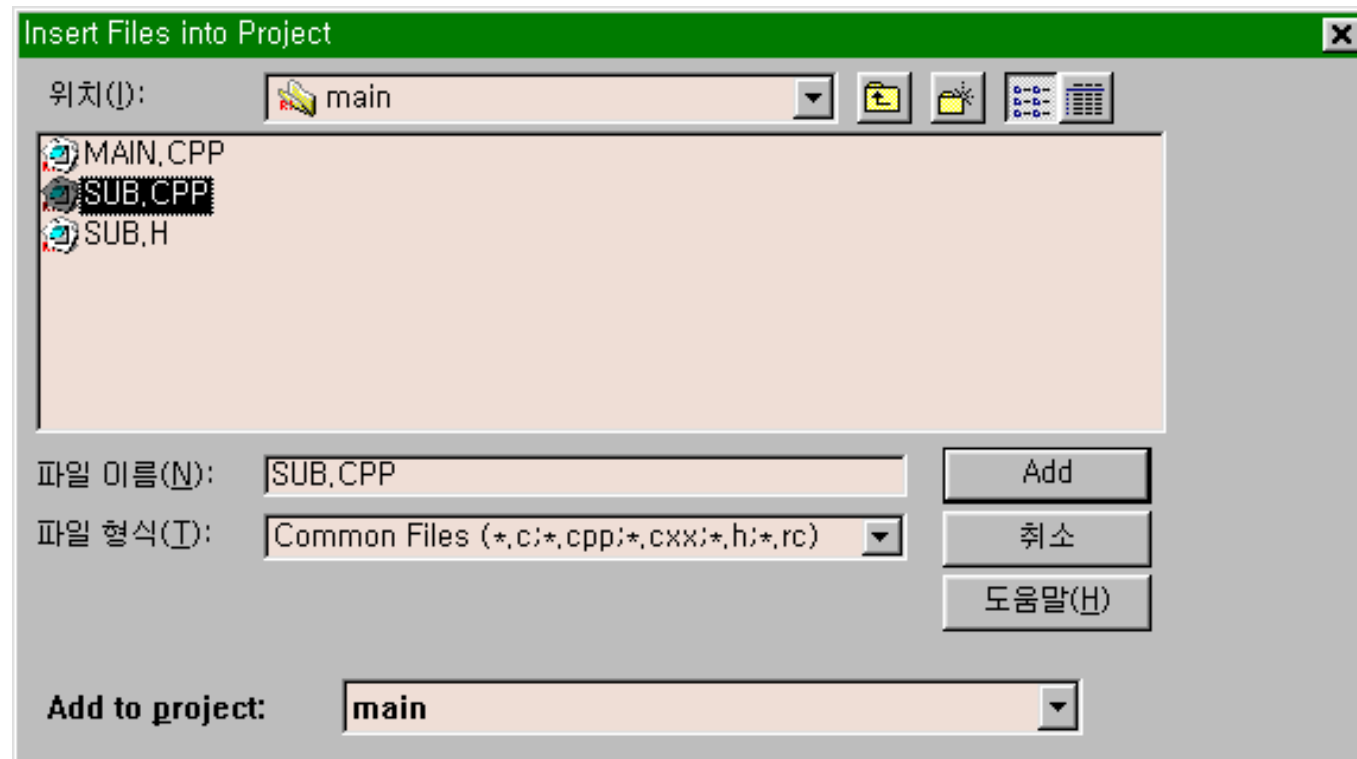
Project Workspace의 선택



Console Application의 선택



Insert->Files into Project의 선택



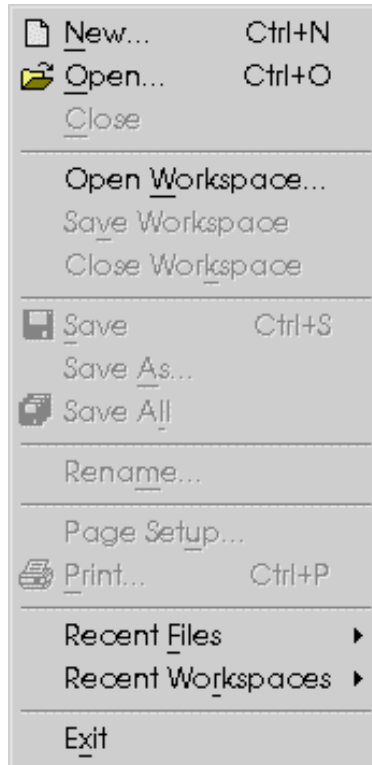
sub.cpp와 main.cpp의 Add



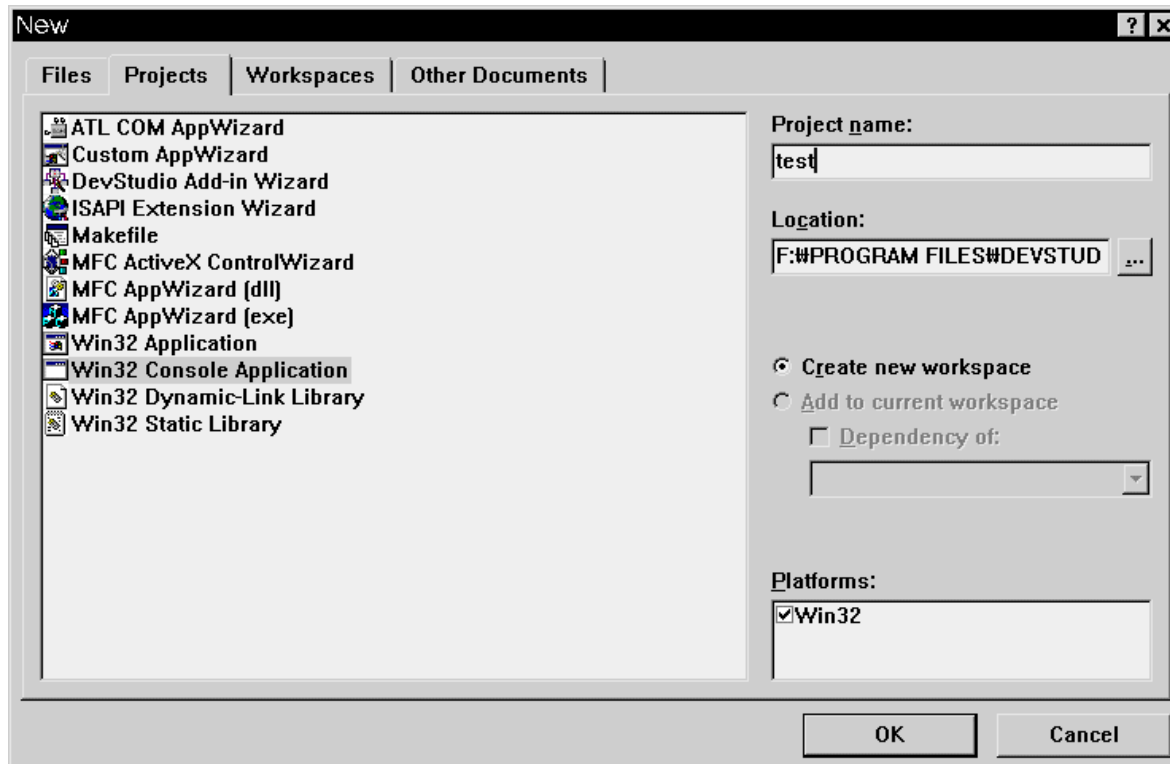
최종의 Project Workspace 윈도우



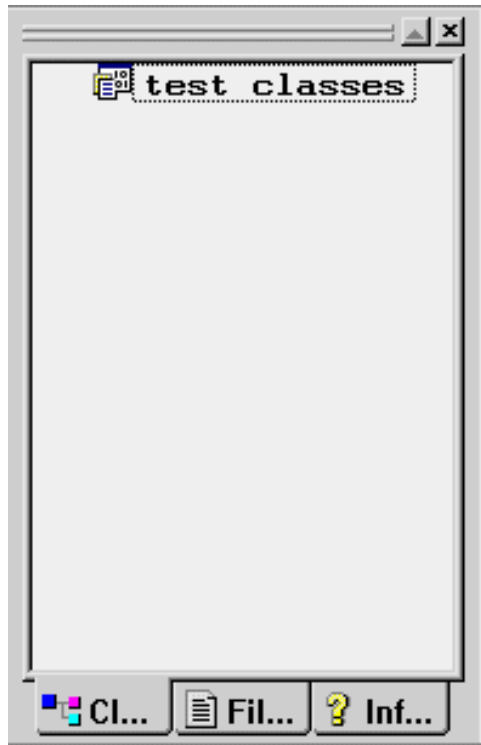
마이크로소프트 비주얼 C++ 5.0인 경우



File->New를 선택하면 프로젝트 타입을 결정할 수 있습니다.



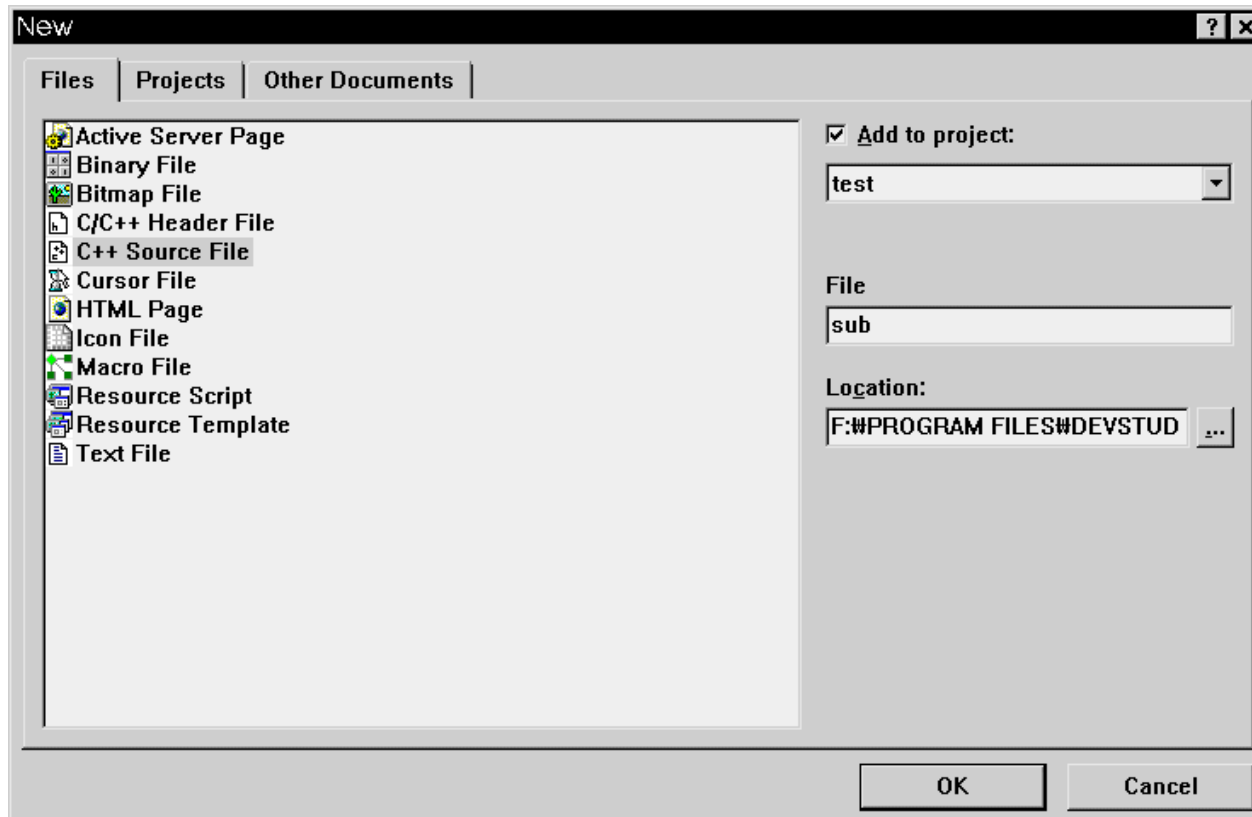
Project 탭에서 Win32 Console Application을 선택 하고, Project Name 글상자에 test를 입력합니다.



Workspace 윈도우의 File View탭에는 아직 어떠한 파일도 추가되지 않았습니다.



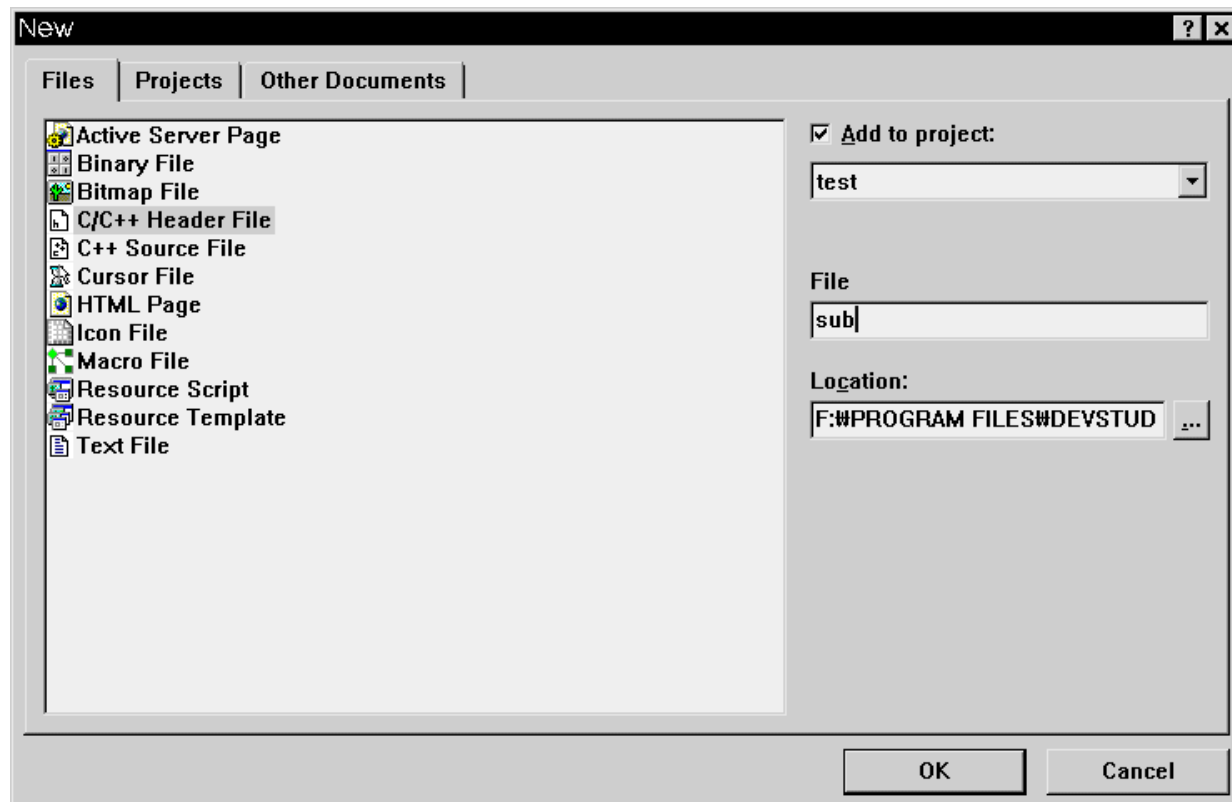
메뉴의 Project->Add To Project->New를 선택하면, 새로운 소스 파일을 만들 수 있습니다.



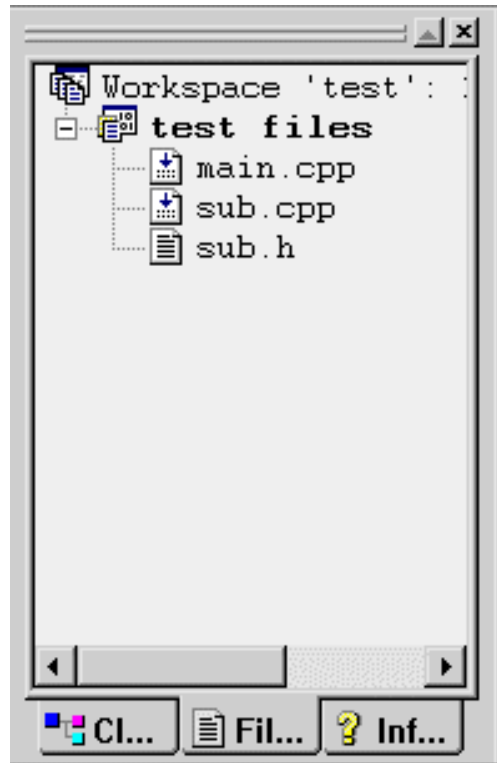
New 대화상자의 Files 탭에서 C++ Source File을 선택하고 File 글상자에 sub를 입력합니다.



다시 Project->Add To Project->New를 선택해 sub.h와 main.cpp를 모두 추가하여야 합니다. 지금의 예는 New를 선택해 파일을 모두 새로 만드는 경우이지만, 소스 파일이 이미 존재하는 경우는 Files를 선택해서 추가합니다.



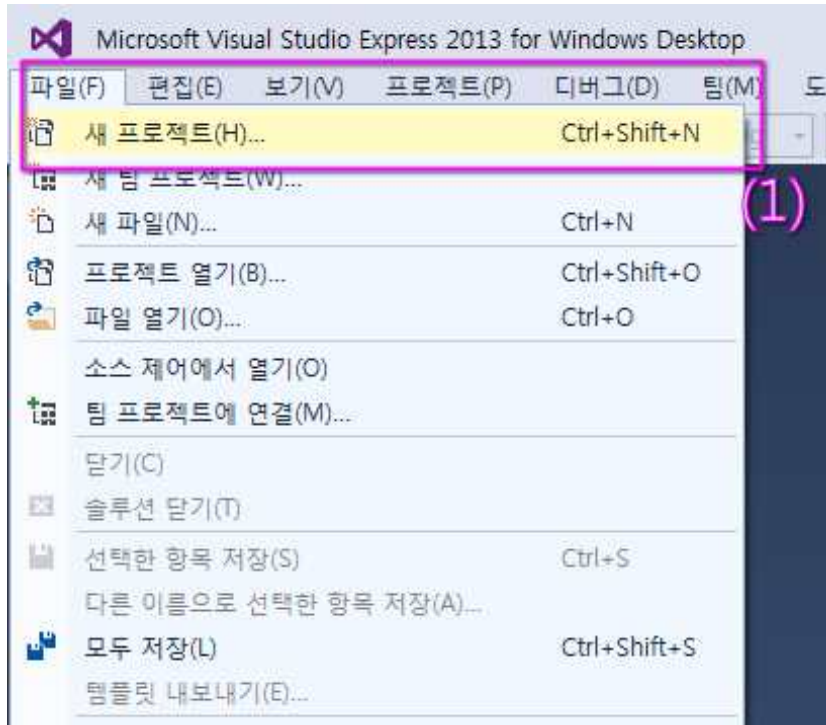
이번에는 C/C++ Header File을 선택하고 File 글상자에 sub.h를 입력합니다. 물론 확장자 .h는 생략 가능합니다.



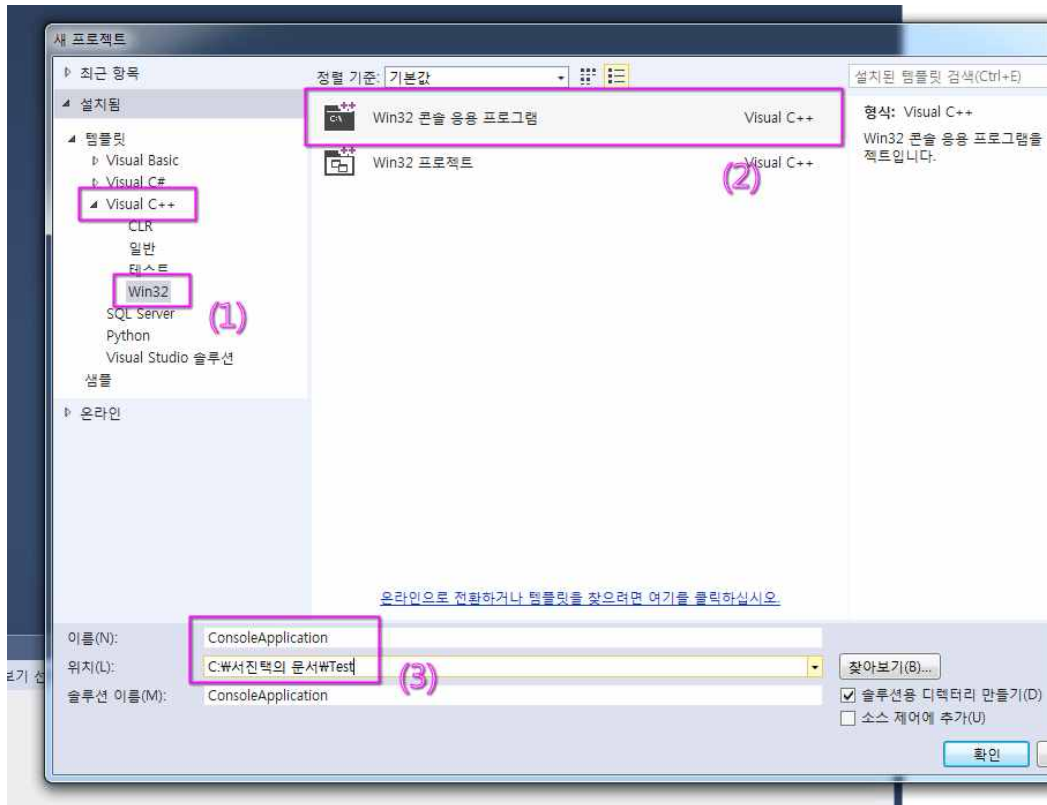
Workspace 윈도우에 이제 모든 파일 - main.cpp, sub.cpp - 이 추가되었습니다: 그림에는 sub.h가 보이지만 실제로 이 파일은 프로젝트에 포함하지 않아도 됩니다. 지금의 경우는 파일이 디스크에 없으므로 만들기 위해서 추가된 것입니다. 만들고 나서는 프로젝트에서 삭제합니다.



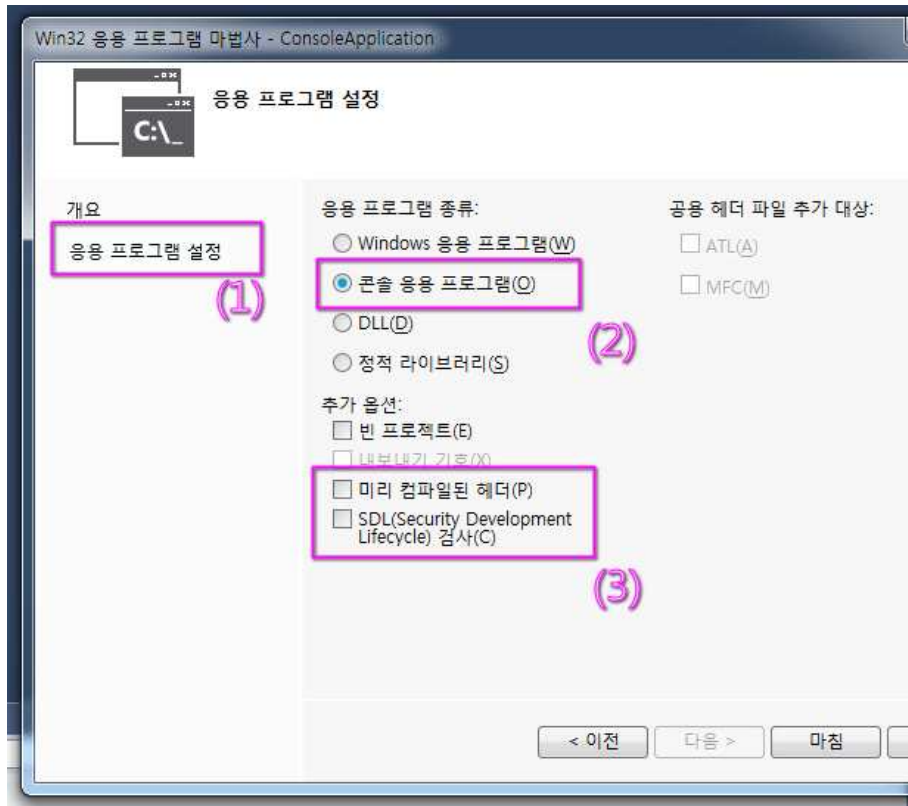
Visual Studio 2013의 경우



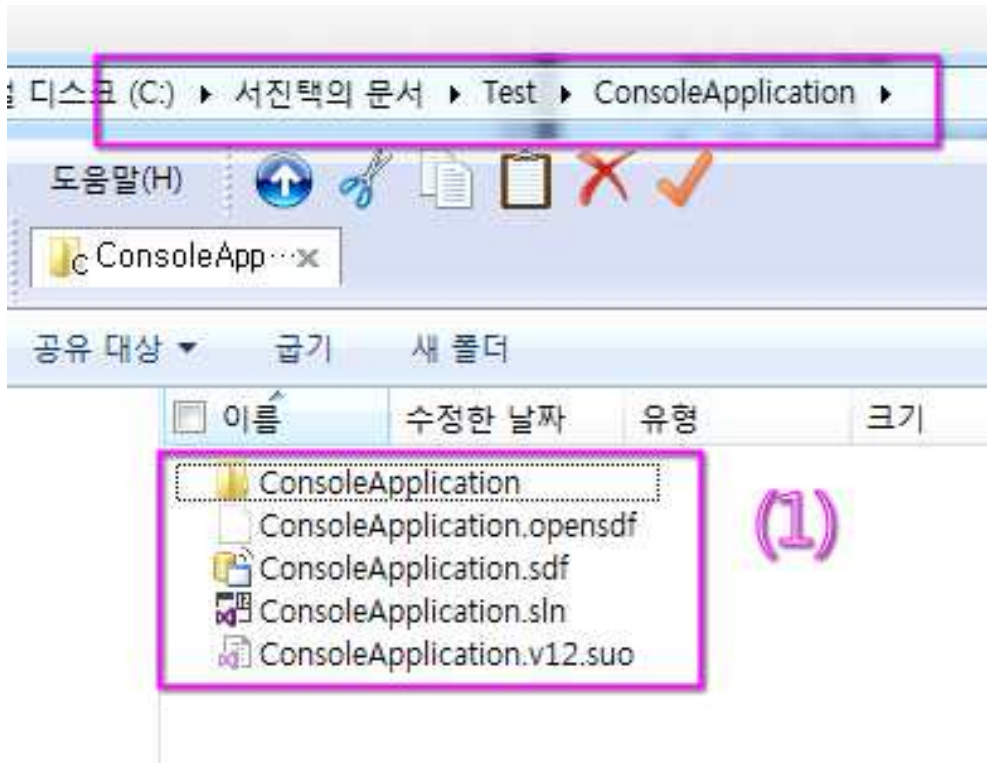
파일→새 프로젝트 메뉴를 선택합니다.



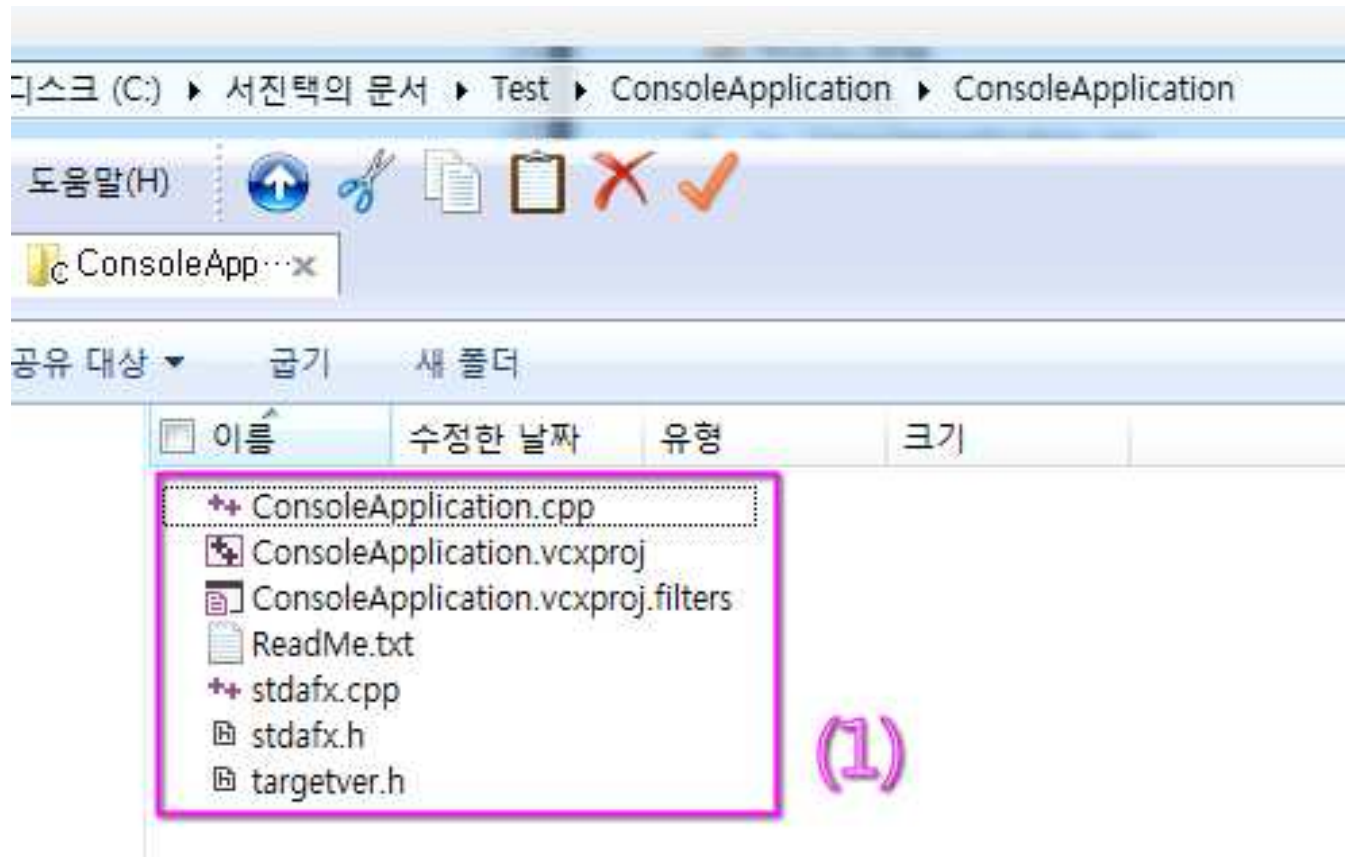
새 프로젝트 대화상자에서 범주를 (1) [Visual C++→Win32]를 선택합니다. (2) 프로젝트의 종류로 [Win32 콘솔 응용프로그램]을 선택합니다. (3) 적당한 폴더 위치를 지정하고, 프로젝트의 이름도 적당하게 지정합니다.



다음 단계에서 (1) [응용 프로그램 설정] 탭을 선택하고, (2) 응용 프로그램 종류로 [콘솔 응용 프로그램(O)]을 선택합니다. (3) 추가 옵션의 [미리 컴파일된 헤더]와 [SDL 검사]는 체크해제 합니다.



(1) 대상 폴더로 이용하여 생성된 파일file을 확인합니다. Visual Studio가 프로젝트를 유지하는데 필요한 파일들을 생성한 것을 확인할 수 있습니다.



ConsoleApplication폴더로 이동하면, C++소스 파일인 ConsoleApplication.cpp를 확인할 수 있습니다. 이 파일에 책의 소스를 입력해서 테스트합니다.

- ConsoleApplication.cpp에 자동으로 생성된 소스 코드의 내용은 아래와 같습니다.

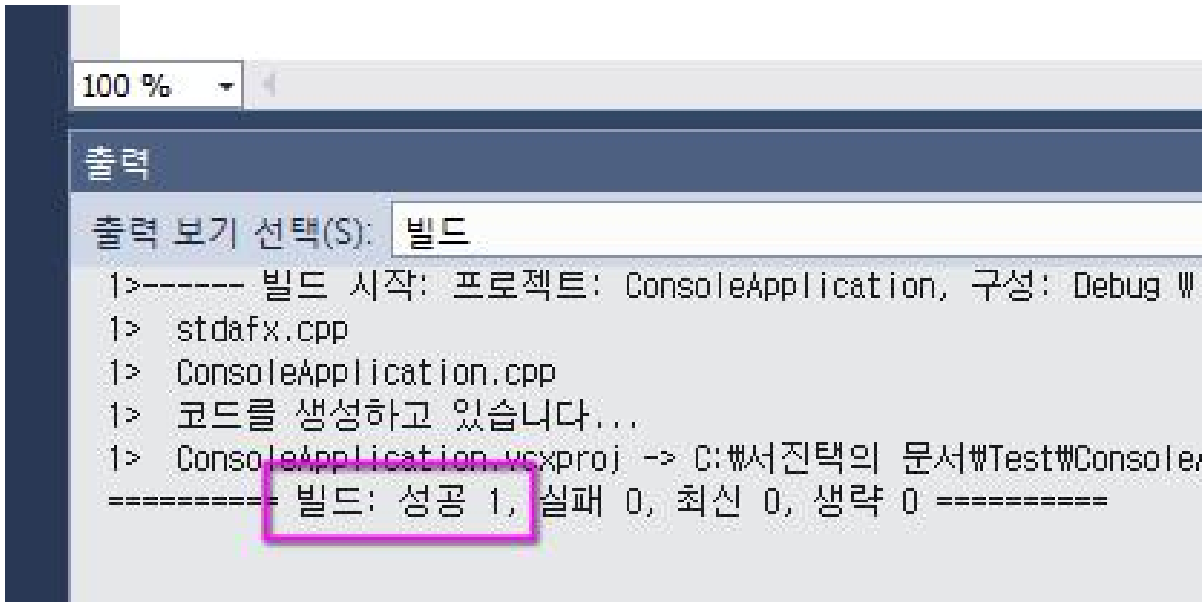
```
#include "stdafx.h"
```

```
int _tmain(int argc, _TCHAR* argv[])  
// int main(void) // 이렇게 작성해도 됩니다.  
{  
    return 0;  
}
```

- 이제 [빌드]메뉴의 [솔루션 빌드(B)]를 선택합니다. 그러면, Visual Studio는 프로젝트에 있는 소스 파일들을 컴파일compile하고 링크link한 후에, 실행 파일을 생성합니다.



[빌드→솔루션 빌드]를 선택합니다.



The screenshot shows the 'Output' window in Visual Studio. The 'Show Output From:' dropdown is set to 'Build'. The output text shows the build process for a project named 'ConsoleApplication'. The final line of the output, '==== 빌드: 성공 1, 실패 0, 최신 0, 생략 0 =====', is highlighted with a pink rectangular box.

```
100 %  
출력  
출력 보기 선택(S): 빌드  
1>----- 빌드 시작: 프로젝트: ConsoleApplication, 구성: Debug W  
1> stdafx.cpp  
1> ConsoleApplication.cpp  
1> 코드를 생성하고 있습니다...  
1> ConsoleApplication.vcxproj -> C:\서진택의 문서\Test\ConsoleA  
==== 빌드: 성공 1, 실패 0, 최신 0, 생략 0 =====
```



출력창에 빌드 성공 메시지가 출력됩니다.



실습문제

1. Android Studio에서 프로젝트를 만드는 과정을 기술 하세요.

2. Unity 4.x/5.x 버전에서 프로젝트를 만드는 과정을 기술 하세요.

3. Visual Studio 2017에서 C++ 프로젝트를 만드는 과정을 기술 하세요.