

C Programming

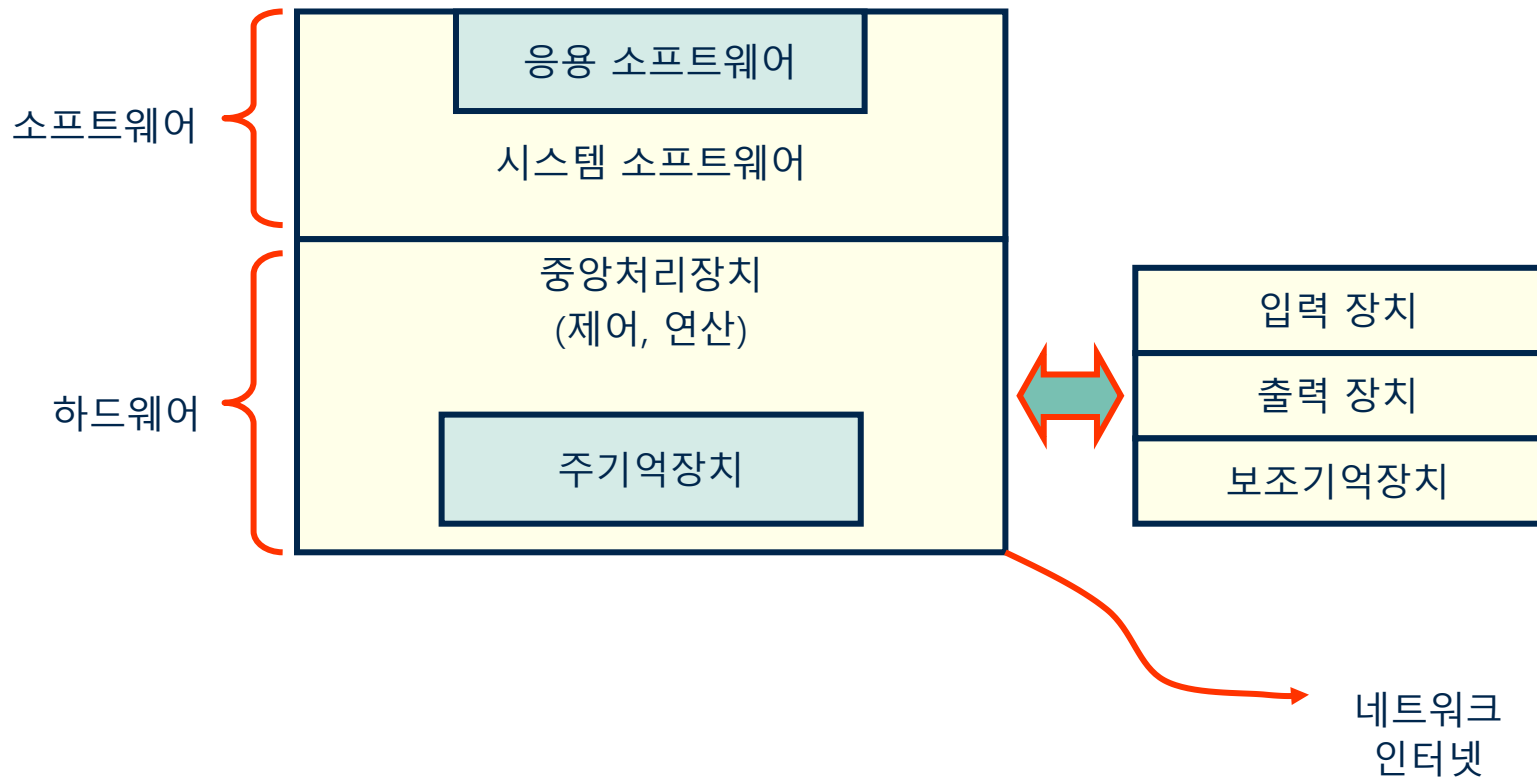
Ch.1-1 C 언어 개요 및 컴파일러 사용 방법

Contents

1. 프로그램 동작 방식 (추가)
2. C 언어 개론
3. 콘솔 Vs. Windows 프로그래밍 (추가)
4. 프로그래밍 작성 과정
5. Visual C++ 디버깅 (옵션)

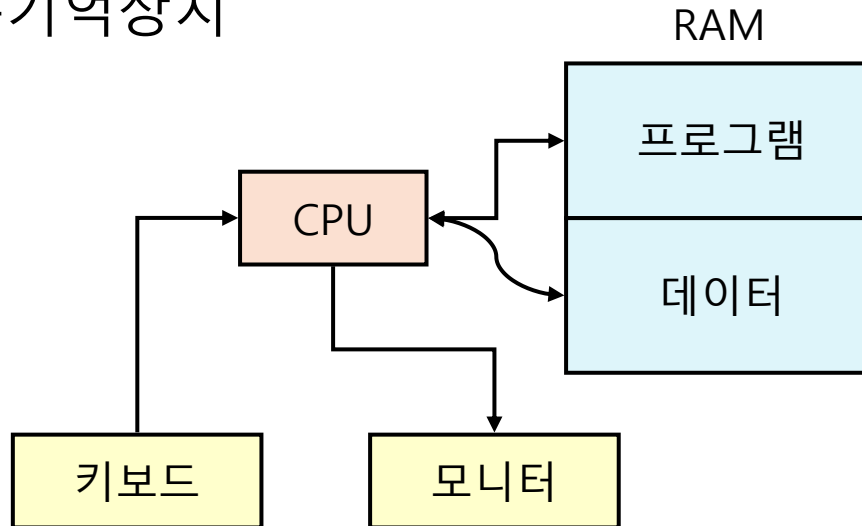
- 추가 : 교재 외 추가 자료
- 옵션 : skip 가능한 참고 자료

컴퓨터의 간략한 구조



프로그램 내장 방식

- ▶ 프로그램 실행 시 프로그램이 저장되는 곳
 - 주기억장치



- ▶ 프로그램 동작 : 명령 → CPU → 실행(키보드, RAM, 모니터)
- ▶ 같은 작업을 하는 프로그램이라 하더라도
 - 얼마나 메모리를 적게 사용하느냐!
 - 얼마나 빨리 수행되느냐!



자료 구조
알고리즘

프로그래밍 언어의 역사 (2)

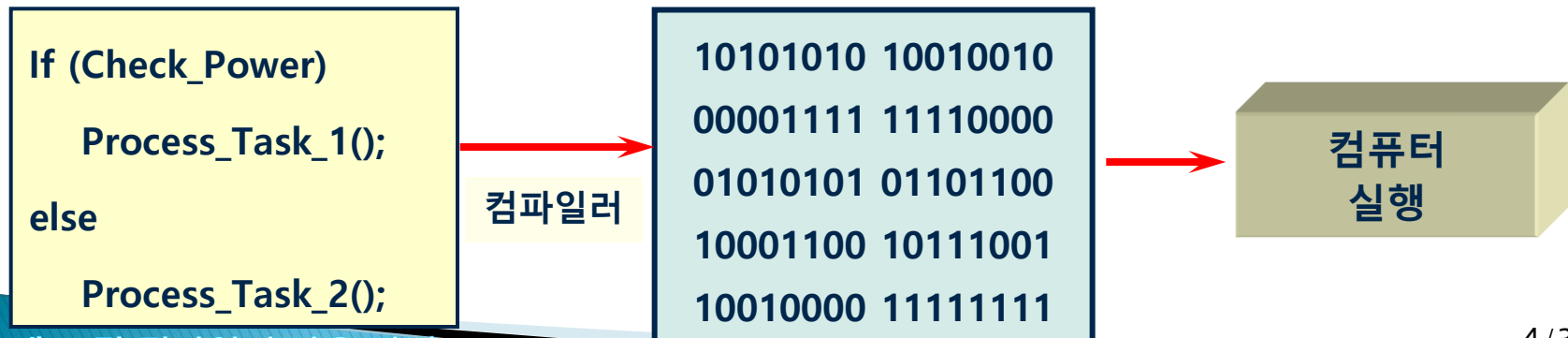
▶ 프로그래밍 언어란 무엇인가?

- 사람과 컴파일러가 이해할 수 있는 약속된 형태의 언어
 - C, C++, Python, Java, ...
 - 해당 언어를 기계어로 번역하는 번역기 필요



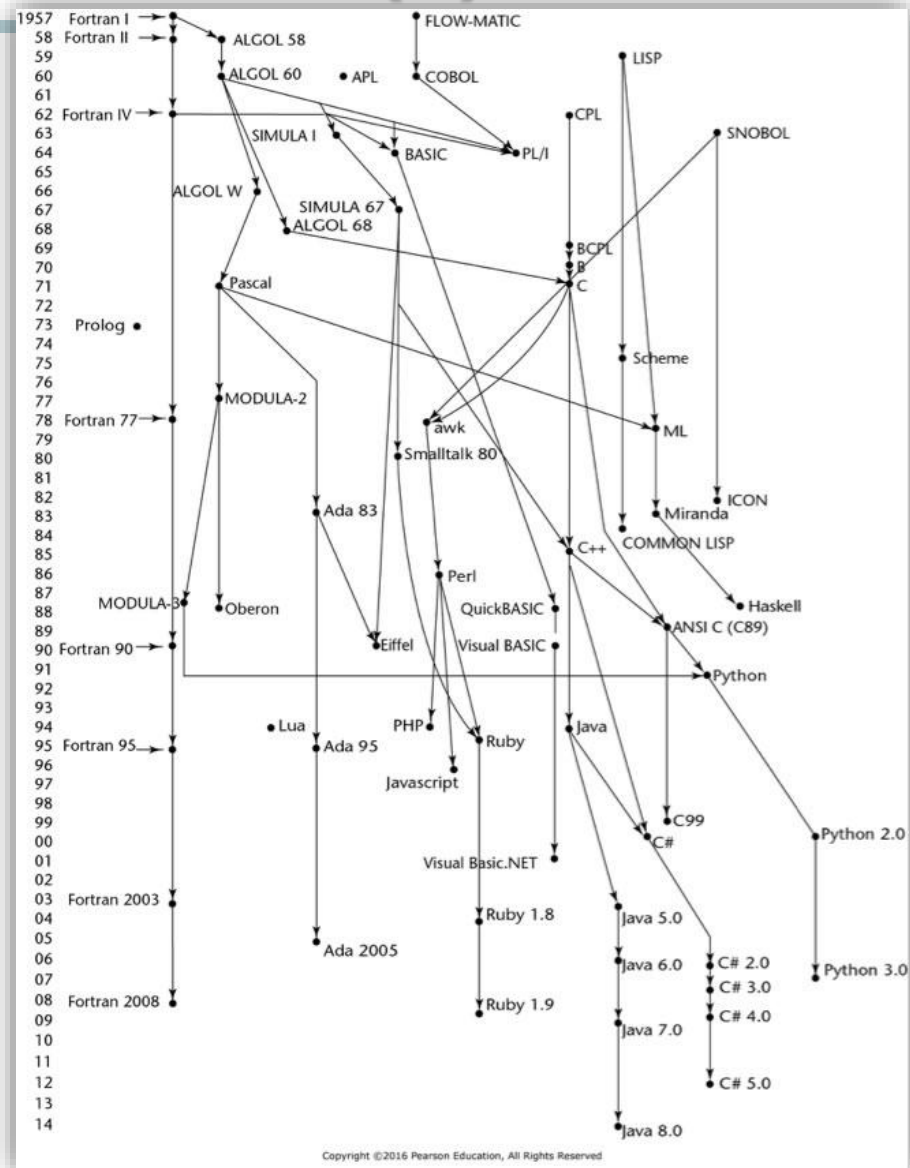
▶ 번역 방식

- 컴파일러 : 프로그램 전체를 기계어로 번역 후 실행
- 인터프리터 : 매 문장 하나씩 기계어로 번역하고 실행



프로그래밍 언어의 역사 (2)

▶ 주요 프로그래밍 언어의 역사



C 언어는 프로그래밍 언어이다.

- ▶ C 언어를 공부한다는 것은?
 - 문법을 이해하는 것
 - 표현 능력을 향상시키는 것
- ▶ C 언어에 익숙해지려면?
 - 많이 사용할수록 표현에 능숙해진다
 - 다른 이의 표현을 참조할수록 표현이 부드러워진다

C 언어의 역사와 특징

- ▶ C언어의 역사
 - 1971 년경 UNIX 운영체제의 개발을 위해 Dennis Ritchie와 Ken Thompson이 함께 설계한 범용적인 고급(high-level)언어
 - 근원: ALGOL 60(1960), CPL(1963), BCPL(1969), B언어(1970)
- ▶ C언어 등장 이전의 유닉스 개발
 - 어셈블리(assembly) 언어라는 저급(low-level)언어로 제작
 - 어셈블리어 : 하드웨어에 따라 그 구성이 달라지기 때문에 CPU 별로 유닉스를 각각 개발해야 됨
- ▶ C언어 등장 이후 유닉스 개발
 - C언어의 구성은 CPU에 따라 나뉘지 않기 때문에 CPU 별로 유닉스를 각각 개발할 필요가 없음 → 표준 C로의 발전 (현재 표준 : C11)
- ▶ 고급 언어? 저급 언어?
 - 고급 언어 : 사람이 이해하기 쉬운 언어
 - 저급 언어 : 기계어에 가까운 언어. 기계어, 어셈블리어
 - C언어는 고급 언어이면서 메모리에 직접 접근이 가능하기 때문에 저급 언어의 특성도 함께 지님

C 언어의 장점

- ▶ C언어는 절차지향적 특성을 지닌다.
 - 인간의 사고하는 방식과 유사하다.
 - 따라서 쉽게 익숙해질 수 있다.
- ▶ C언어로 작성된 프로그램은 이식성이 좋다.
 - CPU에 따라 프로그램을 재작성할 필요가 없다.
 - 그러나 근래에는 C언어보다 이식성이 훨씬 뛰어난 언어들이 등장하고 있어서 장점으로 부각시키기에는 한계가 있다.
- ▶ C언어로 구현된 프로그램은 좋은 성능을 보인다.
 - C언어를 이용하면 메모리의 사용량을 줄일 수 있고, 속도를 저하시키는 요소들을 최소화 할 수 있다.
 - 단, 잘못 구현하면 오히려 성능이 좋지 못한 프로그램이 만들어지기도 한다.

→ C언어의 장점은 앞으로 C언어를 공부해 나가면서 보다 정확히 이해하게 된다.

콘솔 vs. Windows 프로그래밍

- ▶ 콘솔 프로그래밍 (= DOS 프로그래밍)
 - 텍스트 중심의 프로그래밍
 - Windows 운영체제에서 도스 창 실행 방법
 - [시작]-[실행]-cmd
 - 표준 C → 콘솔 프로그래밍
 - 본 강의의 주제

```

관리자: C:\Windows\system32\cmd.exe
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation. All rights reserved.

C:\Users\Wjhhuang>dir /w
C 드라이브의 볼륨: Program
볼륨 일련 번호: 3014-68ED

C:\Users\Wjhhuang 디렉터리

[.]                [..]                [.designer]
[.eclipse]         [.idlerc]             [.keras]
[.kivy]            [.matplotlib]   [.oracle_jre_usage]
[.p2]              [.PyCharm50]    [.PyCharmCE2016.3]
[.rurple]          [.swt]           [.VirtualBox]
[chdemons]         ChIDE.session   ChIDEUser.abbrev
ChIDEUser.properties [Contacts]             [Desktop]
dlmgr_.pro         [Documents]   [Downloads]
[Favorites]        [Intel]       [Links]
[Music]            [Pictures]    [Saved Games]
[Searches]         Sti_Trace.log [Videos]
[VirtualBox UMs]   [USWebCache]  [wekafiles]
[workspace]        .chrc

6개 파일 18,292 바이트
32개 디렉터리 16,229,179,392 바이트 남음

C:\Users\Wjhhuang>
  
```

- ▶ Windows 프로그래밍
 - GUI(Graphical User Interface) 중심의 프로그래밍
 - Windows API, MFC 또는 다른 GUI 라이브러리 사용

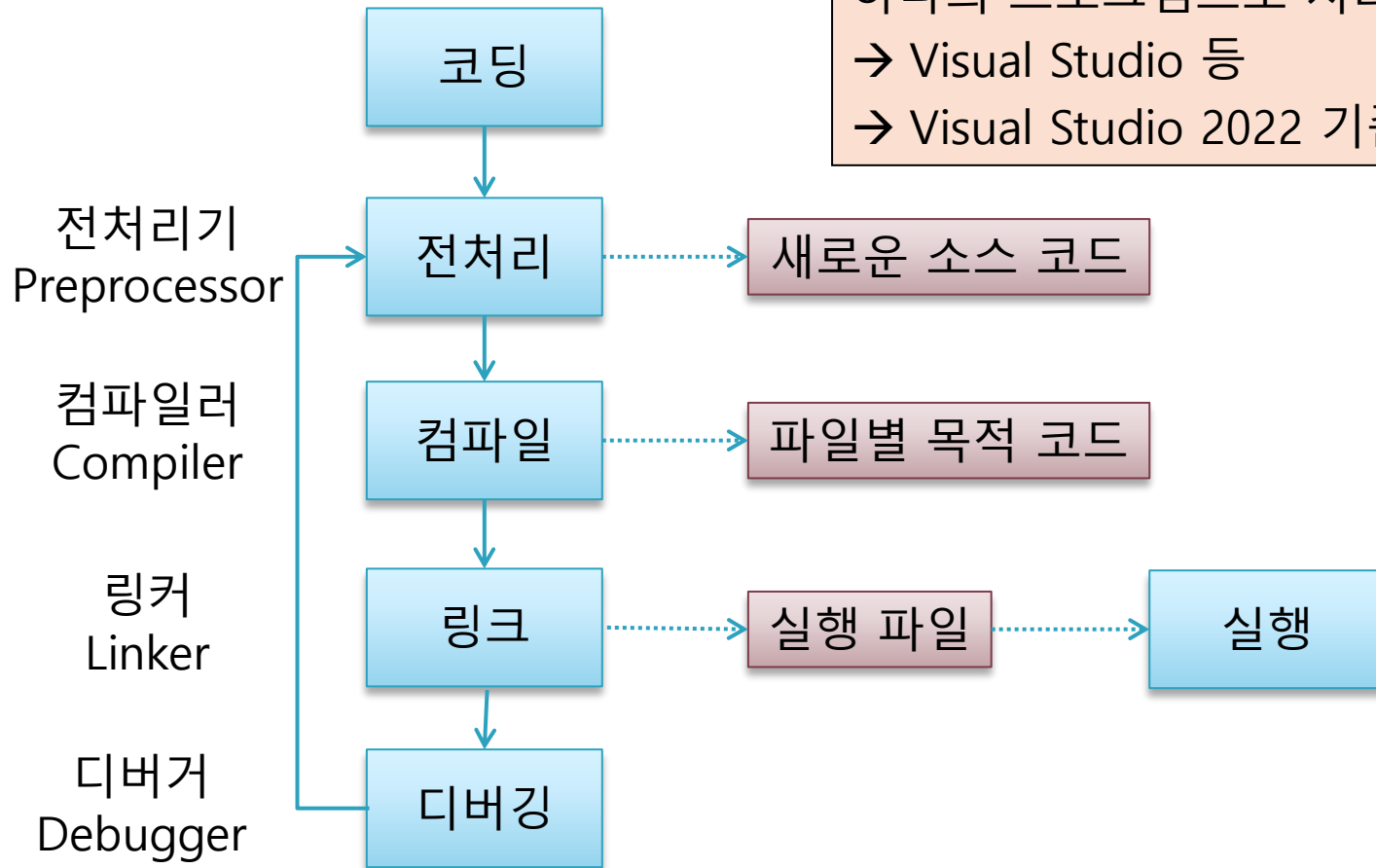
본 강의 내용

- ▶ 표준 C
 - 콘솔 프로그래밍 (Not GUI)
- ▶ 수업 시간 활용 컴파일러
 - Visual Studio 2022 (Visual C++ 17.0)
 - 다른 컴파일러와 개발환경을 사용해도 됨! VSCode, gcc/MinGW 등
- ▶ 참고 사항
 - 프로그래밍에 숙달되려면 - trial and error
 - 모든 걸 다 외울 수는 없다 - 항상 MSDN 도움말 참고 (F1 key)

프로그램 개발 단계

- ▶ 프로그램
 - 컴퓨터가 특정 작업 또는 문제를 해결하도록 나열한 명령어들의 집합
- ▶ 프로그램 개발 단계
 1. 문제 분석 : 수행하고자 하는 작업을 명확히 기술
 - 관련 과목 : 소프트웨어공학
 2. 알고리즘 개발 : 그 작업을 해결할 수 있는 방법 연구
 - 관련 과목 : 데이터구조 및 알고리즘
 3. 코딩 : C 언어로 알고리즘을 기술
 - 본 강좌의 목표 : C 언어의 문법 이해 및 활용
 4. 수행 및 디버깅 : 완성된 프로그램을 테스트하고 수정
 - 본 강좌의 목표
 5. 유지 보수 : 문제의 요구조건 변화 또는 문제점 발견에 따른 수정
 - 관련 과목 : 소프트웨어공학

프로그램 작성 단계

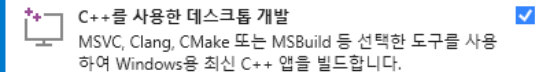


통합개발환경(IDE) : 모든 과정을 하나의 프로그램으로 처리 가능
 → Visual Studio 등
 → Visual Studio 2022 기준 설명

Visual Studio 2022 사용 방법

▶ Visual Studio 2022 Community 버전 (무료)

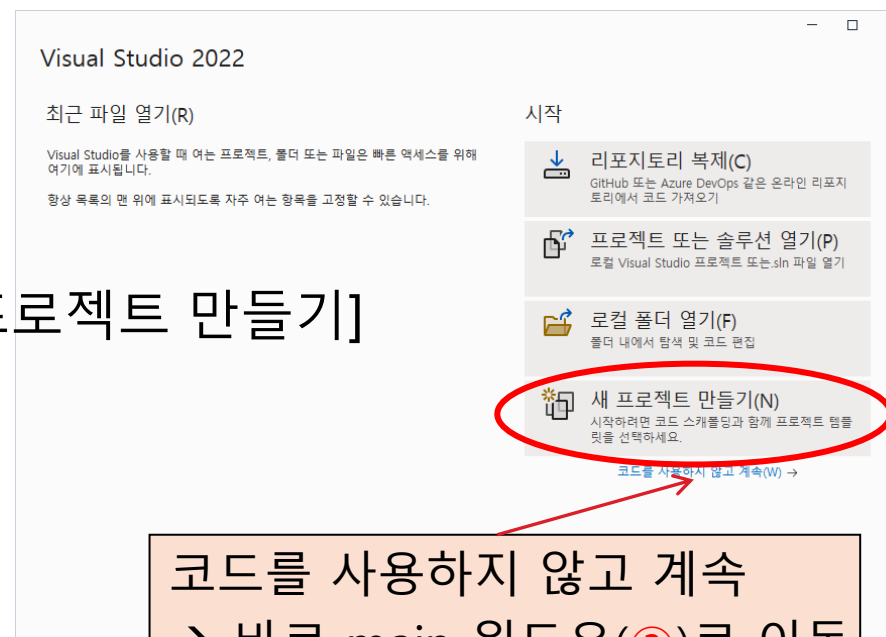
- 설치 시 “C++를 사용한 데스크톱 개발” 선택



▶ Visual Studio 2022 실행

- [시작] → [Visual Studio 2022]
- Visual Studio를 처음으로 실행하는 경우

- 로그인 필요 없음
- 개발 설정 : Visual C++
- 색 테마 : 적절히
- [Visual Studio 시작] → [새 프로젝트 만들기]

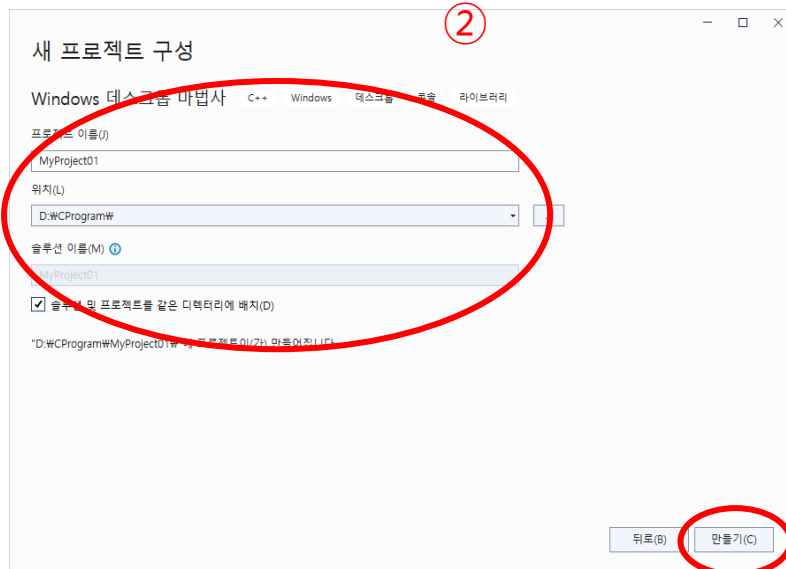


코드를 사용하지 않고 계속
→ 바로 main 윈도우(③)로 이동

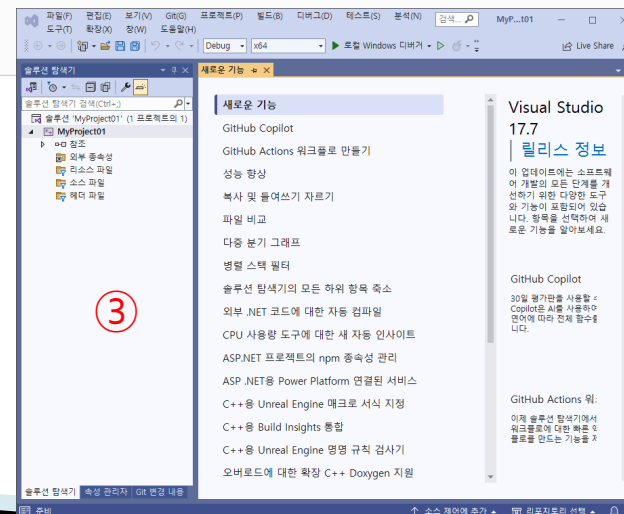
새 프로젝트 만들기 (1)

▶ [빈 프로젝트] 선택

- 프로젝트 이름
- 위치 설정
- 솔루션 및 프로젝트를 같은 디렉터리에 배치

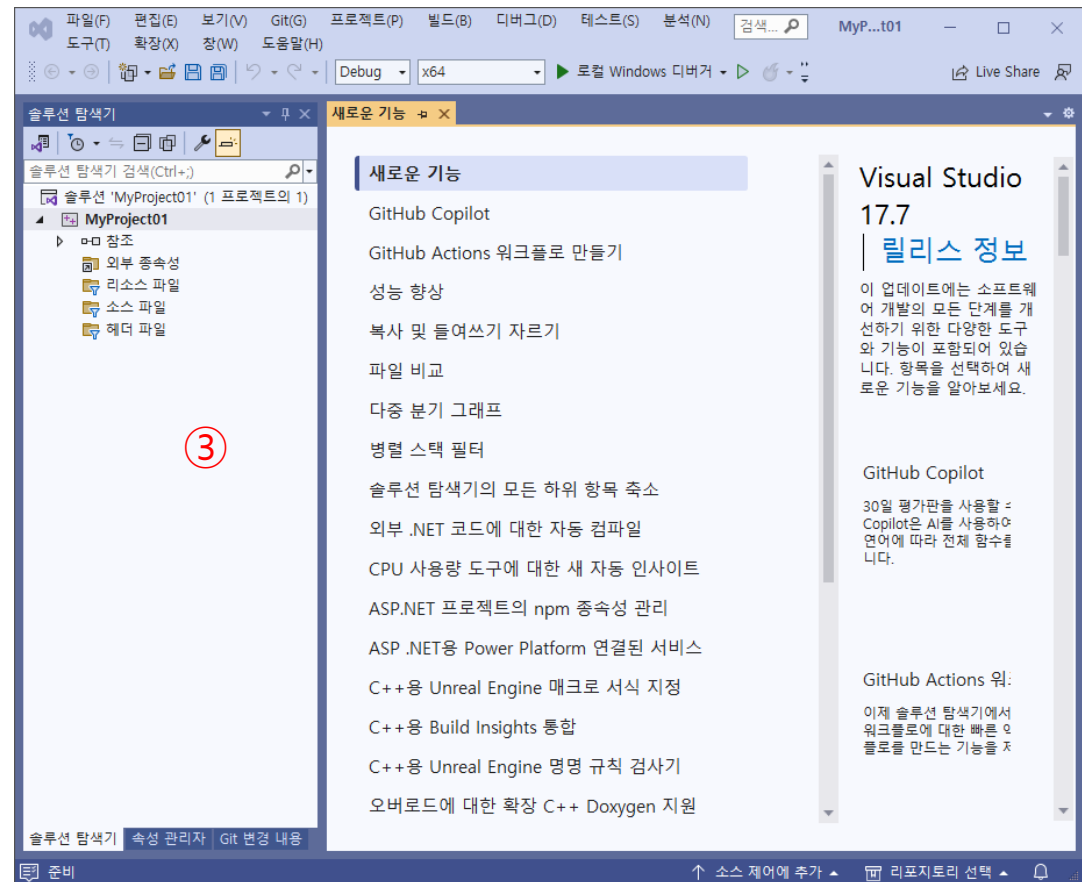


◦ [만들기] 클릭



새 프로젝트 만들기 (2)

- ▶ ③에서 새 프로젝트 만들기
 - [파일]-[새로 만들기]-[프로젝트...]
 - ①부터 다시 시작



프로젝트 만들기 (3)

▶ 용어 설명

◦ 솔루션

- 하나의 소프트웨어 ← 여러 개의 프로그램 포함 가능

◦ 프로젝트

- 하나의 프로그램

▶ 새 프로젝트 작성 시

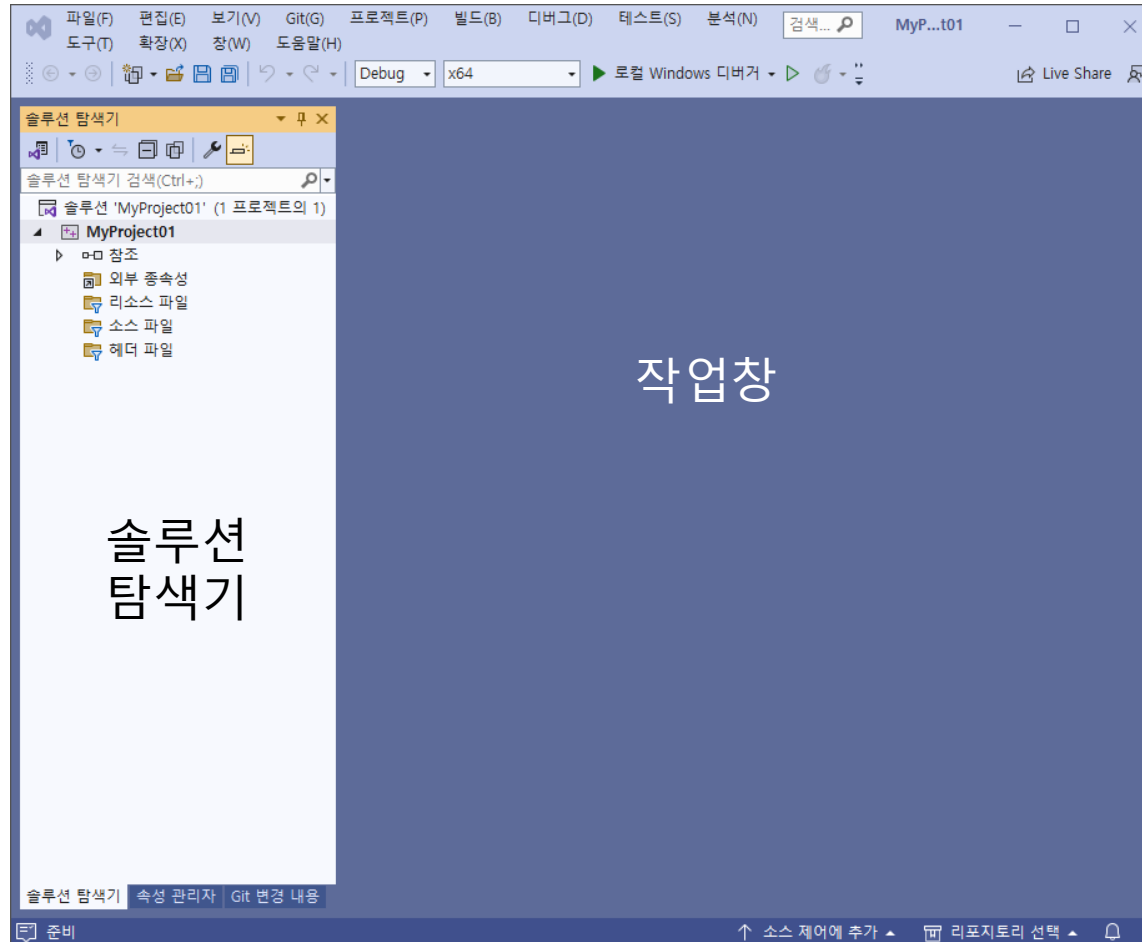
◦ 솔루션 및 프로젝트를 같은 디렉터리에 배치 선택

- 솔루션 폴더 밑에 프로젝트가 함께 생성됨

◦ 선택하지 않을 경우 : 별도의 프로젝트 폴더 생김

프로젝트 만들기 (4)

▶ 프로젝트 생성 완료



솔루션
탐색기

작업창

코딩 (1)

▶ 소스 파일 추가

◦ [프로젝트] → [새 항목 추가...]

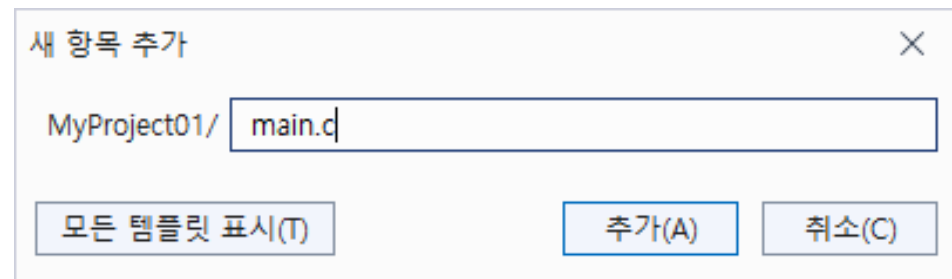
- 또는 솔루션 탐색기의 프로젝트명에서 팝업 메뉴 [추가]-[새 항목]

◦ 파일명 입력



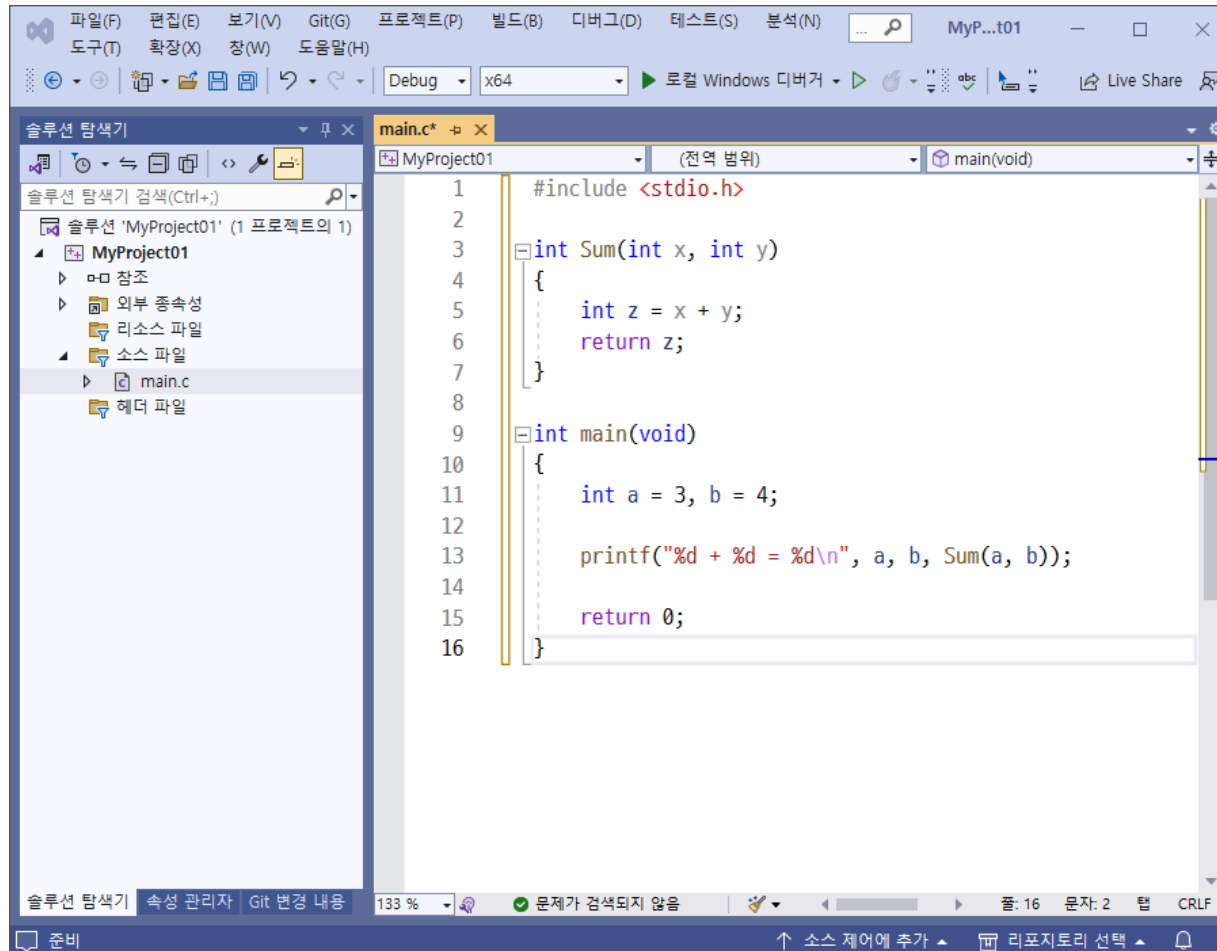
C 언어 : 반드시 확장자로
c 입력 → 예 : main.c

확장자가 cpp인 경우
C++ 프로그램이 됨



코딩 (2)

▶ 소스 코드 입력



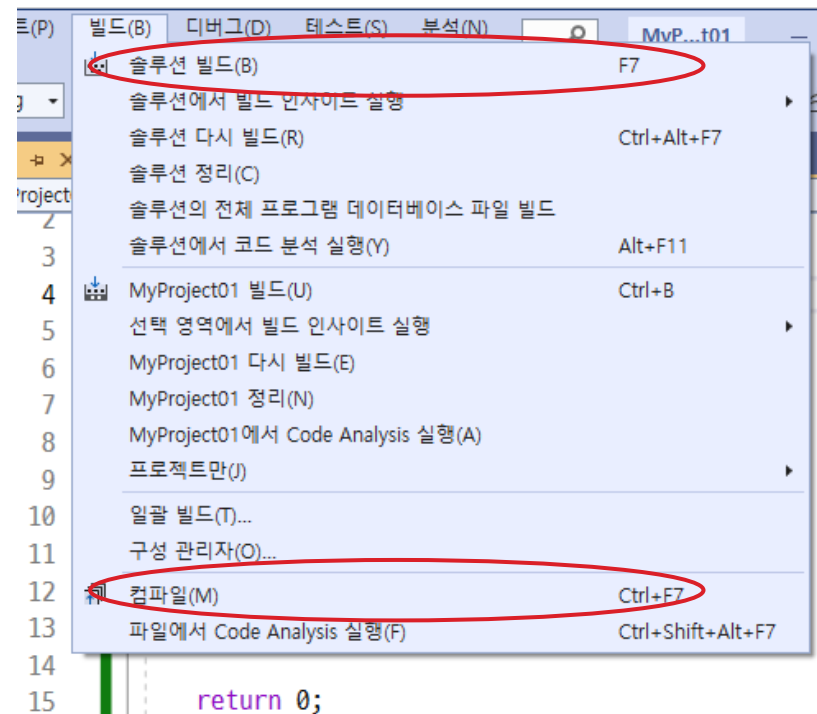
컴파일, 링크 (1)

▶ 컴파일

- [빌드] → [컴파일] 또는 단축키 Ctrl+F7

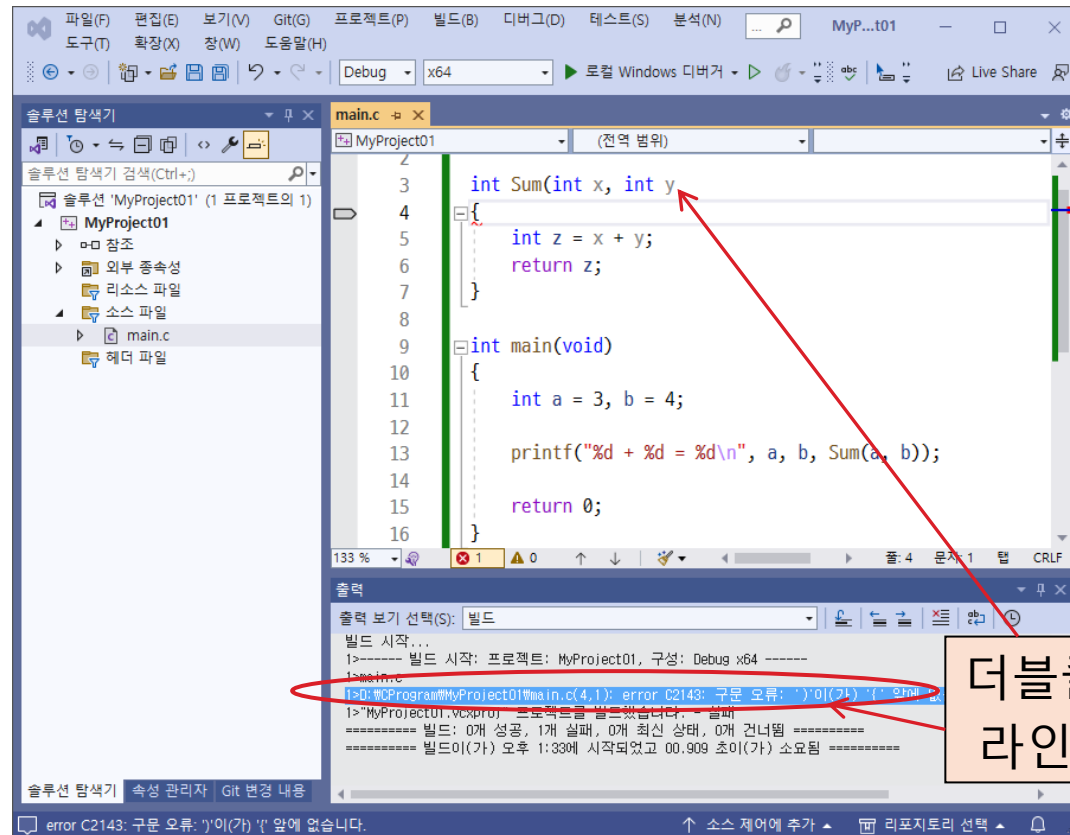
▶ 링크

- [빌드] → [솔루션 빌드] 또는 F7
- 링크를 수행하면 자동으로 컴파일 실행됨



컴파일, 링크 (2)

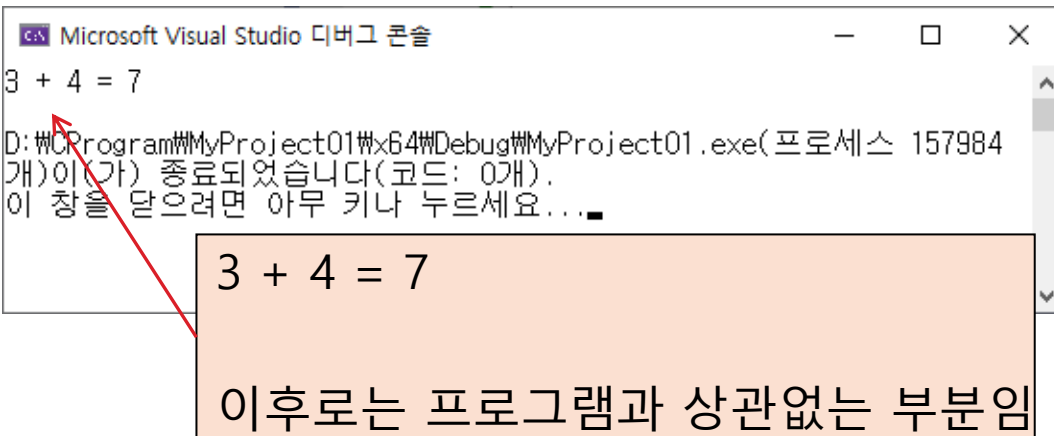
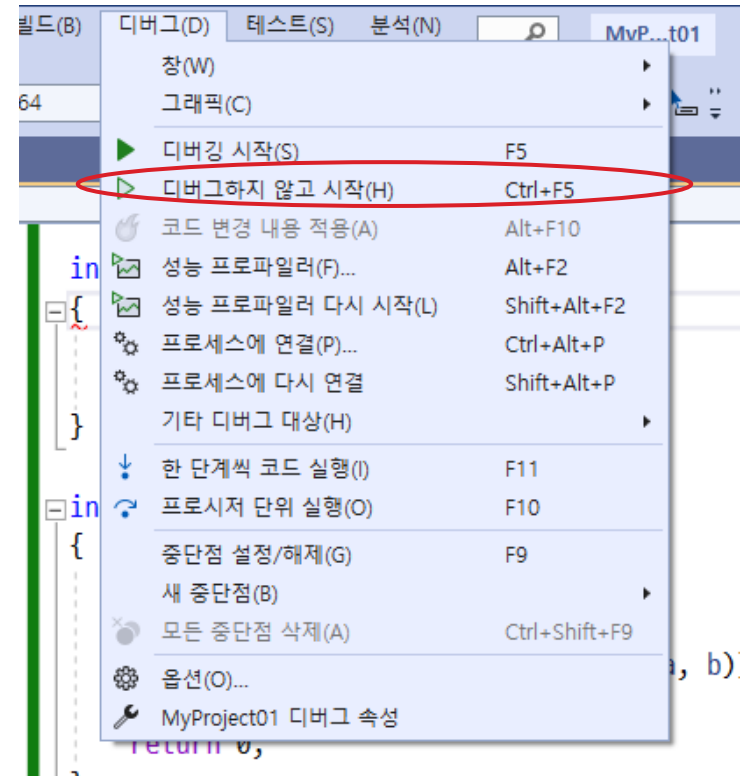
▶ 오류가 발생한 경우



- error(오류) : 프로그램 수행 불가능 상태 → 컴파일 실패
- warning(경고) : 프로그램 수행은 가능하나 논리적 오류 존재 가능

실행 (1)

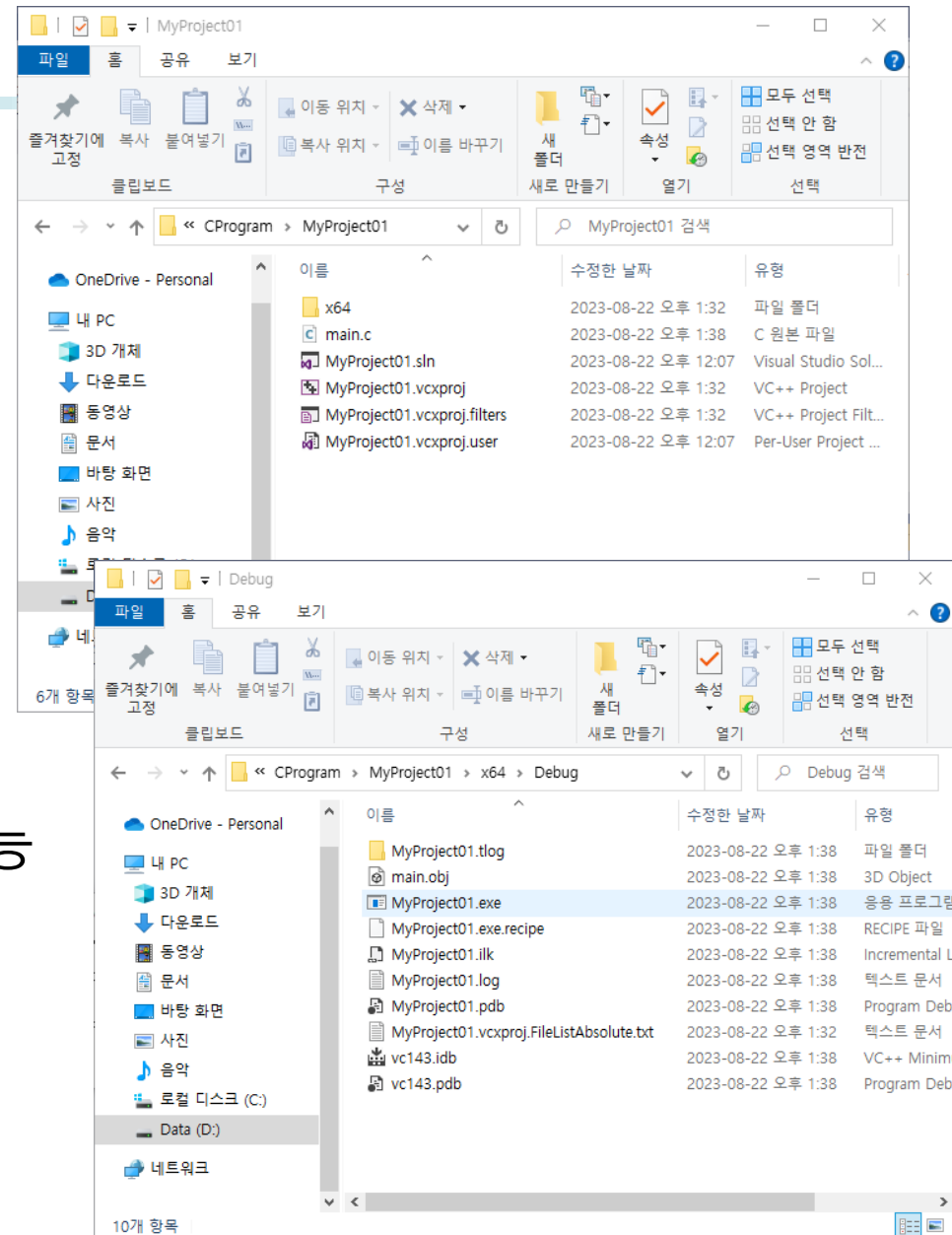
- ▶ 실행 방법 (1) : Visual Studio 내에서 실행
 - [디버그]→[디버그하지 않고 시작]
 - 단축키 : Ctrl+F5
- ▶ 실행 결과 : 콘솔창



* 콘솔창이 실행 후 바로 닫히는 경우
 메뉴 [프로젝트]-[... 속성]-[구성 속성]-[링커]-[시스템]-[하위 시스템] : 콘솔 선택

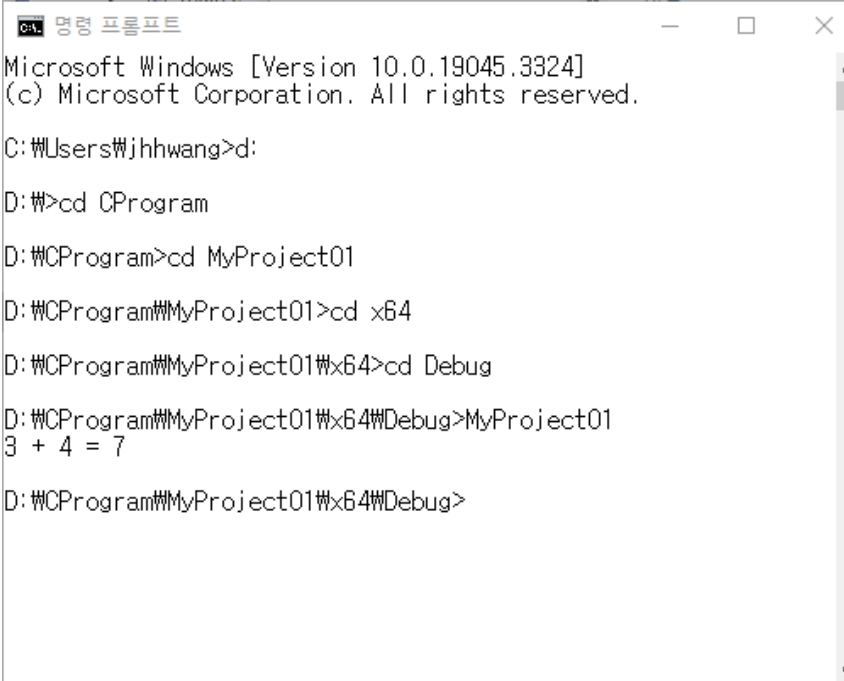
실행 (2)

- ▶ 프로젝트 폴더 구성
 - .sln : 솔루션 관리 파일
 - .vcxproj : 프로젝트 관리 파일
 - .c : 소스 파일
 - x64\Debug 폴더
 - .exe : 실행 파일
 - 탐색기에서 바로 실행 가능
 - 실행 후 바로 종료



실행 (3)

- ▶ 실행 방법 (2) : 도스창에서 실행
 - 도스창 실행 : [시작] → cmd 명령어 실행
 - .exe 파일이 있는 곳으로 이동 (cd 명령어)
 - 파일명 입력 후 엔터(실행)



```
cmd. 명령 프롬프트
Microsoft Windows [Version 10.0.19045.3324]
(c) Microsoft Corporation. All rights reserved.

C:\Users\Wjhwwang>d:

D:\>cd CProgram

D:\CProgram>cd MyProject01

D:\CProgram\MyProject01>cd x64

D:\CProgram\MyProject01\x64>cd Debug

D:\CProgram\MyProject01\x64\Debug>MyProject01
3 + 4 = 7

D:\CProgram\MyProject01\x64\Debug>
```

디버그 모드와 릴리즈 모드

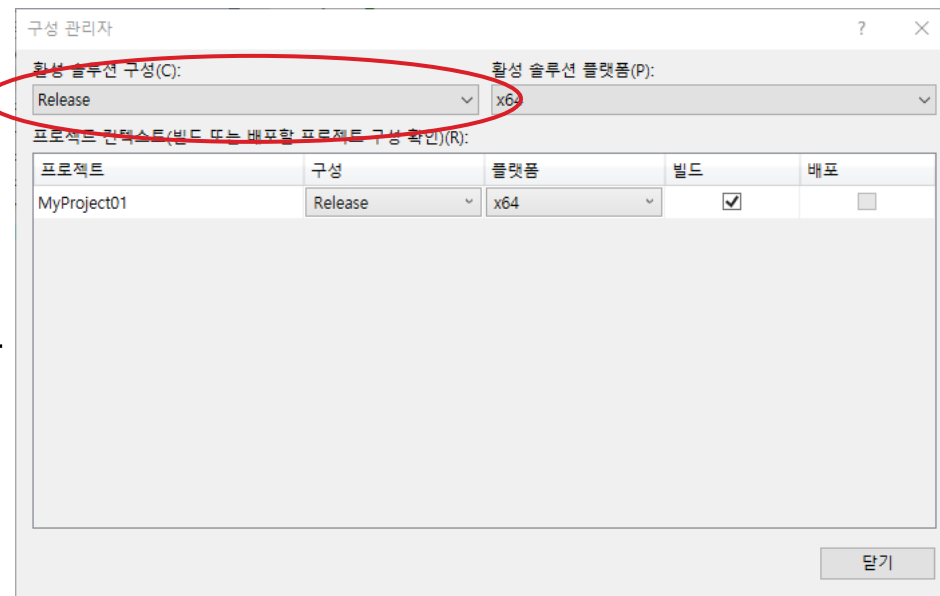
▶ 디버그 모드

- 실행 파일 내에 디버깅 정보 포함
- 실행 파일 크기 증가, 실행 속도 느림
- 생성 폴더 : x64\Debug
- 개발 시 사용

▶ 릴리즈 모드

- 디버깅 정보 포함하지 않음
- 파일 크기 감소, 실행 속도 빠름
- 생성 폴더 : x64\Release
- 배포용 최종 제품을 만들 때 사용

▶ 모드 변경 : [빌드]→[구성 관리자]→[활성 솔루션 구성] 선택



디버깅

▶ 디버깅이란?

- 문법 오류 수정 : 컴파일러에 의해 쉽게 찾을 수 있음
- 논리 오류 수정 : 정상적인 컴파일, 링크 과정을 거쳐 실행 파일이 생성되었지만 실행 결과가 원하는 결과와 다를 경우

→ 일반적으로 디버깅은 논리 오류 수정을 의미함

▶ Visual C++의 디버깅 방법

- 문장 단위 실행
- 해당 문장에서의 변수값 조사
- 실행 중 원하는 위치에서의 멈춤 → break point

디버깅 모드로 실행

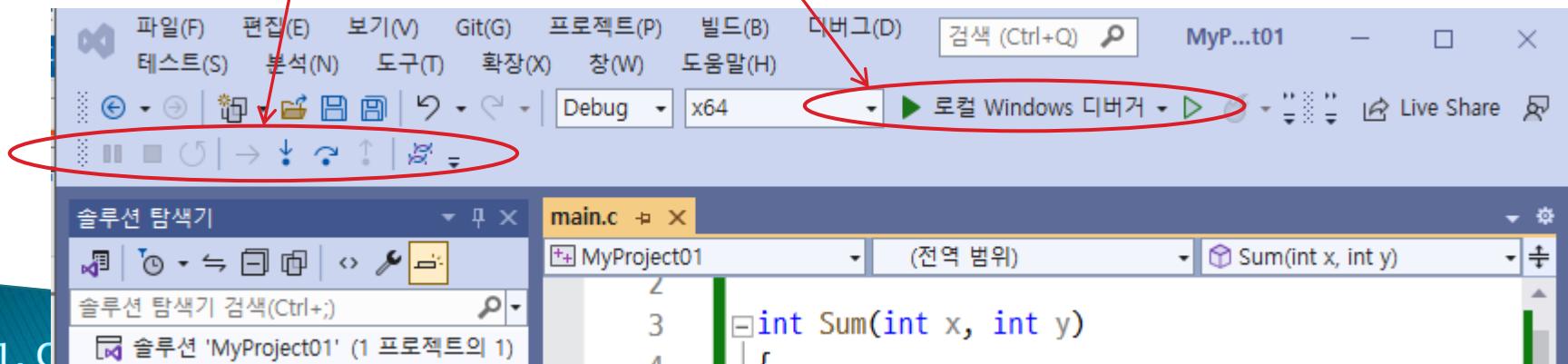
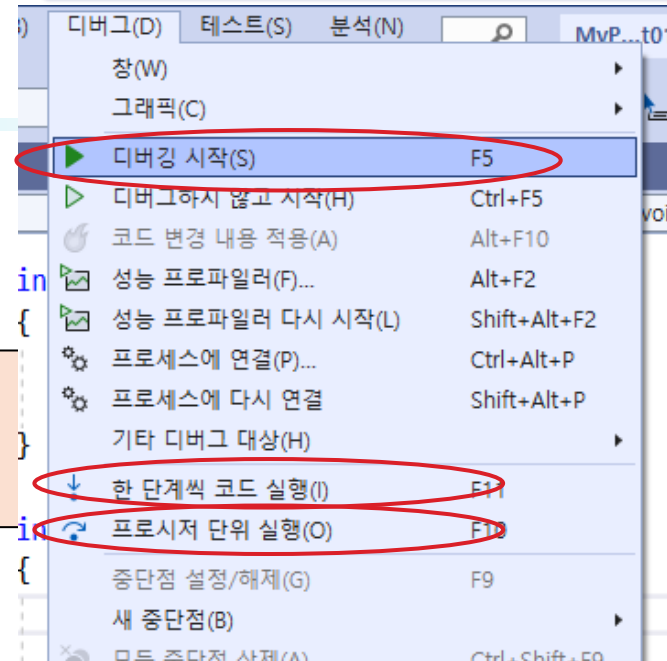
▶ 실행 방법 (1)

- [디버그] → [디버깅 시작]
- 단축키 : F5

▶ 실행 방법 (2)

- 도구모음 [로컬 Windows 디버거]

- 디버그 도구 모음 나타내기 : 도구 모음 위에서 팝업 메뉴(오른쪽 마우스 클릭) → [디버그] 체크 → 디버그 도구모음이 나타남



문장 단위 디버깅

▶ 문장 단위 디버깅 메뉴



- 한 단계씩 코드 실행 : [디버그]→[한 단계씩 코드 실행]
 - 한 문장씩 실행, 함수 호출 시 해당 함수로 이동
- 프로시저 단위 실행 : [디버그]→[프로시저 단위 실행]
 - 한 문장씩 실행, 함수 호출 시 해당 함수를 한 번에 실행
- 프로시저 나가기
 - 현재 함수의 나머지 부분까지 실행 후 함수 호출한 곳으로 이동

▶ 문장 단위 디버깅 시작

- [한 단계씩 코드 실행] 또는 [프로시저 단위 실행]을 수행하면 main 함수부터 한 문장씩 실행함

디버깅 종료

- ▶ 디버깅 종료
 - [디버그]→[디버깅 중지]
 - 단축키 Shift+F5
 - 도구 모음



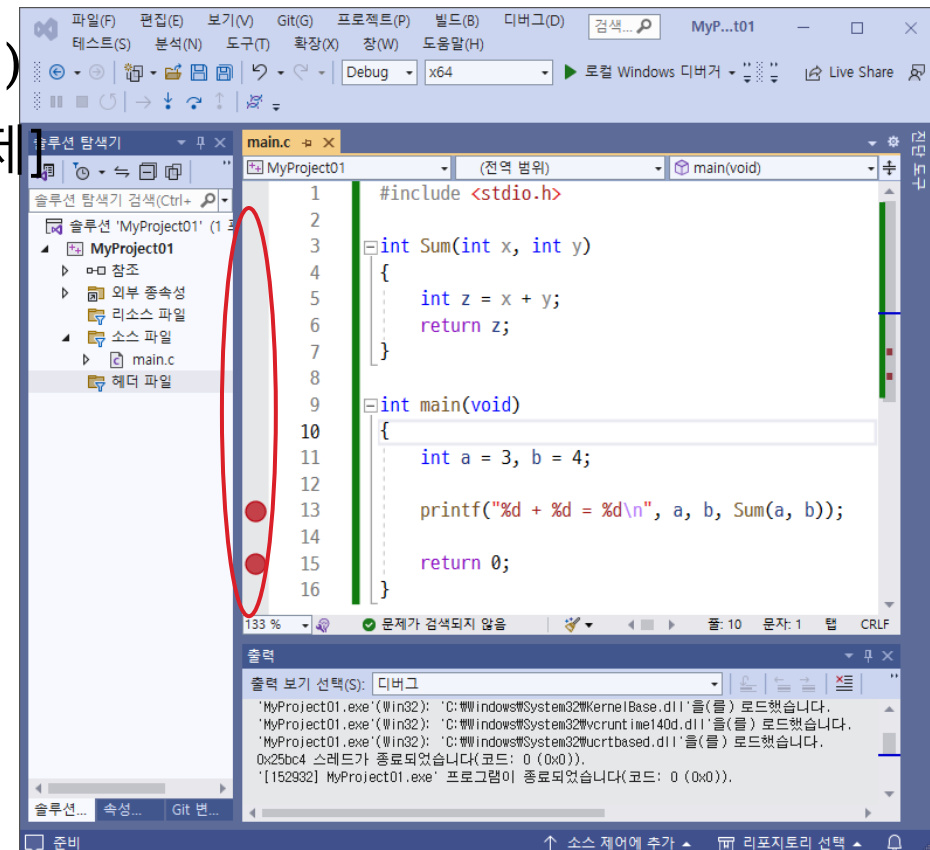
Break Point (1)

▶ break point

- 소스 코드의 임의의 위치에서 실행을 멈추게 하는 방법

▶ beak point 설정 방법

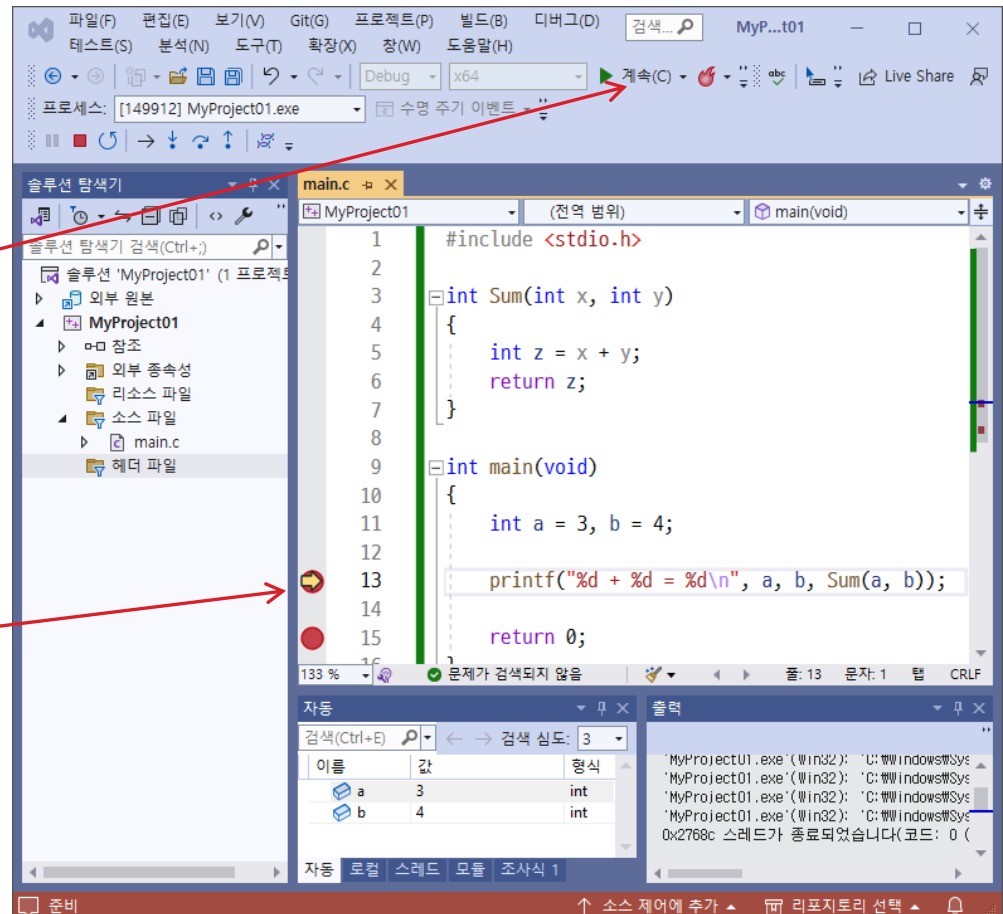
- 커서를 해당 위치로 이동(클릭)
 - [디버그]→[중단점 설정/해제]
 - 단축키 F9
- 해당 문장의 왼쪽 바를 클릭



Break Point (2)

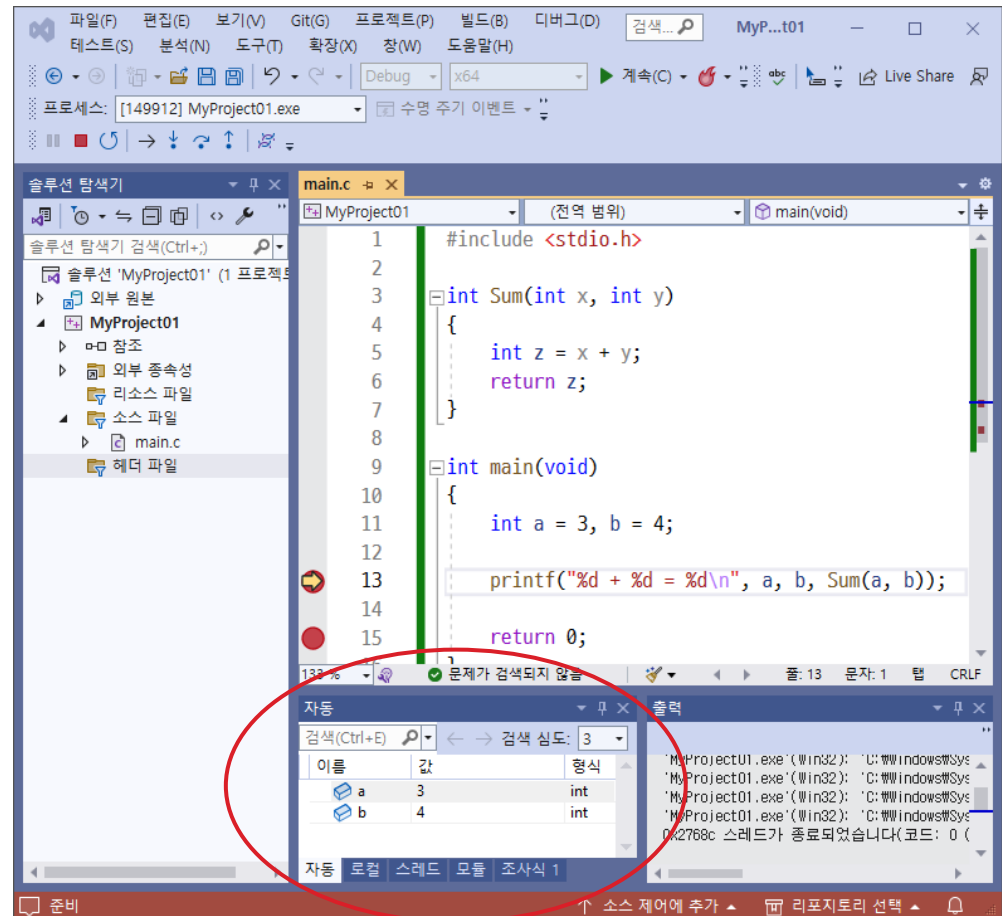
- ▶ 다음 break point까지 실행
 - [디버그]→[계속] : [디버그]→[디버깅 시작]과 아이콘 모양 동일
 - 단축키 F5
 - 도구 모음
 - [계속]

여기서 멈춘 상태



변수값 확인

- ▶ 변수값 확인 방법 (1)
 - 자동 창에서 확인
- ▶ 변수값 확인 방법 (2)
 - 조사식 창에서 값의 확인을 원하는 변수명 입력



이번 장에서 배운 것

- 컴퓨터는 하드웨어와 소프트웨어로 구성되며, 하드웨어는 CPU를 중심으로 주기억장치, 보조기억장치, 입력장치, 출력장치로 구성된다.
- UNIX 운영체제를 개발하기 위해 탄생한 C 언어는 절차지향 언어로서 고급 언어의 특징뿐만 아니라 저급 언어(메모리 접근)의 특징도 가지고 있다.
- 표준 C는 기본적으로 콘솔 프로그래밍을 지원하며, 이 언어를 확장하여 Windows 프로그램을 만들 수도 있다.
- 프로그램 작성 과정은 코딩, 전처리, 컴파일, 링크, 디버깅, 실행으로 이루어진다.
- Visual C++를 사용하면 프로젝트 생성, 코딩 등 프로그램 작성을 위한 모든 과정을 수행할 수 있다.
- Visual C++를 사용하면 디버깅 모드를 통해 한 줄씩 실행이 가능하다.