2강. VC++ 2010 사용 방법

C++ 프로그래밍

jhhwang@kumoh.ac.kr

컴퓨터공학과 황준하

목차

- 프로그램 개발 단계
- 프로그램 작성 단계
- **Visual C++ 2010** 사용 방법 : 따라하기
- 디버깅 방법

프로그램 개발 단계

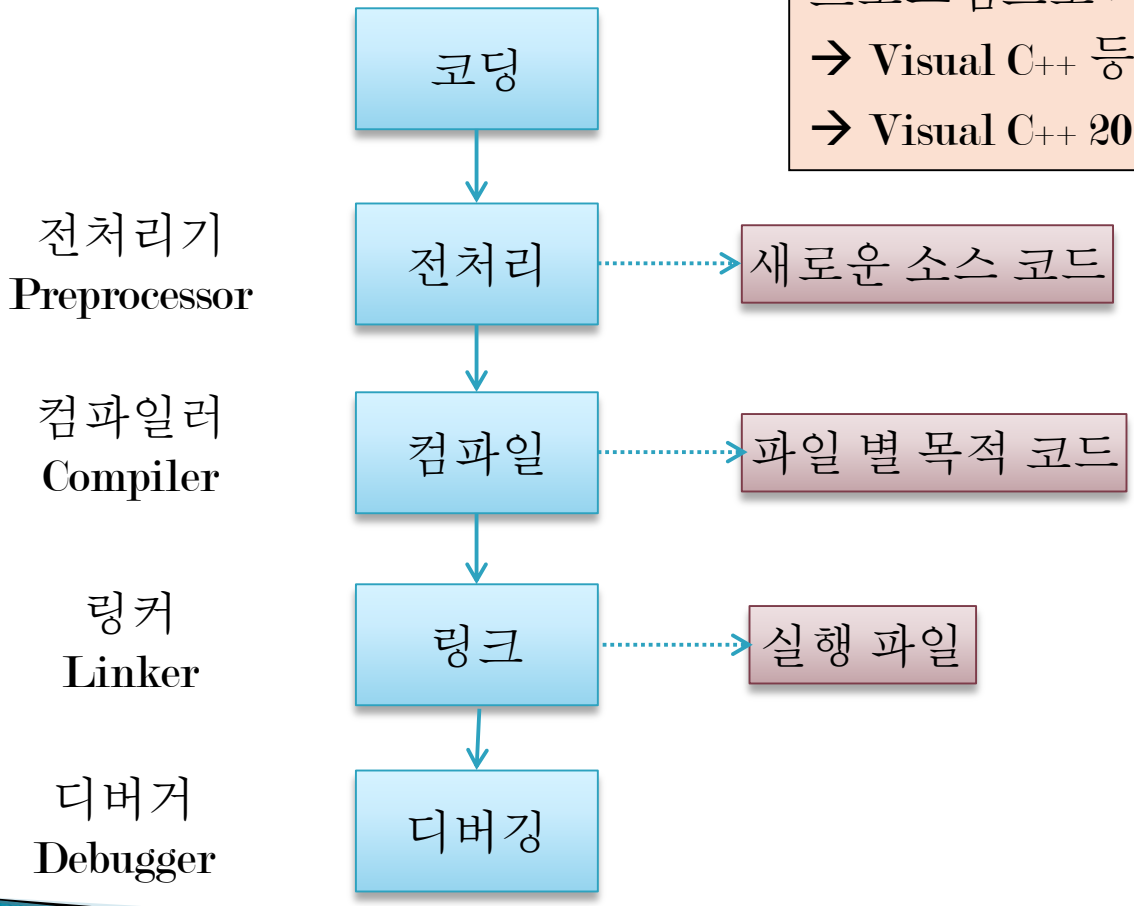
- 프로그램
  - 컴퓨터가 특정 작업 또는 문제를 해결하도록 나열된 명령어들의 집합
- 프로그램 개발 단계

1. 문제 분석 : 수행하고자 하는 작업을 명확히 기술
   - 관련 과목 : 소프트웨어 공학
2. 알고리즘 개발 : 그 작업을 해결할 수 있는 방법 연구
   - 관련 과목 : 데이터 구조 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 : 소프트웨어 공학

프로그램 작성 단계

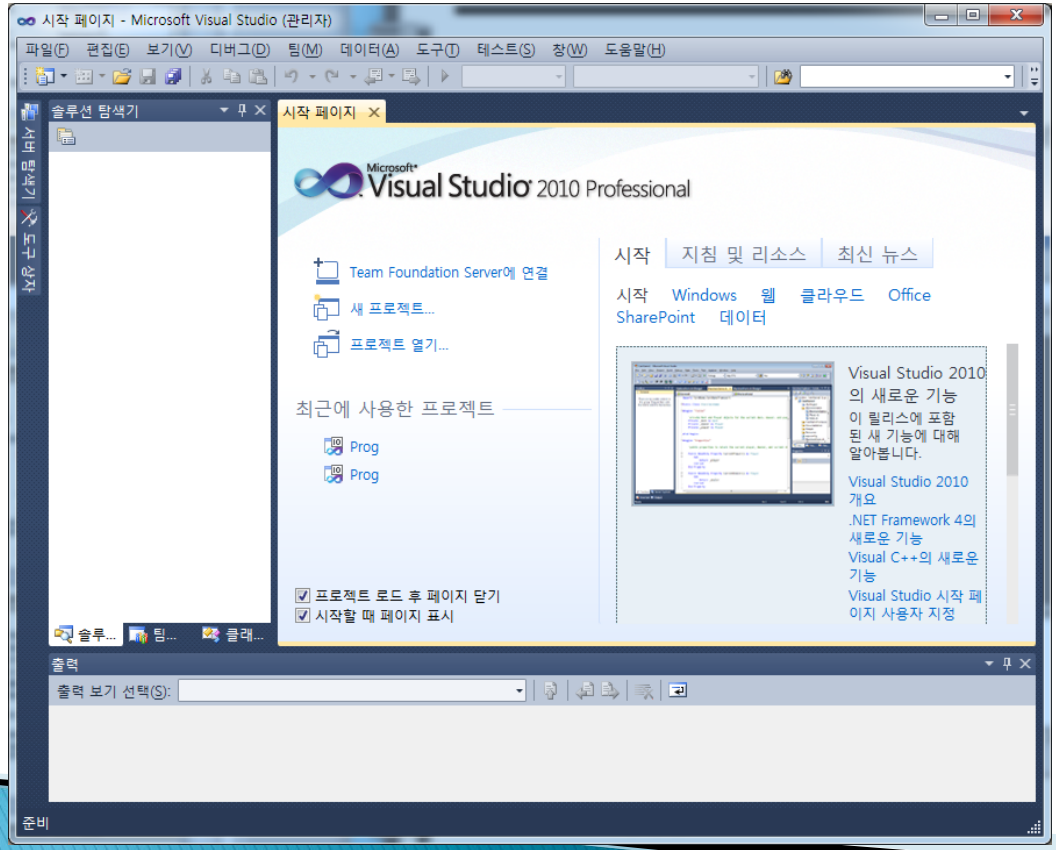
- 프로그램 작성 및 실행 단계

통합개발환경(IDE) : 모든 과정을 하나의 프로그램으로 처리 가능
→ Visual C++ 등
→ Visual C++ 2010(10.0) 기준 설명



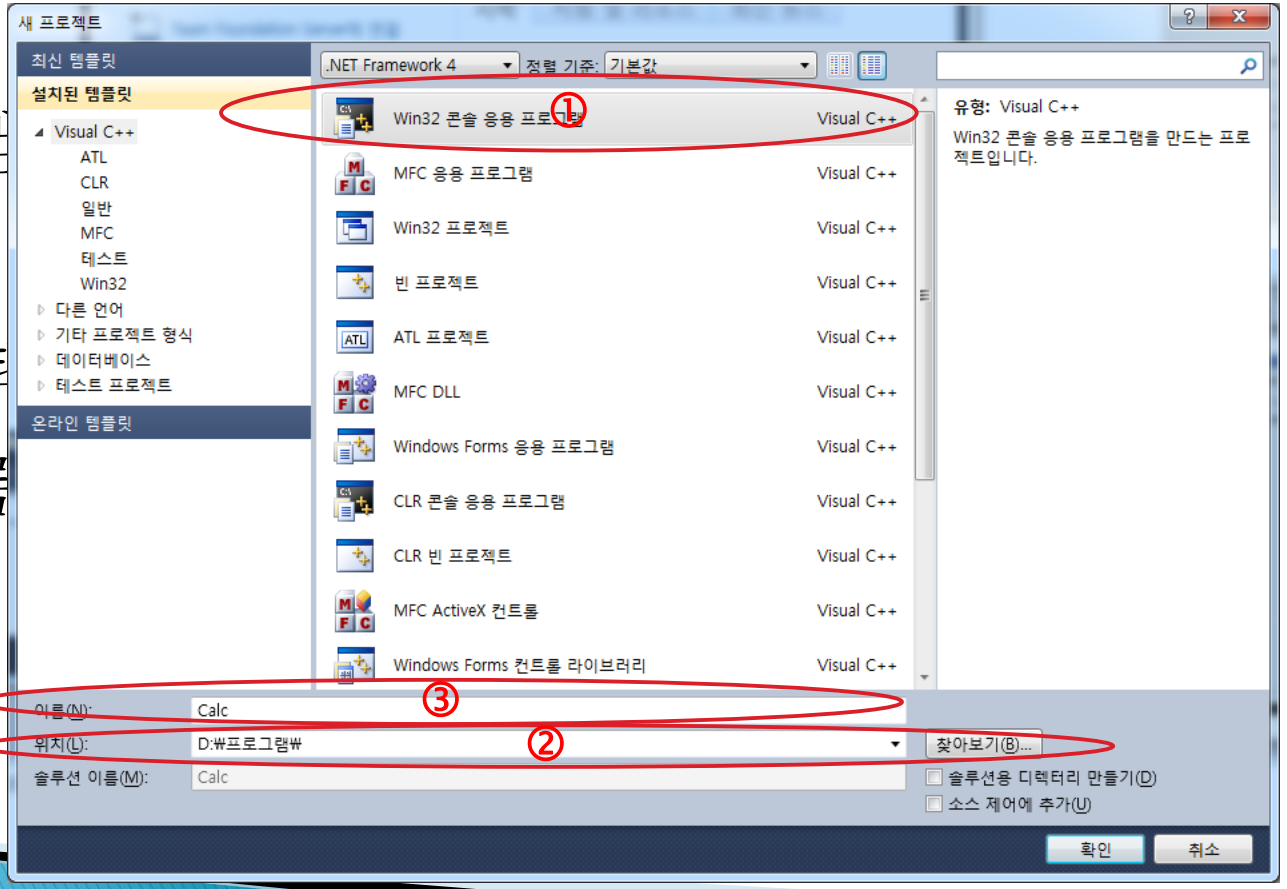
Visual C++ 2010 사용 방법 : 따라하기

- Visual Studio 2010 실행
  - [시작] → [모든 프로그램] → [Microsoft Visual Studio 2010] → [Microsoft Visual Studio 2010]



프로젝트 만들기

- 프로젝트 생성
  - [파일] → [새로 만들기] → [프로젝트]
- **Win32** 콘솔 응용 프로그램 선택
- 폴더 위치 선택
- 프로젝트 이름 입력

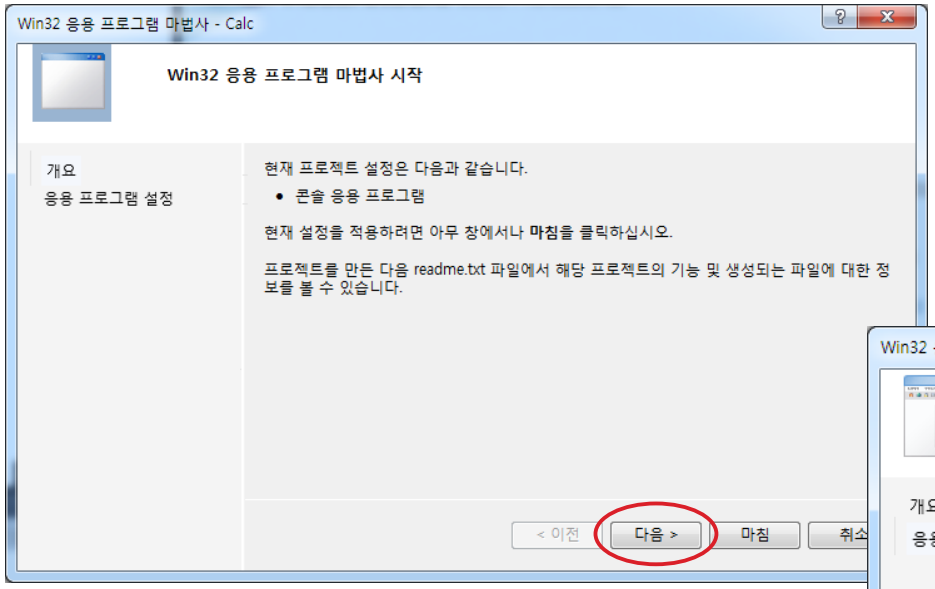


프로젝트 만들기

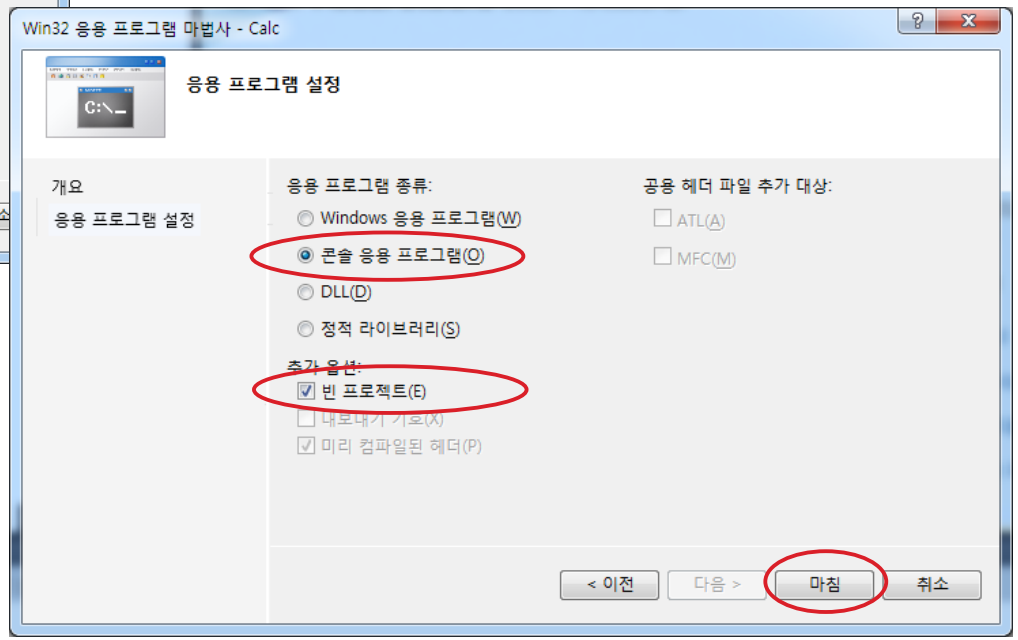
- 용어 설명
  - 솔루션
    - 하나의 소프트웨어 ← 여러 개의 프로그램 포함 가능
  - 프로젝트
    - 하나의 프로그램
- 새 프로젝트 작성 시
  - 솔루션용 디렉터리 만들기 선택 : 별도의 프로젝트 폴더 생김
  - 선택하지 않을 경우 솔루션 폴더 밑에 프로젝트가 함께 생성됨

프로젝트 만들기

- Win32 응용 프로그램 마법사 시작

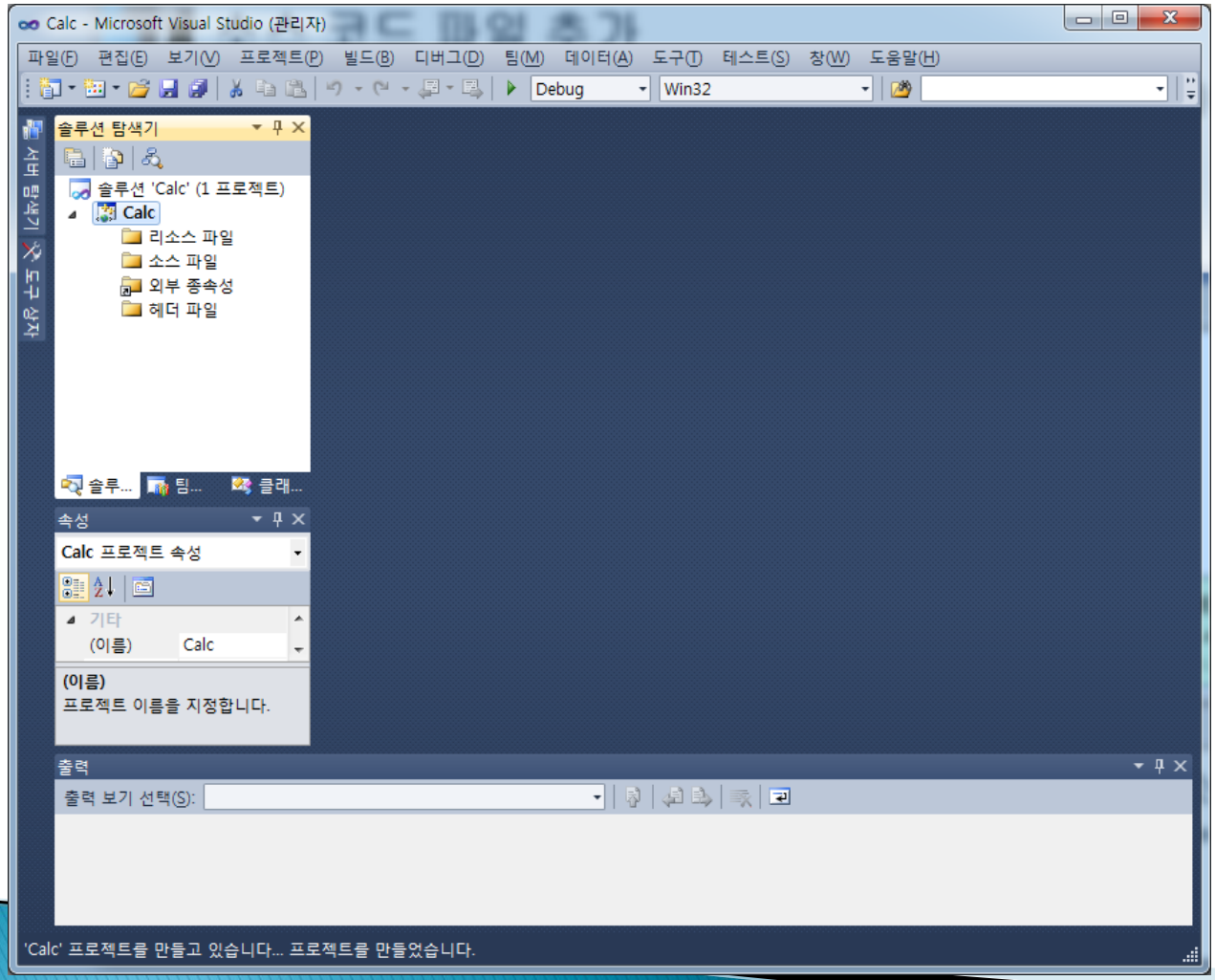


콘솔 응용 프로그램
빈 프로젝트 선택



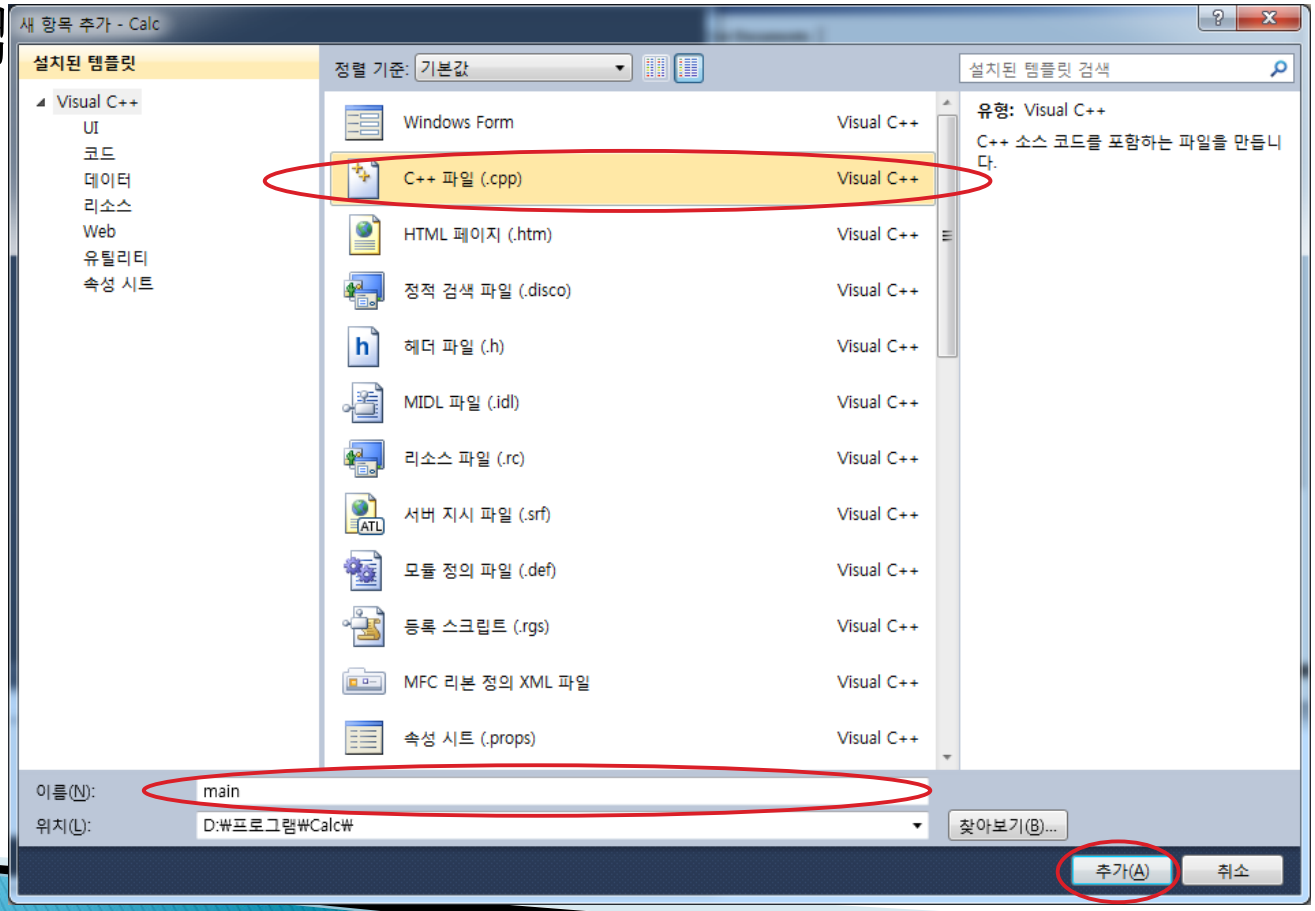
프로젝트 만들기

- 프로젝트 생성 완료



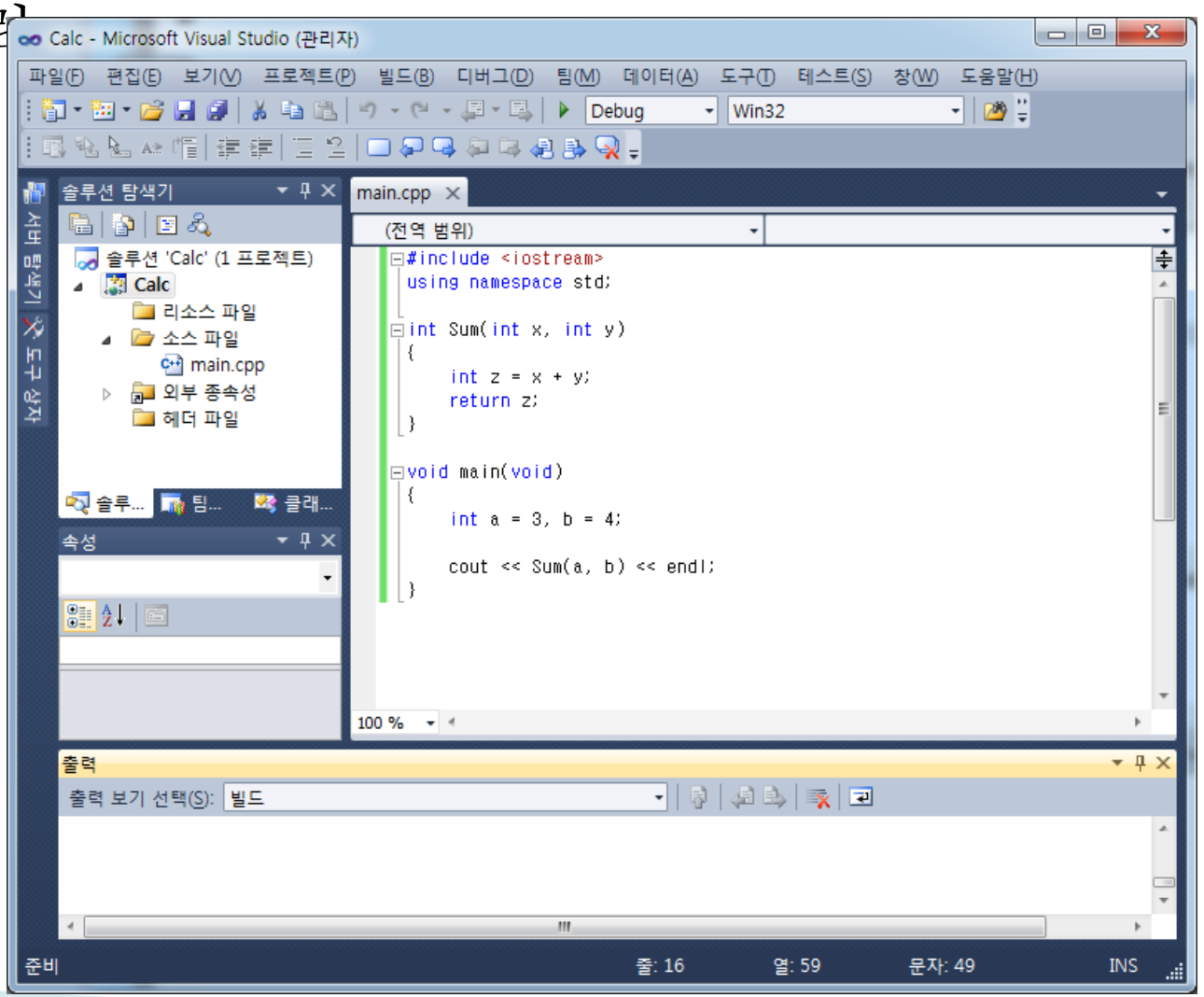
코딩하기

- 소스 파일 추가
  - [프로젝트] → [새 항목 추가...]
  - C++ 파일 선택
  - 파일명 입력



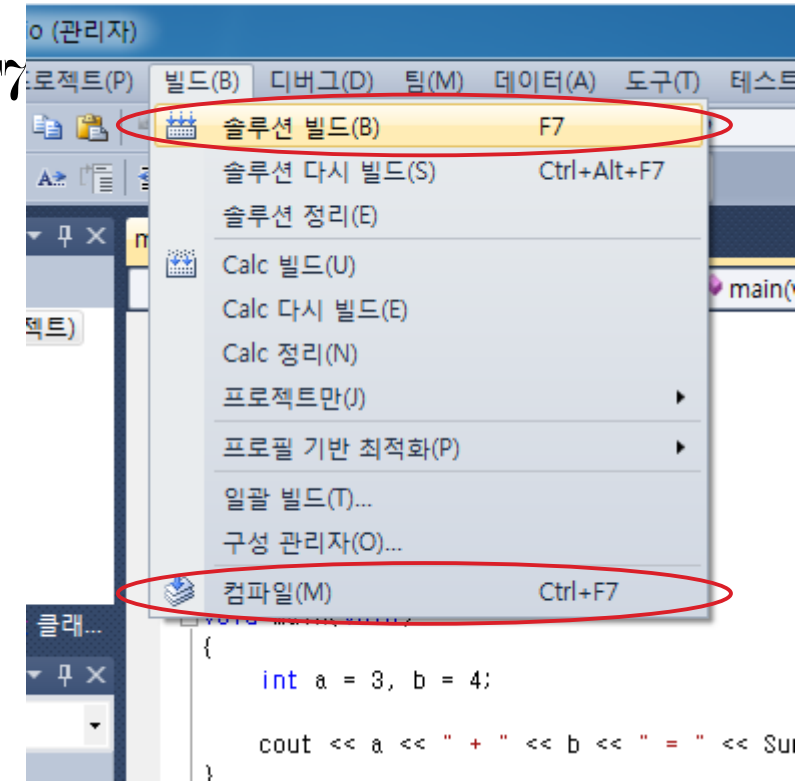
코딩하기

- 소스 코드 입력


Calc - Microsoft Visual Studio (관리자)
파일(F) 편집(E) 보기(V) 프로젝트(P) 빌드(B) 디버그(D) 팀(M) 데이터(A) 도구(T) 테스트(S) 창(W) 도움말(H)
Debug
Win32
솔루션 탐색기
솔루션 'Calc' (1 프로젝트)
Calc
리소스 파일
소스 파일
main.cpp
외부 종속성
헤더 파일
main.cpp
(전역 범위)
#include <iostream>
using namespace std;
int Sum(int x, int y)
{
int z = x + y;
return z;
}
void main(void)
{
int a = 3, b = 4;
cout << Sum(a, b) << endl;
}
100 %
출력
출력 보기 선택(S): 빌드
준비
줄: 16
열: 59
문자: 49
INS

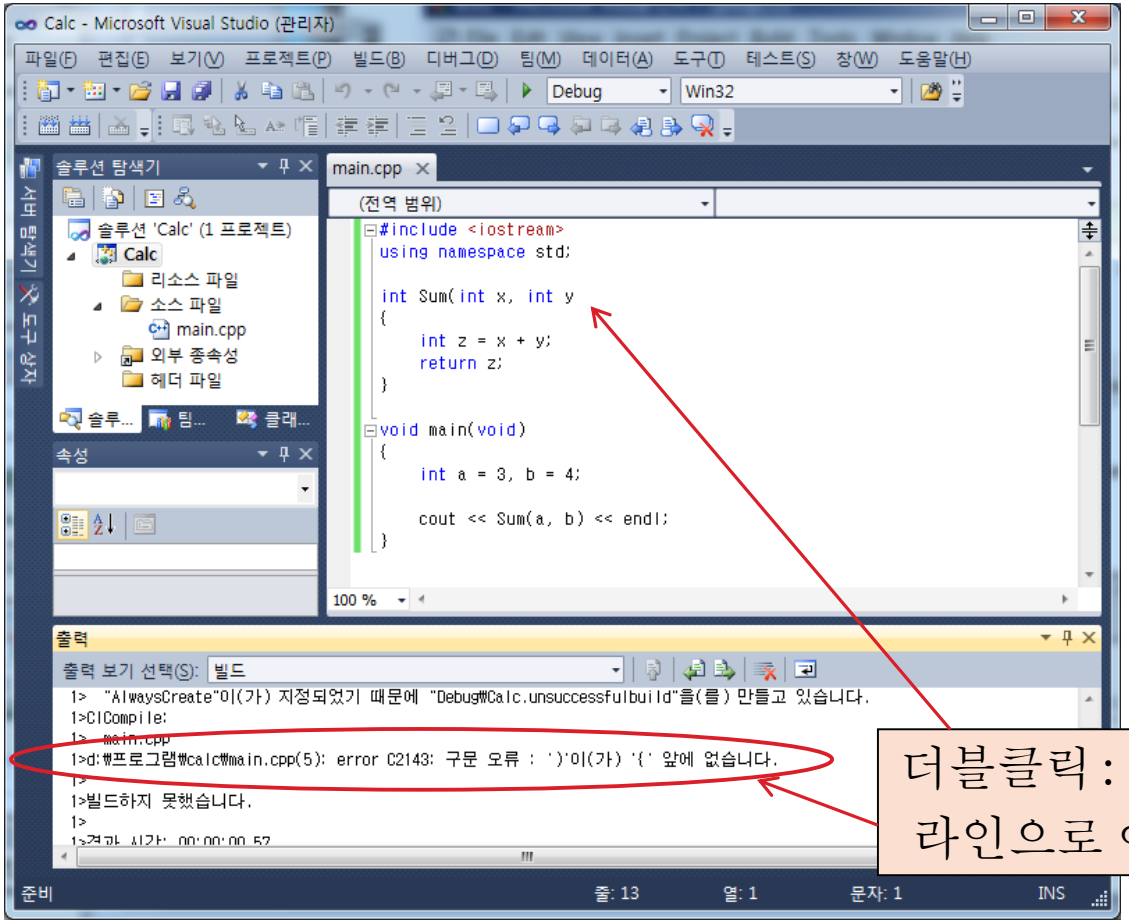
컴파일하기

- 컴파일
  - [빌드] → [컴파일] 또는 단축키 Ctrl+F7
- 링크
  - [빌드] → [솔루션 빌드] 또는 F7
  - 링크를 수행하면 자동으로 컴파일 실행됨



컴파일하기

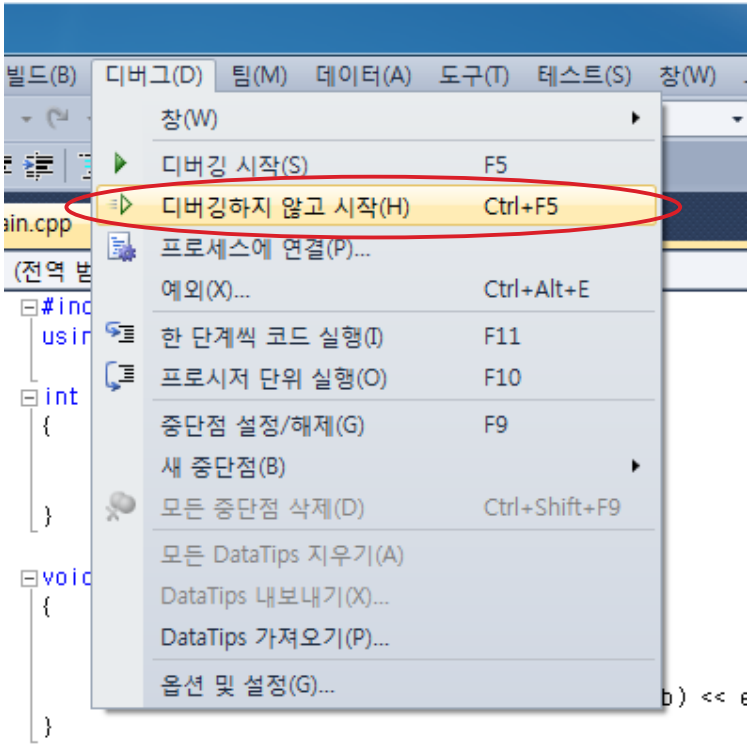
- 오류가 발생한 경우

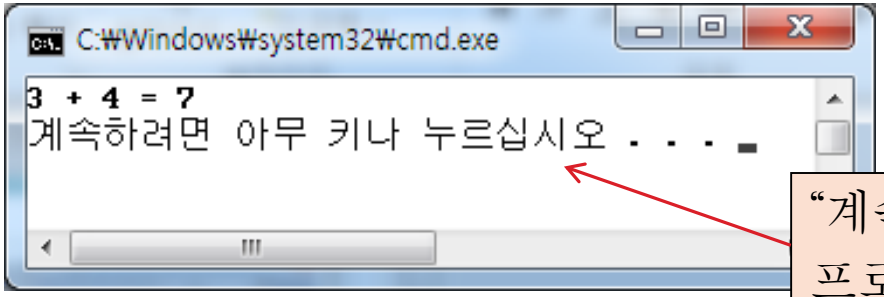


- error(오류): 프로그램 수행 불가능 상태 → 컴파일 실패
- warning(경고): 프로그램 수행은 가능하나 논리적 오류 가능

실행하기

- 실행 방법 (1) : VC++ 내에서 실행
  - [디버그] → [디버깅하지 않고 시
  - 단축키 : Ctrl+F5

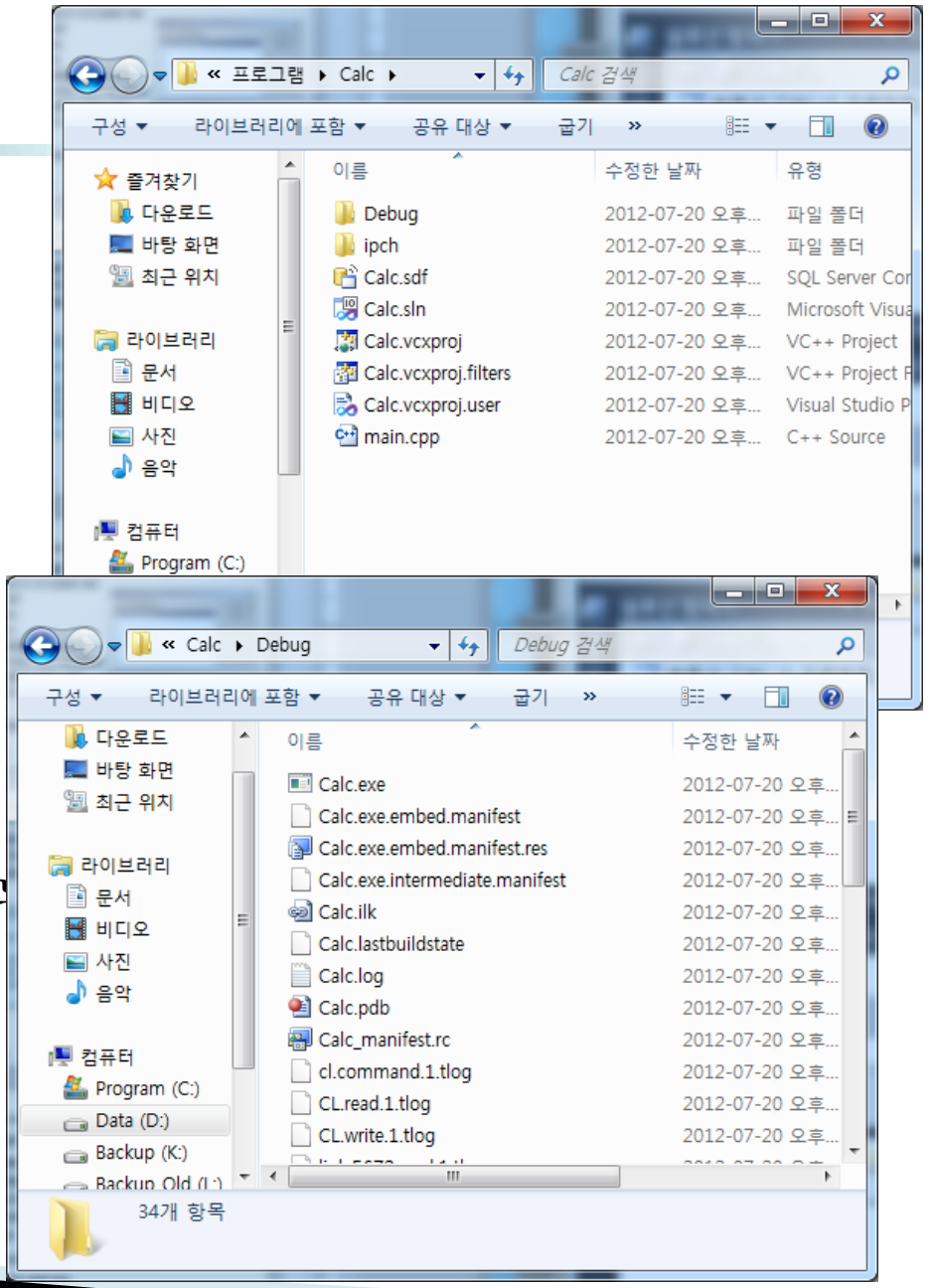




"계속하려면 아무 키나 누르십시오..."는 프로그램과 상관없음

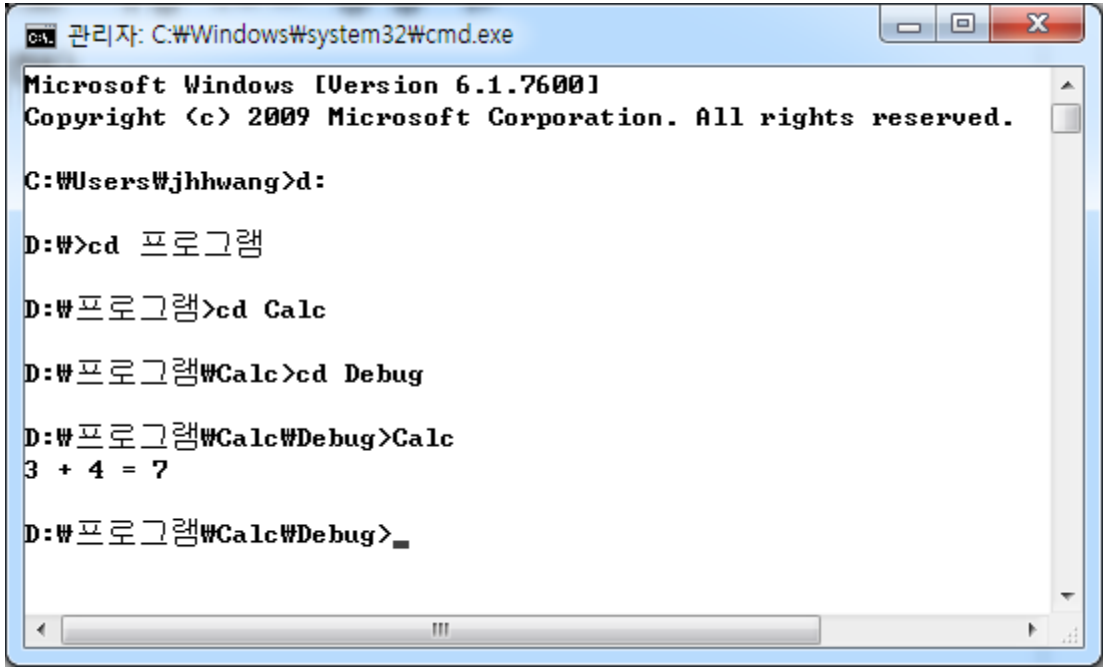
실행하기

- 프로젝트 폴더 구성
  - .sln : 솔루션 관리 파일
  - .vcxproj : 프로젝트 관리 파일
  - .cpp : 소스 파일
  - 디버그 폴더
    - .exe : 실행 파일
    - 탐색기에서 바로 실행 가
      - 실행 후 바로 종료



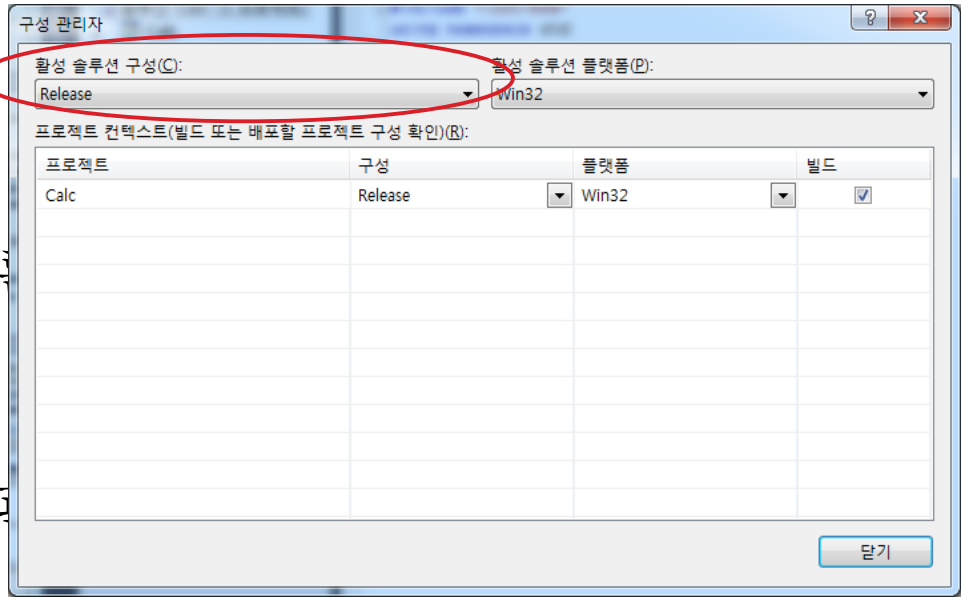
실행하기

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



디버그 모드와 릴리즈 모드

- 디버그 모드
  - 실행 파일 내에 디버깅 정보 포함
  - 실행 파일 크기 증가, 실행 속도 느림
  - 생성 폴더 : Debug
  - 개발 시 사용
- 릴리즈 모드
  - 디버깅 정보 포함하지 않음
  - 파일 크기 감소, 실행 속도 빠
  - 생성 폴더 : Release
  - 최종 제품 만들 때 사용
- 모드 변경 : [빌드]→[구성
  선택



연습 문제

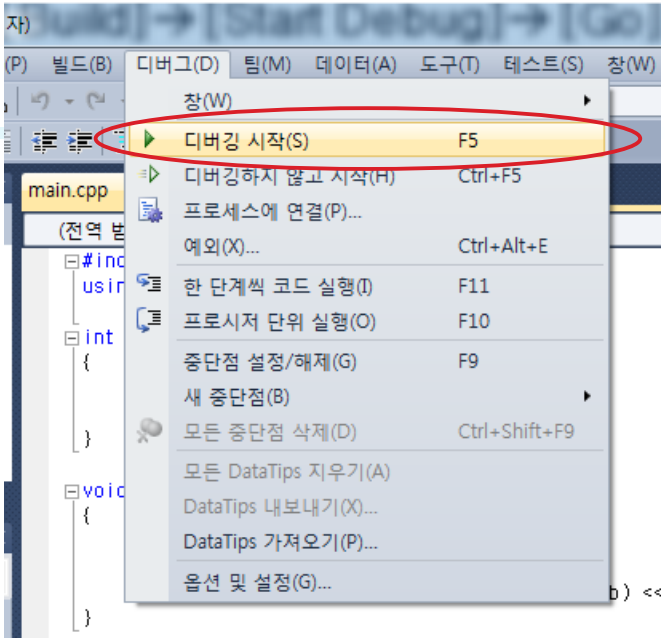
- 2개의 실수값을 입력값으로 받아 사칙연산 결과를 출력하는 Calc 함수를 만들고 main 함수 내에서 Sum 함수를 테스트해 보라.
  - VC++ 컴파일러를 사용하여 코딩부터 실행에 이르기까지의 전 과정을 실습해 보라.
  - 디버그 모드와 릴리즈 모드로 컴파일한 후 실행 파일의 크기를 비교해 보라.

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**

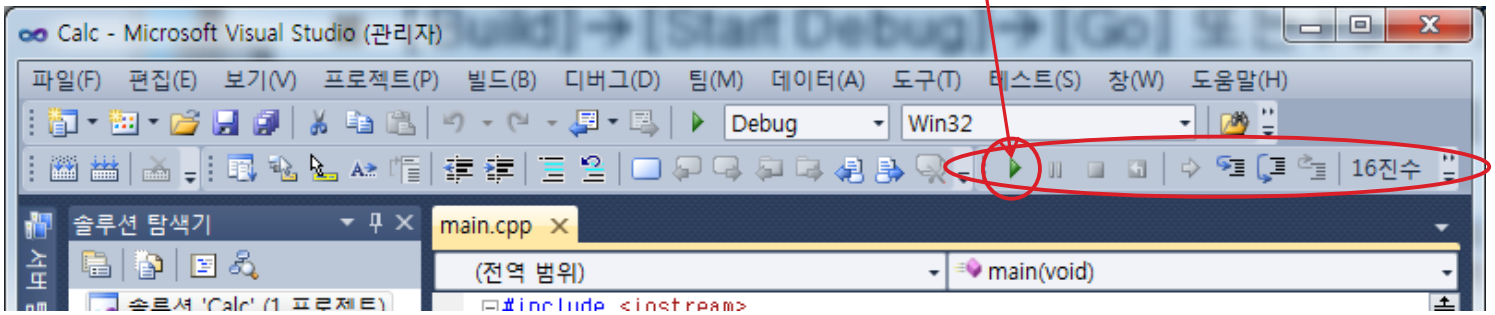
디버깅 모드로 실행



- 실행 방법 (1)
  - [디버그] → [디버깅 시작]
  - 단축키 : F5
- 실행 방법 (2)
  - 도구모음 위로 마우스 이동

→ 팝업 메뉴(오른쪽 마우스 클릭) → [디버그] 체크

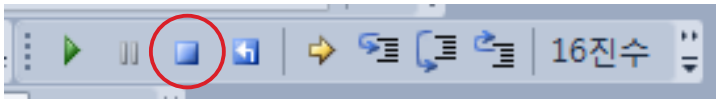
→ 디버그 도구모음이 나타남 → 디버깅 시작 버튼 클릭



문장 단위 디버깅

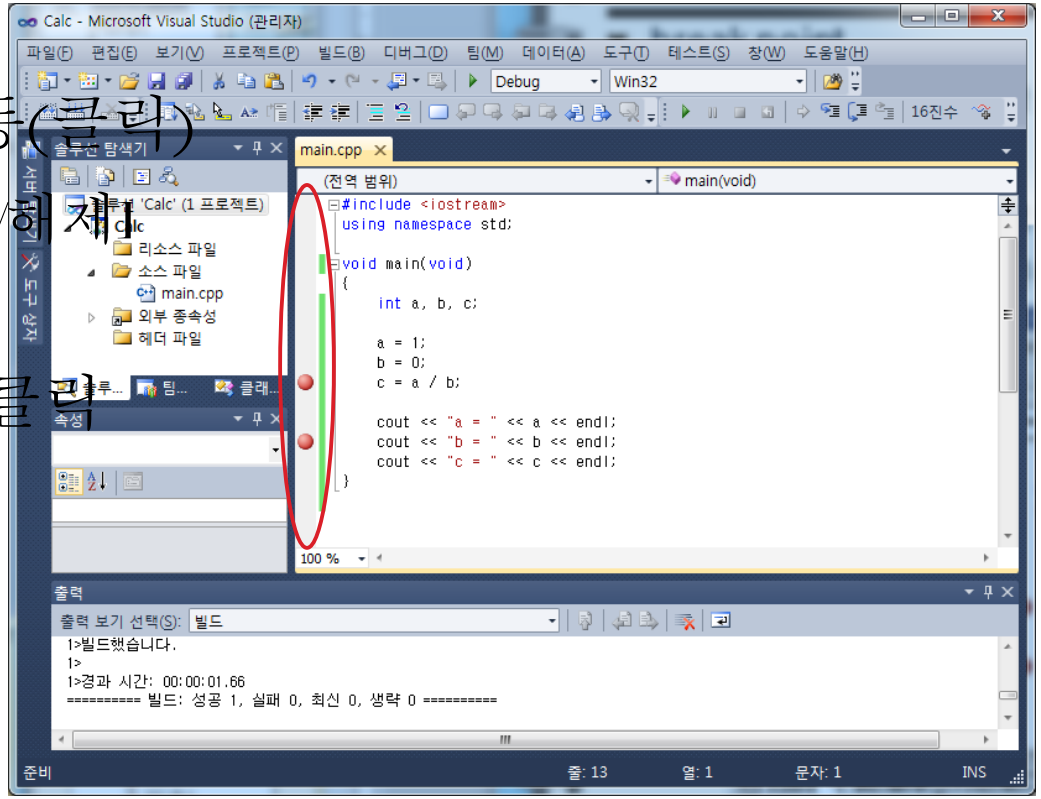
- 문장 단위 디버깅 메뉴
  - 한 단계씩 코드 실행 : [디버그]→[한 단계씩 코드 실행]
    - 한 문장씩 실행, 함수 호출 시 해당 함수로 이동
  - 프로시저 단위 실행 : [디버그]→[프로시저 단위 실행]
    - 한 문장씩 실행, 함수 호출 시 해당 함수를 한 번에 실행
  - 프로시저 나가기
    - 현재 함수의 나머지 부분까지 실행 후 함수 호출한 곳으로 이동
- 문장 단위 디버깅 시작
  - [한 단계씩 코드 실행] 또는 [프로시저 단위 실행]을 수행하면 main 함수부터 한 문장씩 실행함

디버깅 종료

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

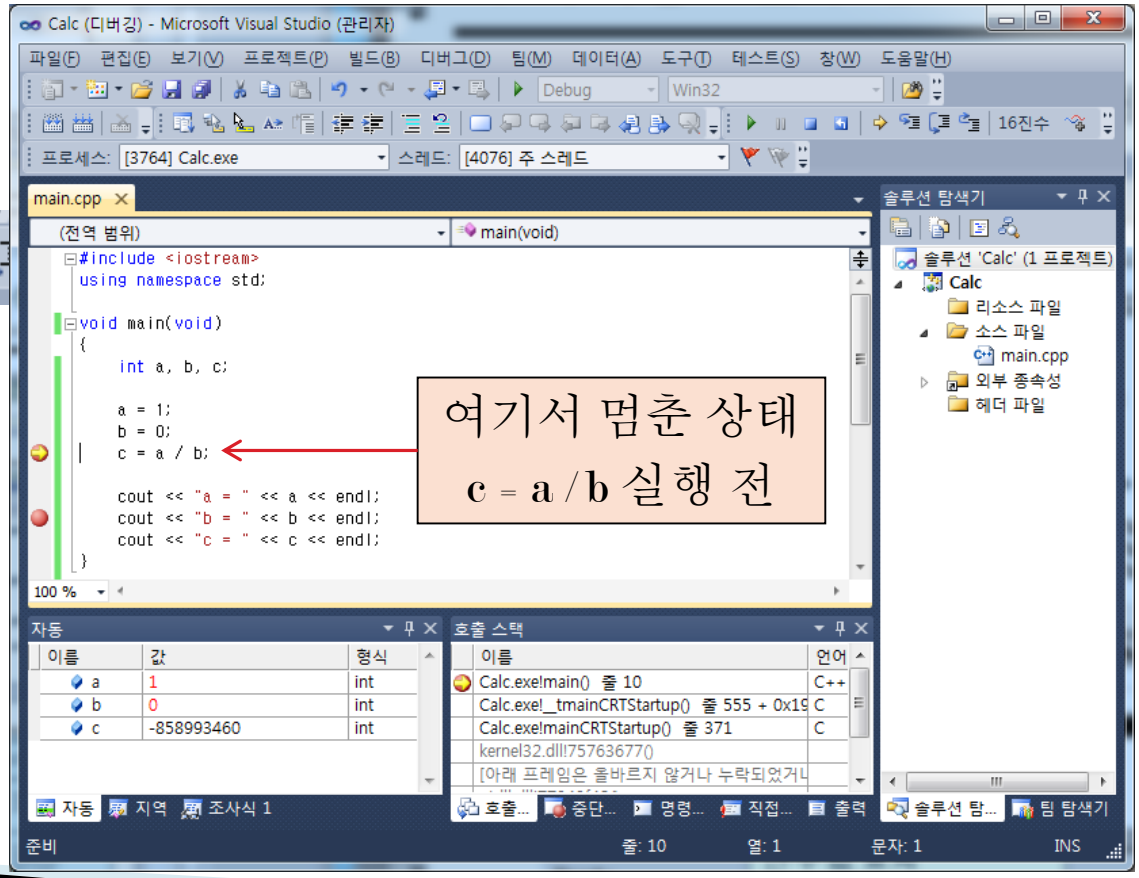
Break Point

- break point
  - 소스 코드의 임의의 장소에서 실행을 멈추게 하는 방법
- beak point 설정 방법
  - 커서를 해당 위치로 이동(클릭)
    - [디버그]→[중단점 설정/해제]
    - 단축키 F9
  - 해당 문장의 왼쪽 바를 클릭



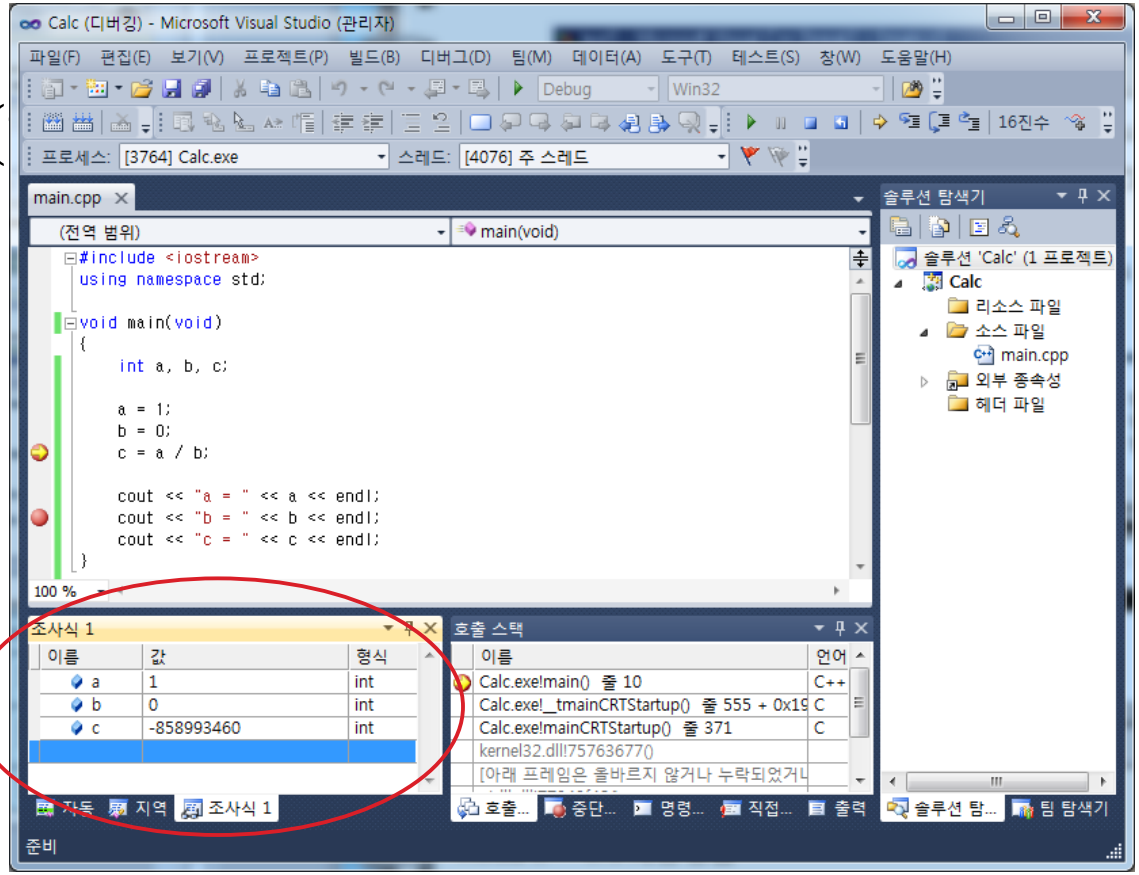
Break Point

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



변수값 확인

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



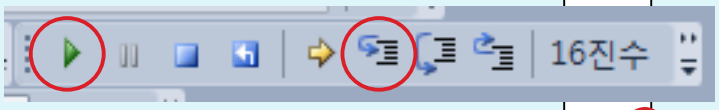
연습 문제

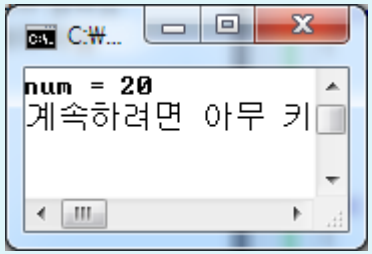
- 다음 코드를 실행해 보고 실행 화면과 같이 최종 결과가 20(10 + 10의 결과)인지 확인해 보라. 그렇지 않다면 다음과 같은 디버깅 방법들을 통해 디버깅 연습을 해 보라.
  - break point 설정, 단계별 실행, 변수값 확인

```
#include <iostream>
using namespace std;

int func1(void);
int func2(int x);

void main(void)
{
    int num = func1();
    cout << "num = " << num << endl;
}
```






```
int func1(void)
{
    int num = 10;
    int num2 = func2(num);
    return (num2);
}

int func2(int x)
{
    int ret = x * x;
    return (ret);
}
```