

IT CookBook, 전공자를 위한 C 언어 프로그래밍

[강의교안 이용 안내]

- 본 강의교안의 저작권은 주우석과 한빛아카데미(주)에 있습니다.
- 이 자료는 강의 보조자료로 제공되는 것으로, 학생들에게 배포되어서는 안 됩니다.

1장. C 언어와 컴파일러

- 1.1. C 언어 소개
- 1.2. Visual C 컴파일러 사용법
- 1.3. Hello, world.
- 1.4. 프로그램 작성 및 실행



- Compile
 - to collect into a volume
 - to compose out of materials from other documents
- Compiler
 - one that compiles
 - a computer program that translates instructions written in a higher-level symbolic language into machine language

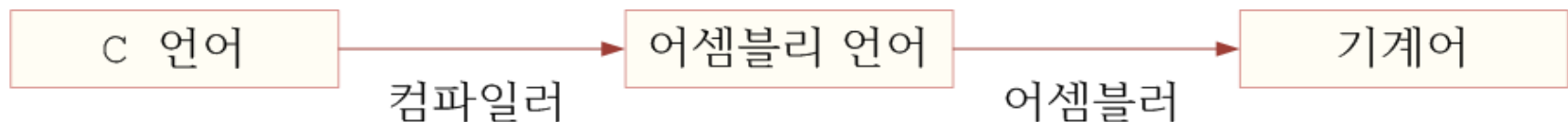
(Webster's New Collegiate Dictionary)

고급 언어와 저급 언어

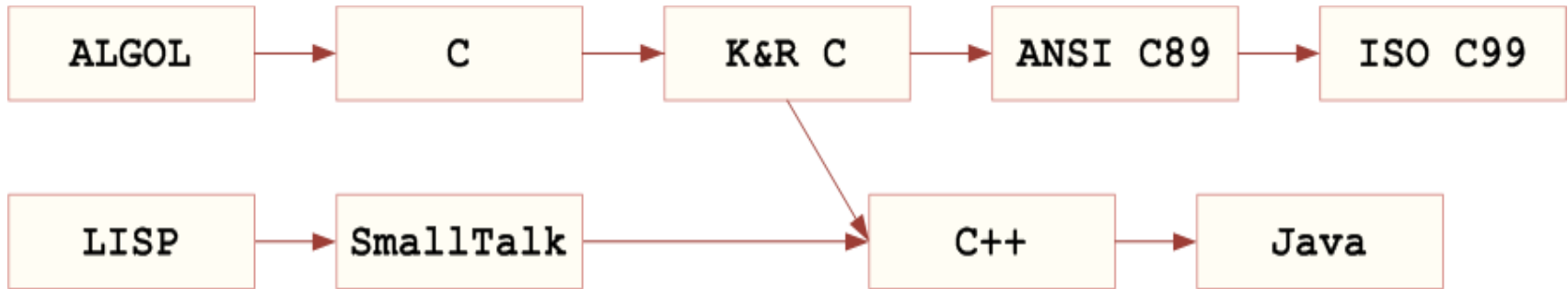
- 기계어 10011111 00000011
- 어셈블리 언어 `mov %ax, $3`
- C 언어 `a = 3;`

C 언어	어셈블리 언어	기계어
<code>count = 1;</code>	<code>mov %ax, \$1</code>	10011111 00000001
<code>while (count <= 10)</code>	<code>loop: add \$2, %ax</code>	00101101 00000010
<code>count = count + 2;</code>	<code>cmp \$10, %ax</code>	00110001 00001010
	<code>jle loop</code>	11010001 10110001

- 고급 언어와 저급 언어



고급 언어의 진화



- C by Dennis Ritchie and Kenneth Thompson, 1972
- K&R C: Kernighan and Ritchie C
- ANSI C89가 사실상의 표준 (de facto standard)
- 절차적 언어: 절차에 주안점
 - 이렇게 하라. 그 다음에는 저렇게 하라.
- 객체지향 언어: 객체 (Object, 대상)와 객체의 임무에 주안점
 - 객체 A는 이러이러한 작업을 담당하라.
 - 객체 B는 이러이러한 작업을 담당하라.

C 언어의 장점

- 어셈블리 언어
 - **고속성**: 레지스터나 메모리를 직접 다룰 수 있는 **구체적 수준**의 언어
- C 언어
 - **추상적 수준**의 언어: 어셈블리 언어로 번역하는 것은 컴파일러에게 일임. 추상적 수준에서 프로그램을 여러 모듈로 분할하는 구조적 프로그래밍이 가능.
 - **간결성**: 실행 파일 크기가 작음.
 - **이식성**: 하드웨어나 운영체제가 바뀌어도 호환성이 높음.
 - **고속성**: 다른 고급 언어에 비해 실행 속도가 빨라 고급 어셈블리 언어라 부름. 운영체제, 컴파일러, 데이터베이스, 임베디드 프로세서 작성에 활용.
 - 오랫동안 쌓인 풍부한 함수 라이브러리

Visual C 컴파일러 사용법

시작 페이지

시작

5분 안에 첫 번째 앱 빌드

이러한 Visual Studio 관련 팁과 요령을 활용하여 생산성 극대화

최신 기술을 활용하여 멋지고, 저렴하며, 신뢰할 수 있는 웹 사이트 배포

완전한 네이티브의 최신 Android 및 iOS 앱 개발

최근 항목

로컬에서 연 프로젝트, 솔루션 및 폴더가 여기에 표시됩니다.

Git 리포지토리 및 다른 소스 제어 공급자의 원격 호스트가 로그인한 다른 장치의 최근 목록에 표시됩니다.

열기

원격 버전 제어 시스템에서 코드를 가져오거나 로컬 드라이브의 항목을 엽니다.

다음에서 체크 아웃:

Visual Studio Team Services

프로젝트/솔루션 열기

폴더 열기

웹 사이트 열기

새 프로젝트

프로젝트 템플릿 검색

최근 프로젝트 템플릿:

빈 프로젝트 C++

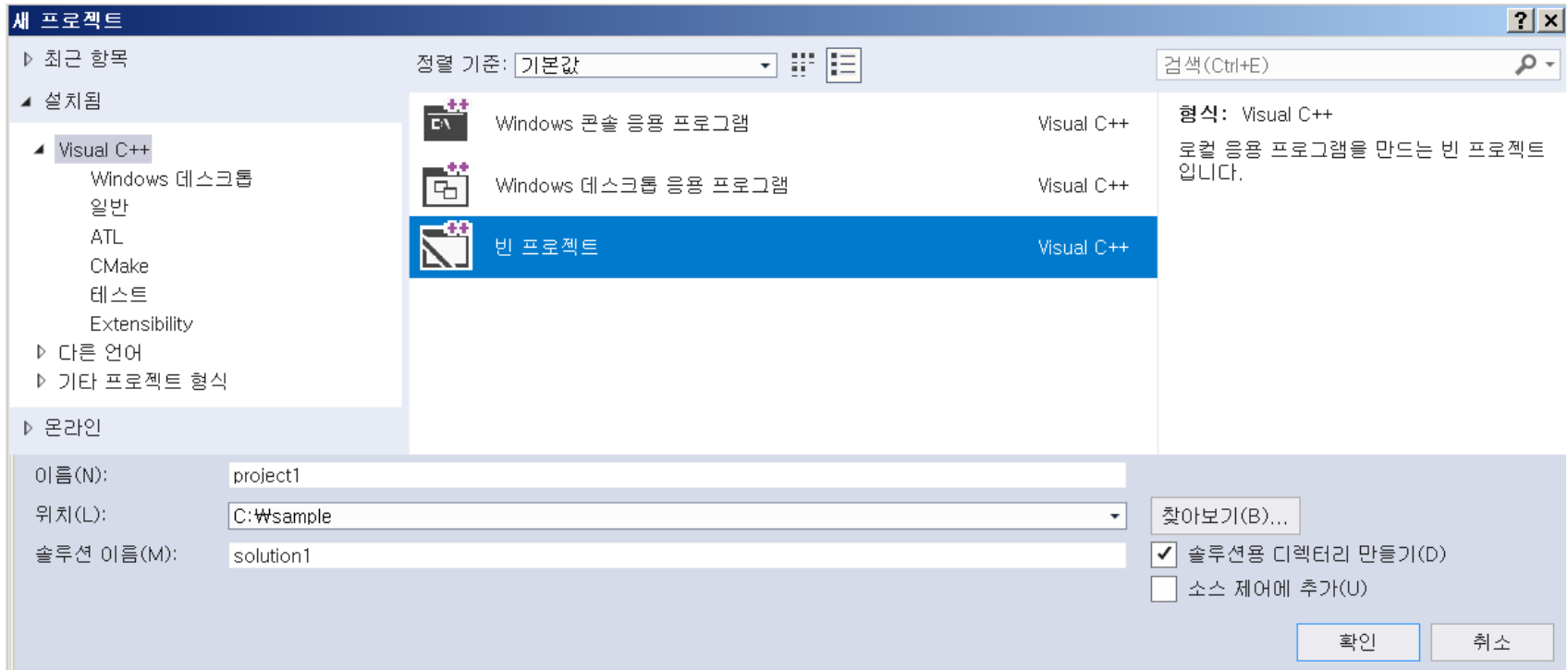
Windows 데스크톱 마법사 C++

Windows 콘솔 응용 프로그램 C++

빈 솔루션 Visual Studio 솔루션

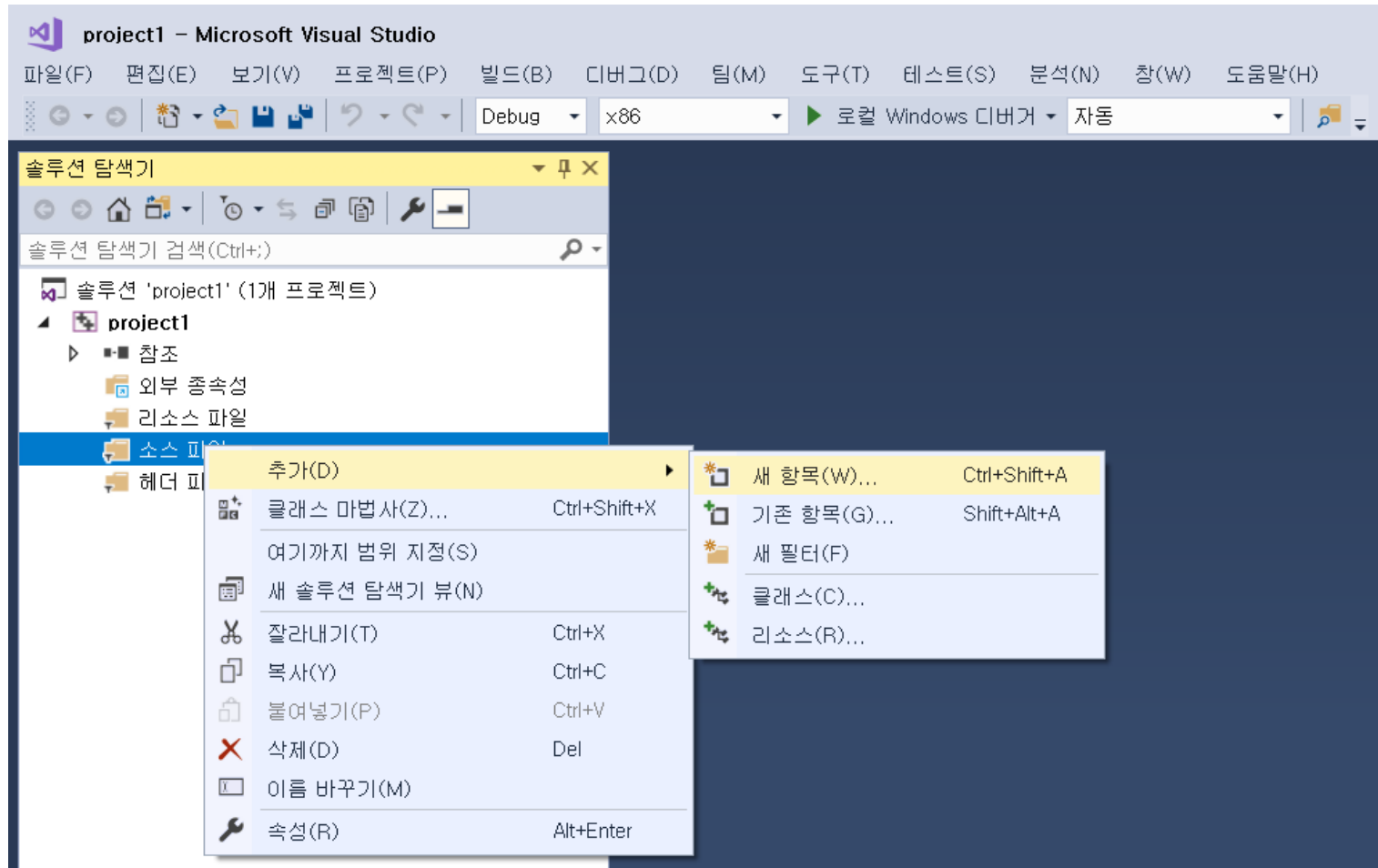
새 프로젝트 만들기...

Visual C 컴파일러 사용법



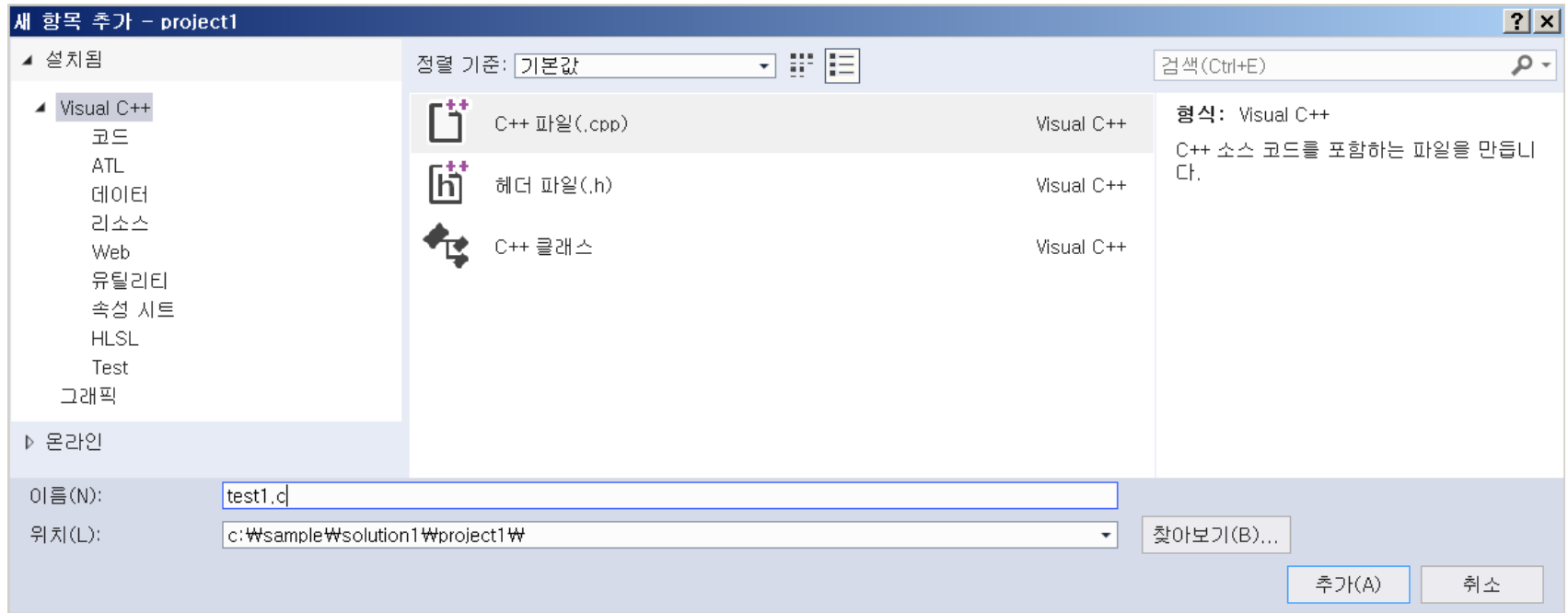
- 소스코드가 모여서 프로젝트를 이루고, 프로젝트가 모여 솔루션을 이룬다.

Visual C 컴파일러 사용법



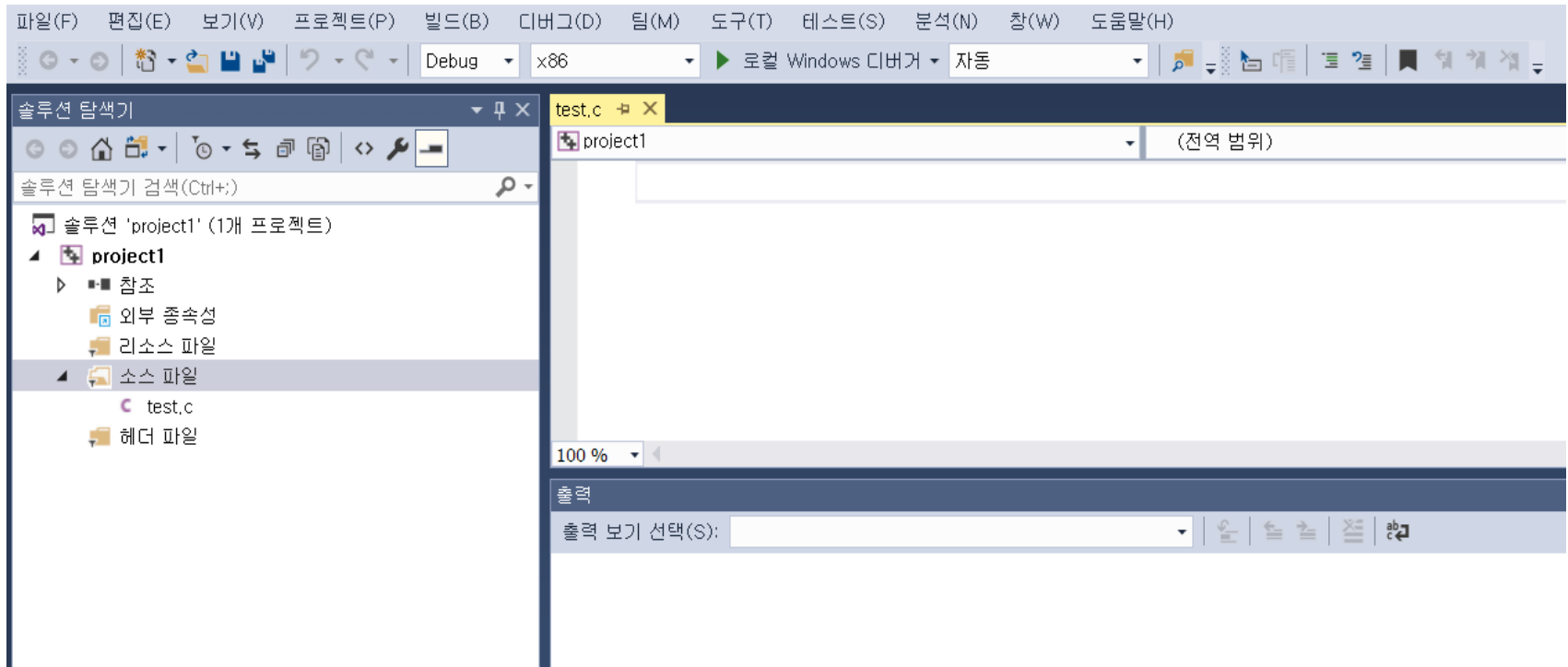
- 소스파일 (우 클릭) → 추가 → 새 항목

Visual C 컴파일러 사용법



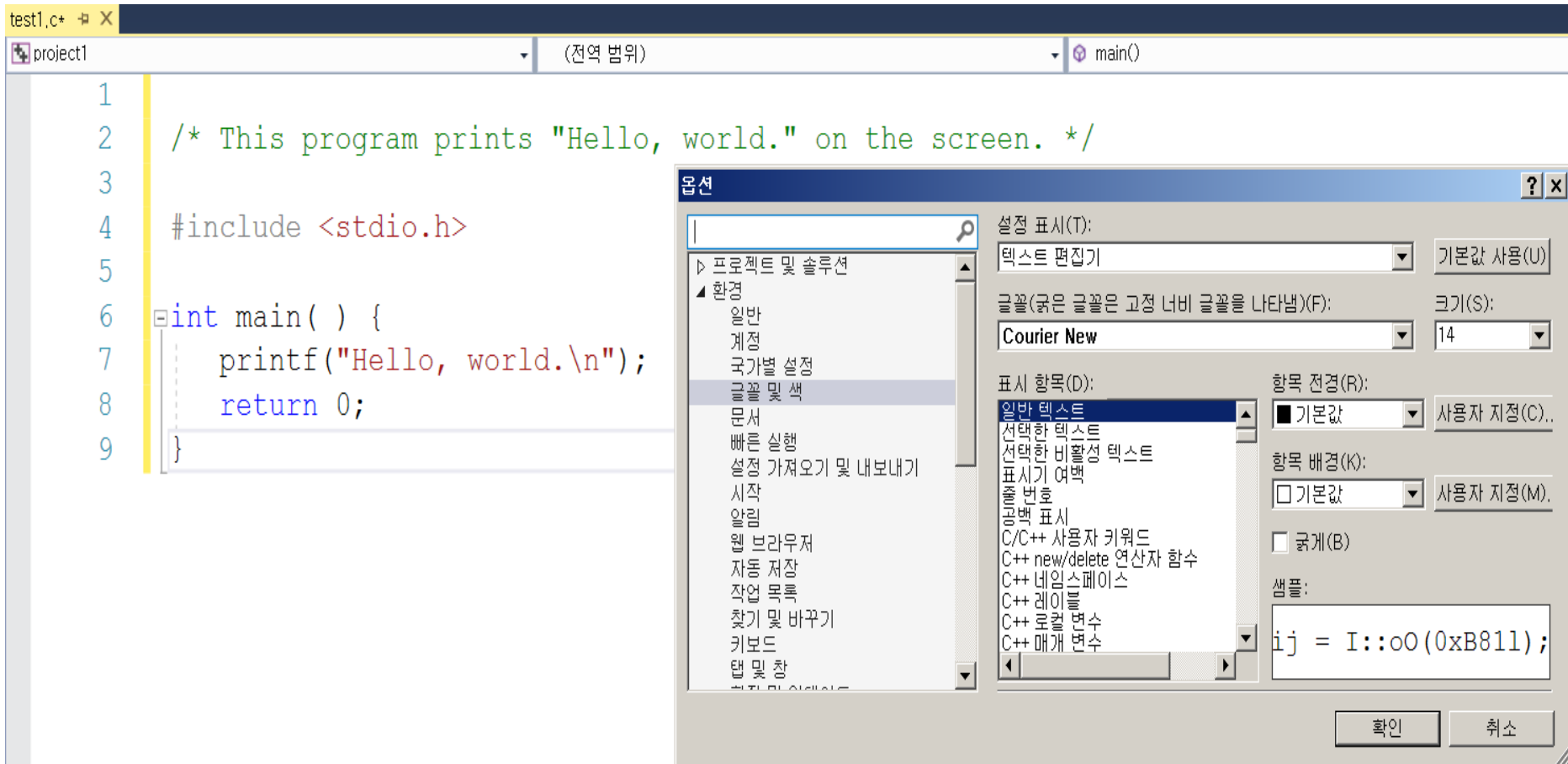
- C 컴파일러는 C++ 컴파일러의 일부분. 그러나 확장자는 반드시 .c로 입력해야 함.

Visual C 컴파일러 사용법



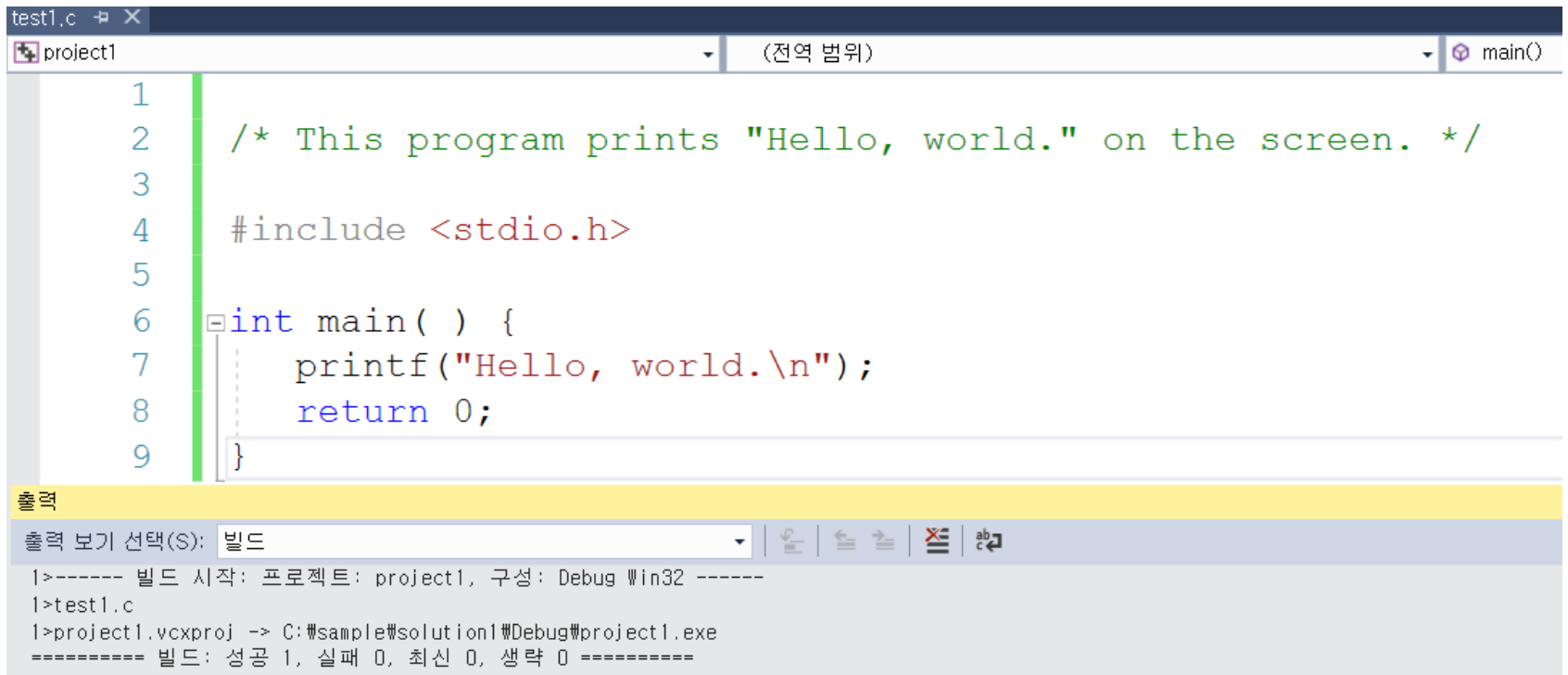
- 세 개의 창
 - 솔루션 탐색기 창, 소스코드 편집 창, 출력 창

Visual C 컴파일러 사용법



- 글꼴 조정은 도구 → 옵션
- 위 소스 코드를 입력하고 **Ctrl-F5**를 누름.

Visual C 컴파일러 사용법



The screenshot displays the Visual Studio IDE. The top pane shows a C program named `test1.c` with the following code:

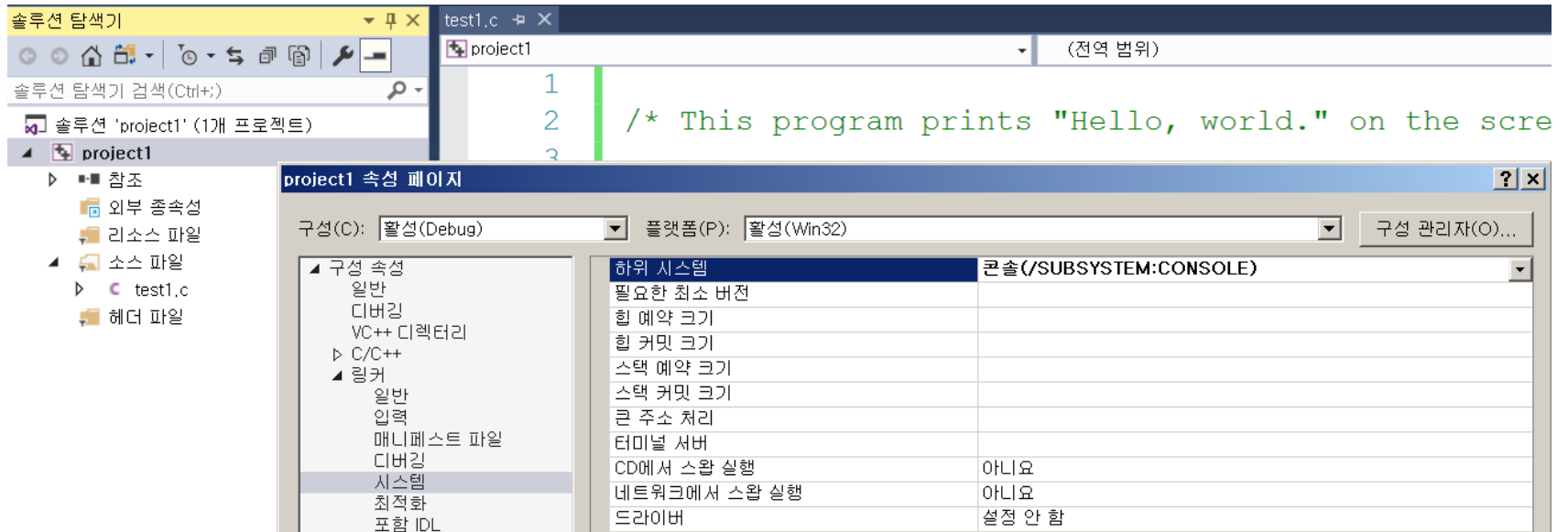
```
1
2  /* This program prints "Hello, world." on the screen. */
3
4  #include <stdio.h>
5
6  int main( ) {
7      printf("Hello, world.\n");
8      return 0;
9  }
```

The bottom pane, titled "출력" (Output), shows the build results for the project:

```
출력 보기 선택(S): 빌드
1>----- 빌드 시작: 프로젝트: project1, 구성: Debug Win32 -----
1>test1.c
1>project1.vcxproj -> C:\sample\solution1\Debug\project1.exe
===== 빌드: 성공 1, 실패 0, 최신 0, 생략 0 =====
```

- **출력 창**에 컴파일 결과가 보임.
- 그러나 실행 결과 **콘솔 창**이 너무 빨리 지나가 버림.

Visual C 컴파일러 사용법



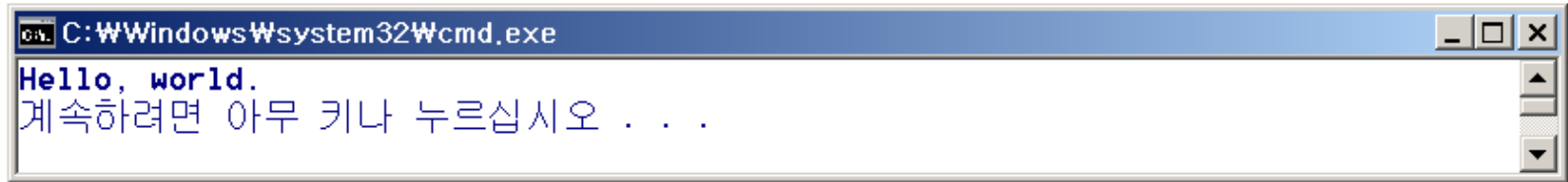
- 콘솔 창을 멈추는 방법
 - 프로젝트 (우 클릭) → 속성 → 링커 → 시스템 → 하위 시스템 → 콘솔을 선택
 - 또는, `system("pause");`

Visual C 컴파일러 사용법



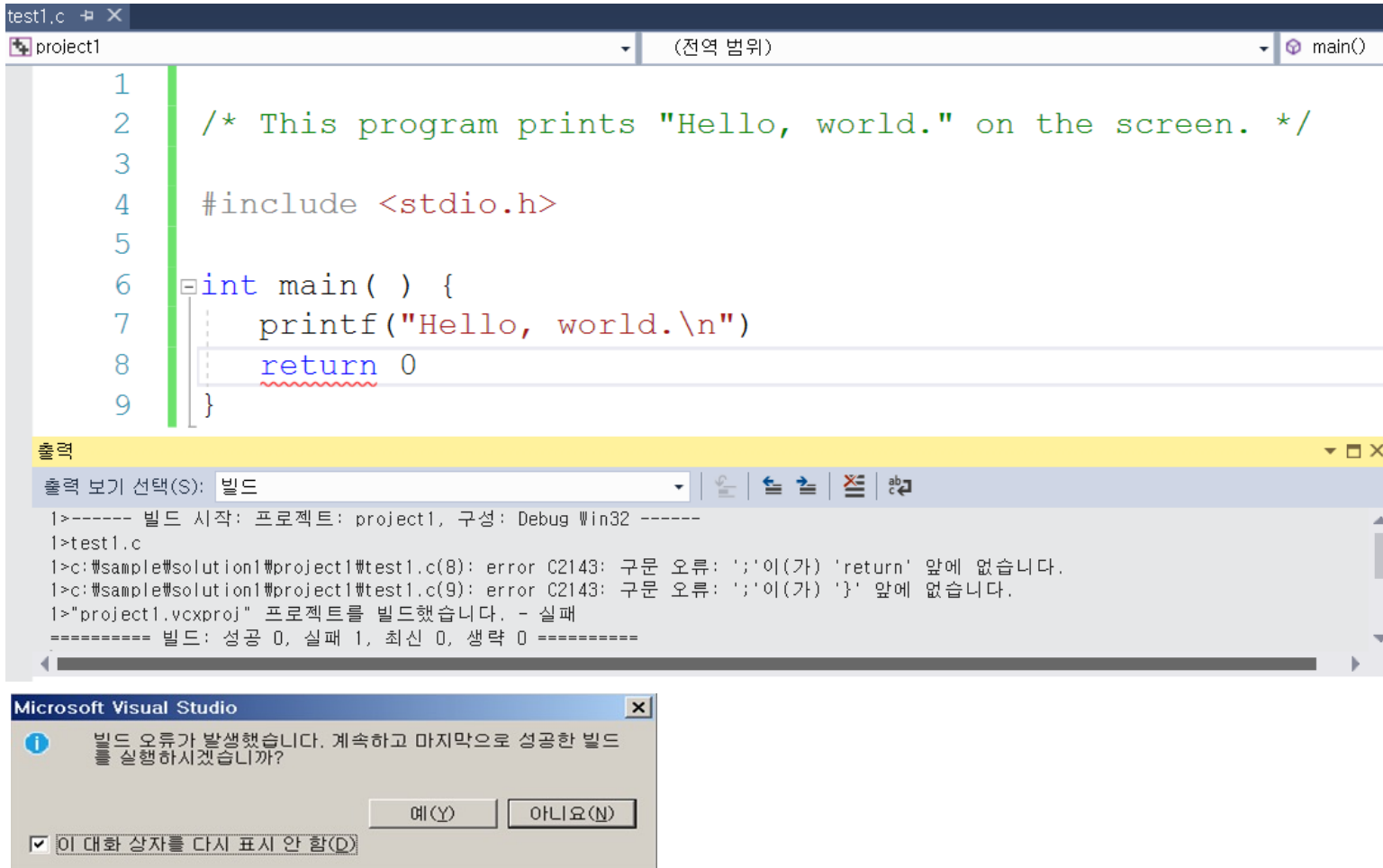
- 프로젝트 속성 변경: 두가지 방법

Visual C 컴파일러 사용법



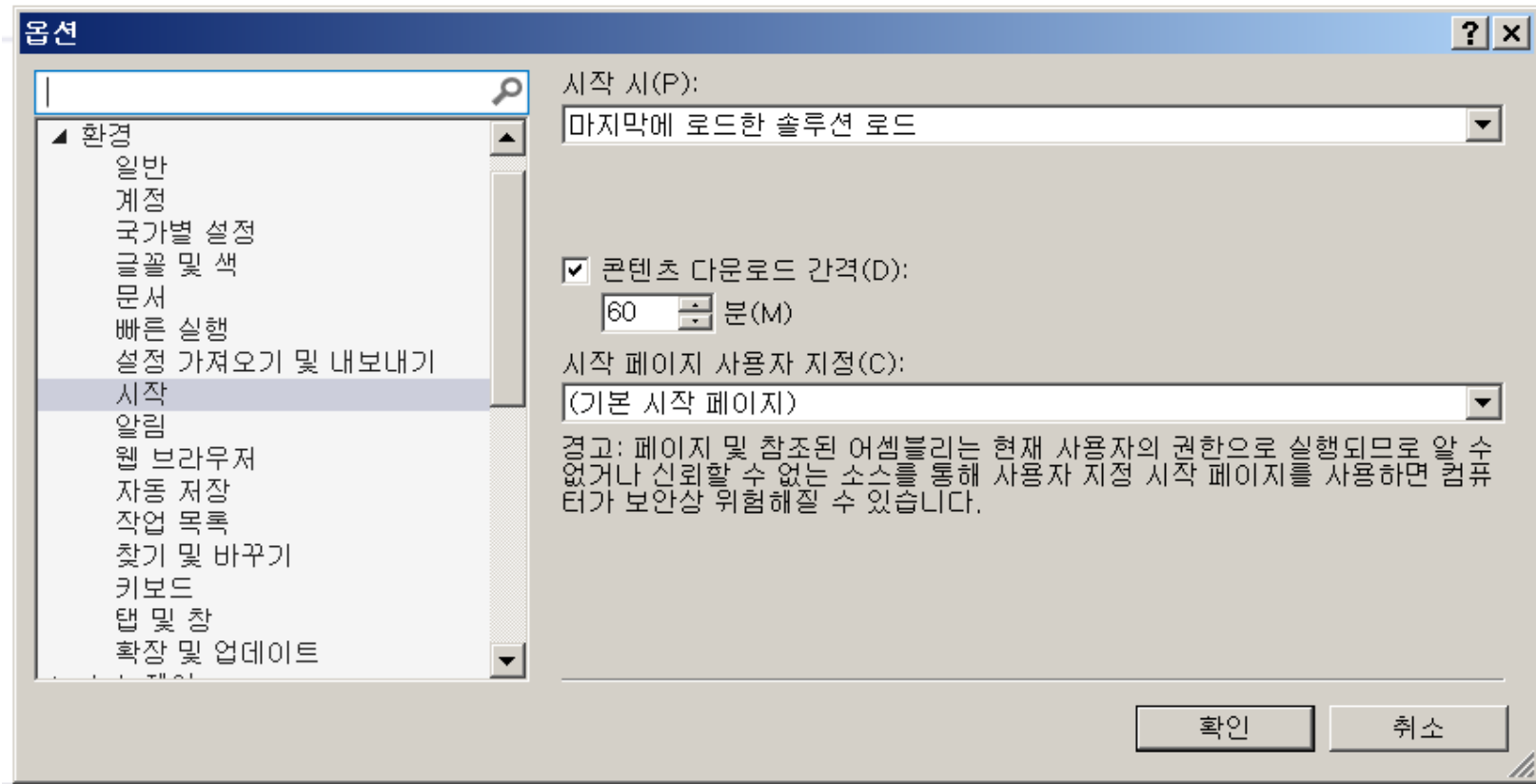
- Ctrl-F5. 샘플 프로그램 실행 결과.

Visual C 컴파일러 사용법



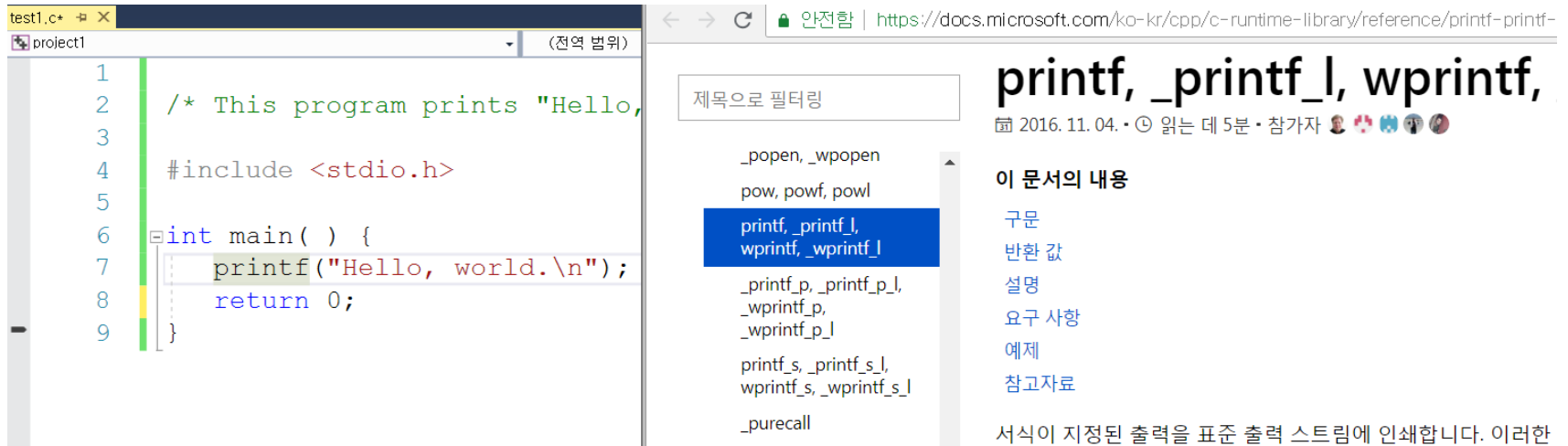
- 세미콜론을 뺀기 때문에 일어난 오류: 회색 박스 및 붉은 줄
- 항상 출력 창의 첫 오류 메시지를 더블 클릭해야 한다.

Visual C 컴파일러 사용법

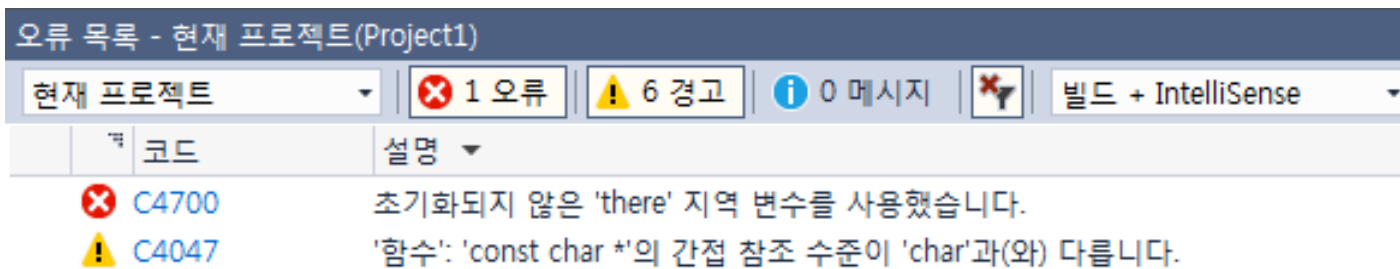


- 프로젝트 별로 매번 속성을 다시 정의하기 귀찮을 경우
 - 도구 → 옵션 → 환경 → 시작 → **마지막에 로드한 솔루션 로드**

Visual C 컴파일러 사용법



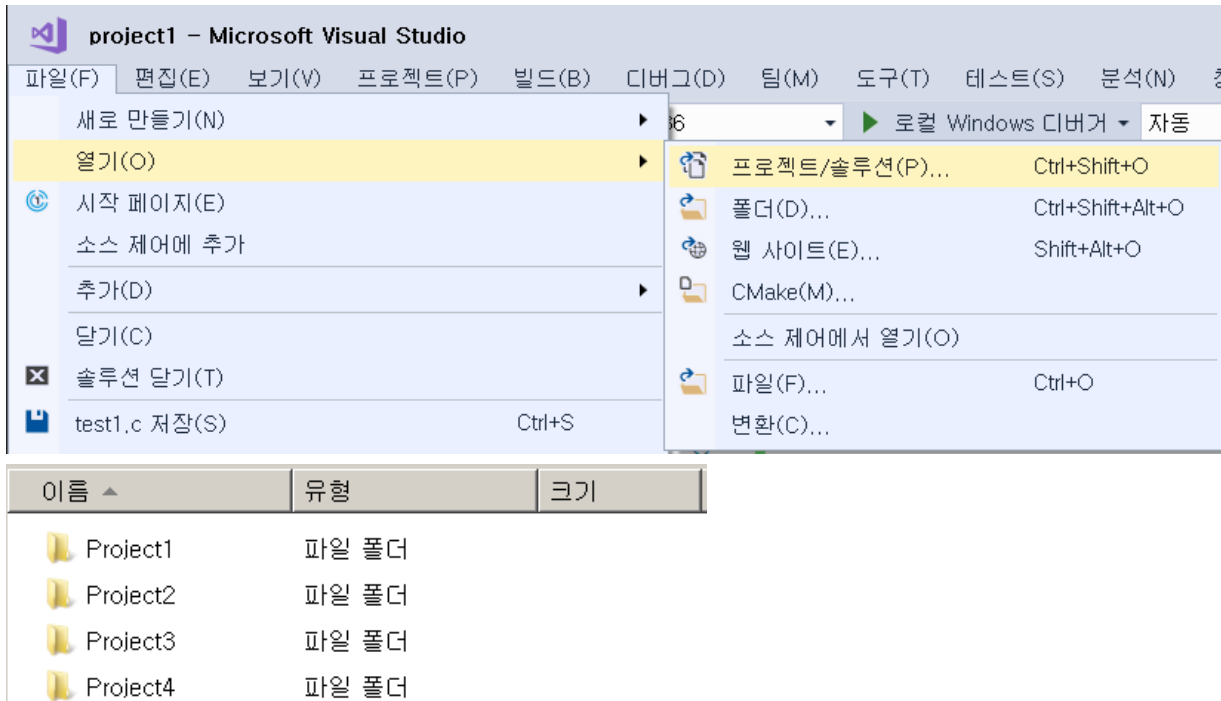
- F1 : 도움말



- 보기 → 오류 목록
 - 오류 메시지와 경고 메시지:

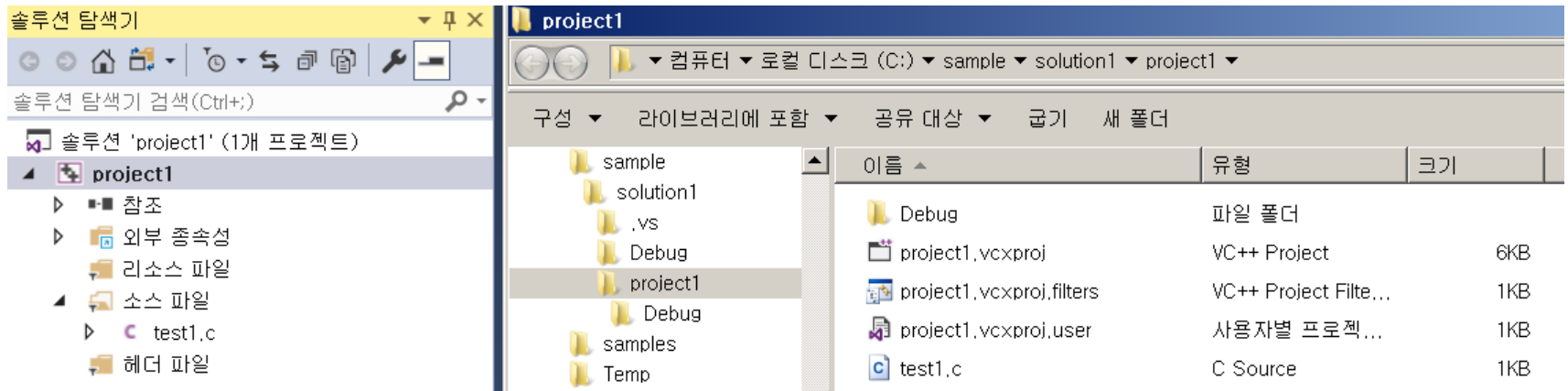
Visual C 컴파일러 사용법

- 파일 -> 모두 저장
 - 소스 코드, 프로젝트, 솔루션 파일을 모두 저장



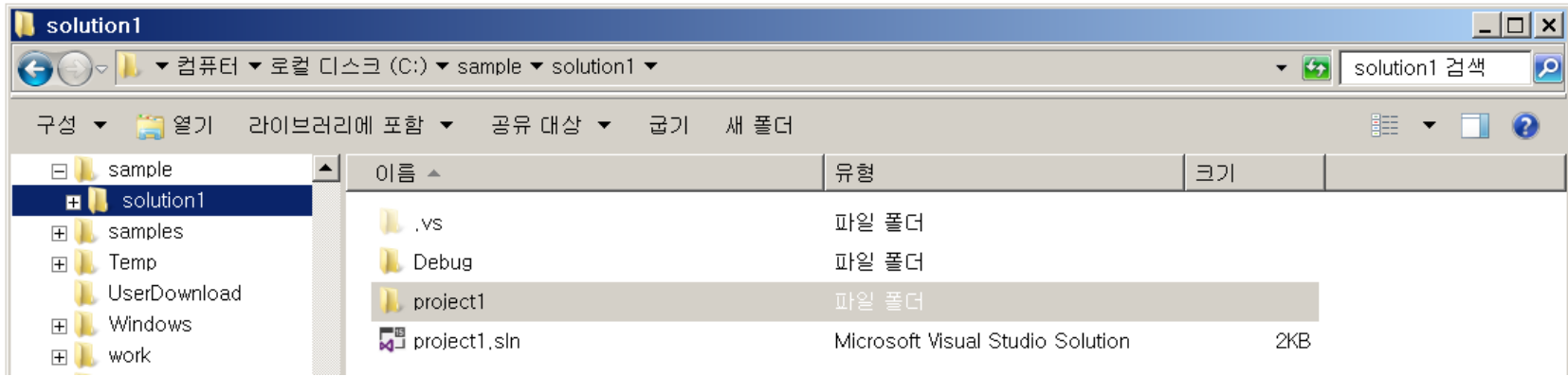
- 이전에 저장한 솔루션 파일 열기
 - 파일 → 열기 → 프로젝트/솔루션 → solution1 폴더 → `project1.sln` 더블 클릭

Visual C 컴파일러 사용법

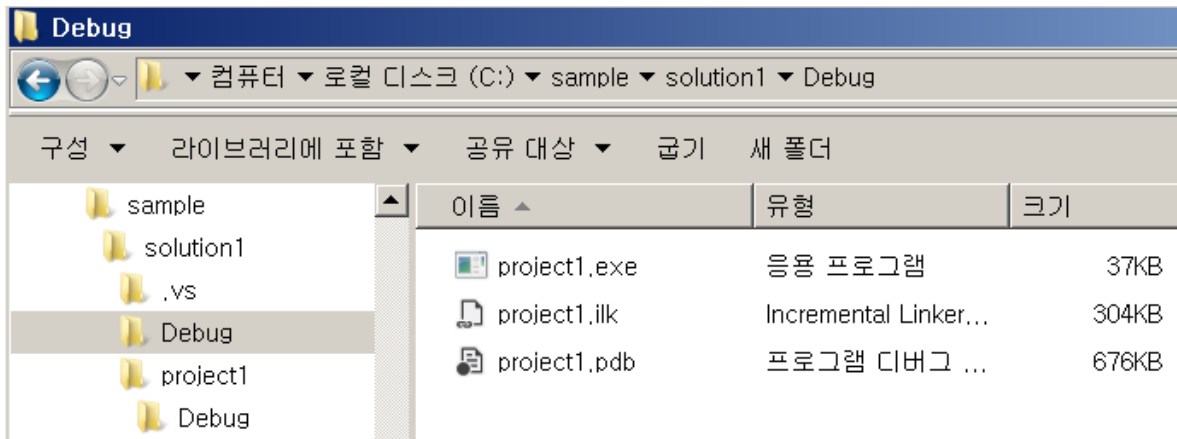


- 소스코드 파일 (.c) 위치 확인
 - 솔루션 탐색기 창의 프로젝트 명을 우 클릭
 - "파일 탐색기에서 폴더 열기" 메뉴를 선택

Visual C 컴파일러 사용법



- 솔루션 파일 (.sln) 위치



- 실행 파일 (.exe) 위치
- 실행 파일명은 소스코드명이 아니라 프로젝트명을 따른다.

샘플 프로그램 해설

```
/* This program prints "Hello, world." on the screen. */ ← 주석
```

```
#include <stdio.h> ← 지시어
```

```
    리턴 타입    함수 명  
    ↓           ↓  
int main( ) {  
    printf("Hello, world.\n"); ← 명령문  
    return 0; ← 명령문  
    }  
    ↑  
    리턴 값
```

타입 매칭

함수 본체(블록)

주석(Comments)을 다는 두 가지 방법

```
/* This program prints "Hello, world." on the screen. */
```

- /* (Slash Asterisk)와 */ (Asterisk Slash) 사이

```
printf("Hello world. \n"); // prints hello message
```

- //(Double Slash)가 시작하는 곳부터 그 줄이 끝날 때까지

지시어(Directives)

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

- **지시어**는 #으로 시작
 - 컴파일러가 번역해야 할 **명령어**가 아님. (세미콜론도 없음)
 - **전 처리기** (Pre-processor)가 처리
 - “여기에 `stdio.h`라는 파일의 내용을 포함시켜 주세요.”
- `stdio.h` (standard input/output header)
 - 표준 입출력을 위한 헤더 파일
 - 표준 입력 장치 = 키보드
 - 표준 출력 장치 = 화면
 - `printf`나 `scanf` 등 입출력 함수가 선언되어 있음.

main 함수

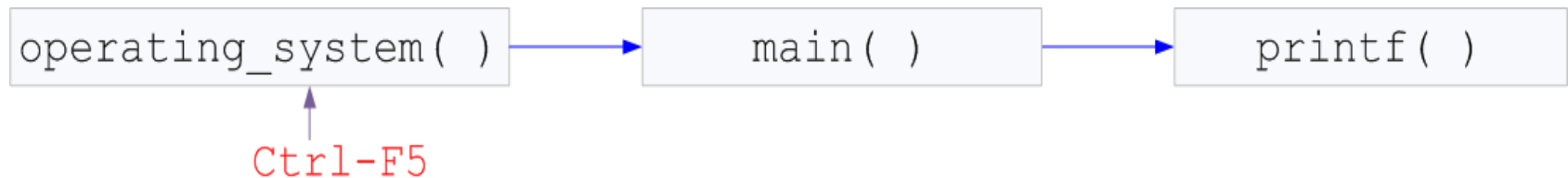
```
int main( ){  
    printf("Hello, world.\n");  
    return 0;  
}
```

- `int`는 `integer`(정수)를 의미
 - 자신을 호출한 함수에게 돌려주는 **리턴 값의 타입**
 - 0을 리턴하고 있으므로 타입이 매칭
 - `return 1.0;` 이라고 했으면 타입 미스매치
- `main`은 함수명 (진입점 함수)
- 중괄호로 둘러싸인 부분이 **함수 본체** (Function Body) 또는 **함수 블록** (Function Block)

함수의 역할

```
icebar get_icebar(money 1000) {  
    take 500 as service charge;  
    go to market;  
    pay 500 and buy a melona;  
    return melona;  
}
```

- 함수는 심부름꾼
 - 건네받은 것이 있을 수도 있고 없을 수도 있음.
 - 돌려주는 것이 있을 수도 있고 없을 수도 있음.



- 연쇄적으로 함수를 호출해야 할 때도 있음.

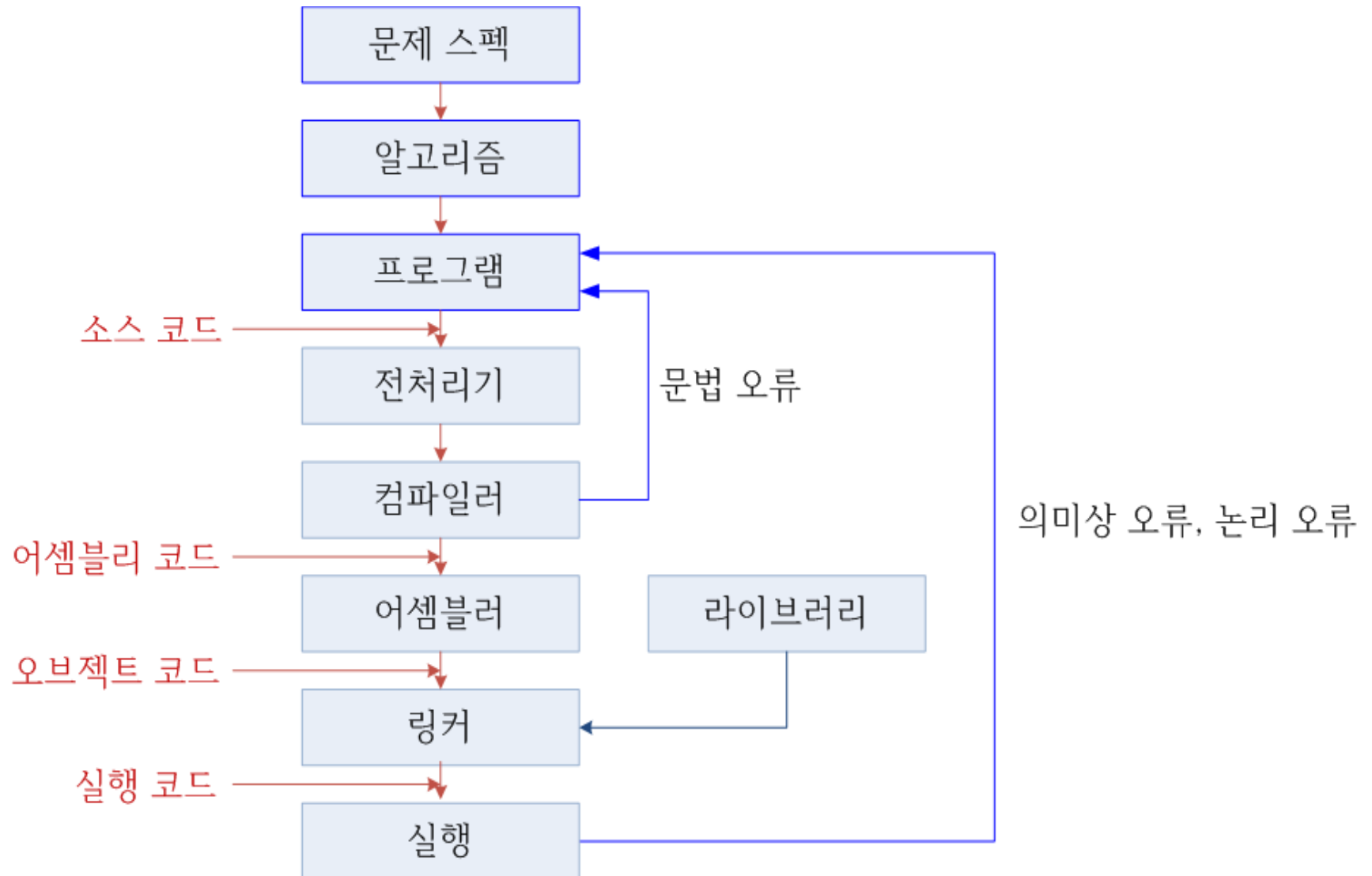
Example 1-1 실습 및 해설

알고리즘

1. 냄비에 적당한량의 물을 담아 가스 불 위에 올려놓는다.
2. 물이 다 끓을 때 찜, 라면 봉지를 뜯어 라면을 넣는다.
3. 대파를 길쭉하게 썰어서 준비한다.
4. 면발이 풀릴 때 찜, 준비해 놓은 대파와 계란을 넣는다.
5. 라면 봉지에 있던 수프를 넣는다.

- 알고리즘 (Algorithms)
 - 문제 해결 방법을 단계적으로 기술한 것.
 - 의사코드 (擬似, Pseudo Code)

프로그램 작성 및 실행 절차

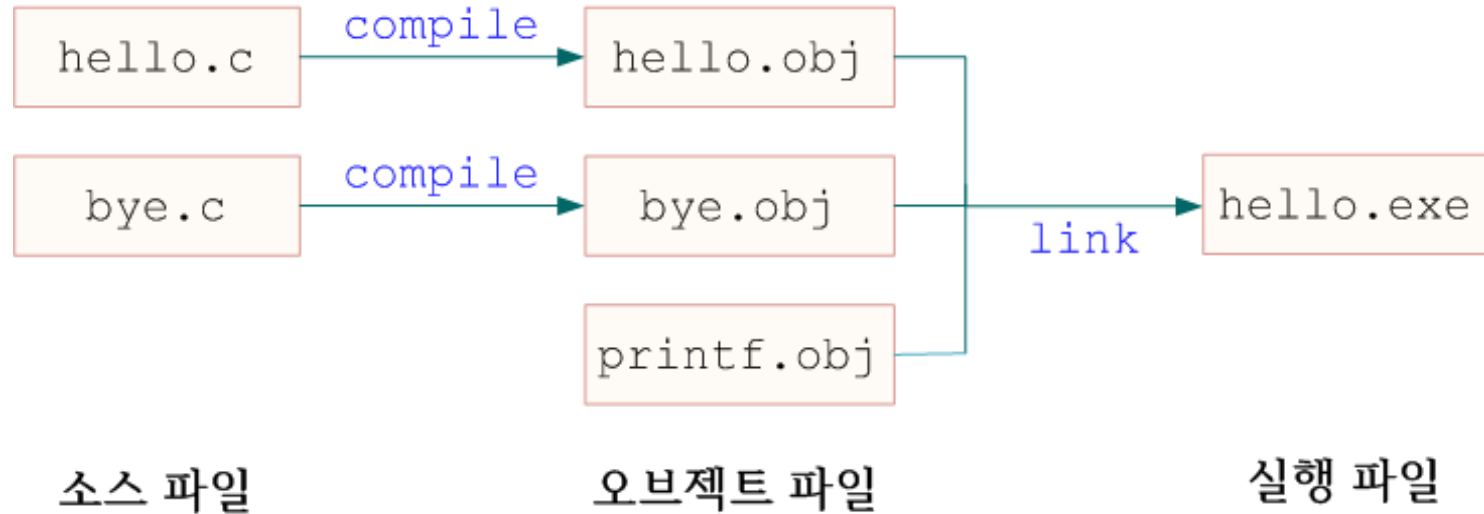


세 가지 오류

- 문법 오류, 의미상 오류, 논리 오류 (알고리즘 오류)
- 논리 오류의 예:
 - 소득이라는 단어가 들어간 문장은 몇 개인가?
 - 알고리즘: 소득이라는 단어 이후에 마침표이면 한 문장.
 - 그러나, 아래 입력에 대해 이 알고리즘은 오류

소득이 부진했던 지난 해 경제 성장률이 외환위기 이후 최저인 3.1%에 불과한데, 국민 소득이 10.1%가 늘어나게 된 이유는 국민 소득의 개념 및 계산 방법의 차이에 있다.

코드 형태 변화



- 소스 코드 → 오브젝트 코드 → 실행 코드
- 오브젝트 파일 및 실행 파일은 2진 파일
- 표준 헤더 파일에 선언된 함수는 **오브젝트 코드 형태**로 저장되어 있음. (WHY?)

링커의 역할

함수 명	주소	오브젝트 코드
printf()	219	11010001

	200	00110001
main()	199	00101101
	...	
	164	01000001
	160	00101001 = printf()
	...	
	100	10011111

- 연결 정보를 오브젝트 코드에 삽입하여 실행 파일을 생성.

Related URL's

- How Software is Made
 - <https://youtu.be/bWdeGTJxMQc>
- Coding is not difficult | Mark Zuckerberg
 - <https://youtu.be/F9GujgK0y2M>
- Why C Programming is awesome
 - <https://www.youtube.com/watch?v=smGalmxPVYc>
- Introduction To C-Teaser Video
 - <https://youtu.be/tsn11i2tpE4>
- Evolution Of C-Teaser Video
 - https://youtu.be/vm_lQeMSurc

Related URL's

- Introduction to C Programming Language | History | Why Study It | Video Tutorial for Beginners
 - <https://www.youtube.com/watch?v=6VT8hDr2GhU&t=244s>
- Introduction to programming and programming languages: C Programming Tutorial 0
 - <https://www.youtube.com/watch?v=AWliApDc6lw1>
- Introduction To Programming Language-Teaser Video
 - <https://youtu.be/QaM5-joItPk>
- Introduction To Compiler And Compiling-Teaser Video
 - <https://youtu.be/M14jeJGF32Y>
- Introduction To Compiler And Compiling - Computer Programming - Basics
 - <https://youtu.be/jTdY0hh5mTM>

Related URL's

- Introduction to C Language
 - https://youtu.be/Bkqkt72_Z_g
- Introduction to programming and programming languages:
C Programming Tutorial 01
 - https://www.youtube.com/watch?v=AWliApDc61w&list=PL2_aWCzGMAwLSqGsERZGXGkA5AfMhcknE
- What is an Algorithm?
 - https://youtu.be/e_WfC8HwVB8
- BBC Learning - What Is An Algorithm
 - <https://youtu.be/Da5TOXCwLSq>
- Library Files-Teaser Video
 - <https://youtu.be/ymFBQn5hY9k>



감사합니다.