

CHAPTER

# 02

## 프로그래밍 작성 과정

- 프로그램이 개발되는 과정을 이해할 수 있다
- 비주얼 스튜디오를 사용할 수 있다
- 화면에 “Hello World!” 를 출력하는 예제 프로그램을 이해하고 실행할 수 있다.

# Contents

3

2.1

프로그램 개발 과정

2.2

통합 개발 환경

2.3

비주얼 스튜디오 설치

2.4

비주얼 스튜디오 사용하기

2.5

예제 프로그램의 간략한 설명

2.6

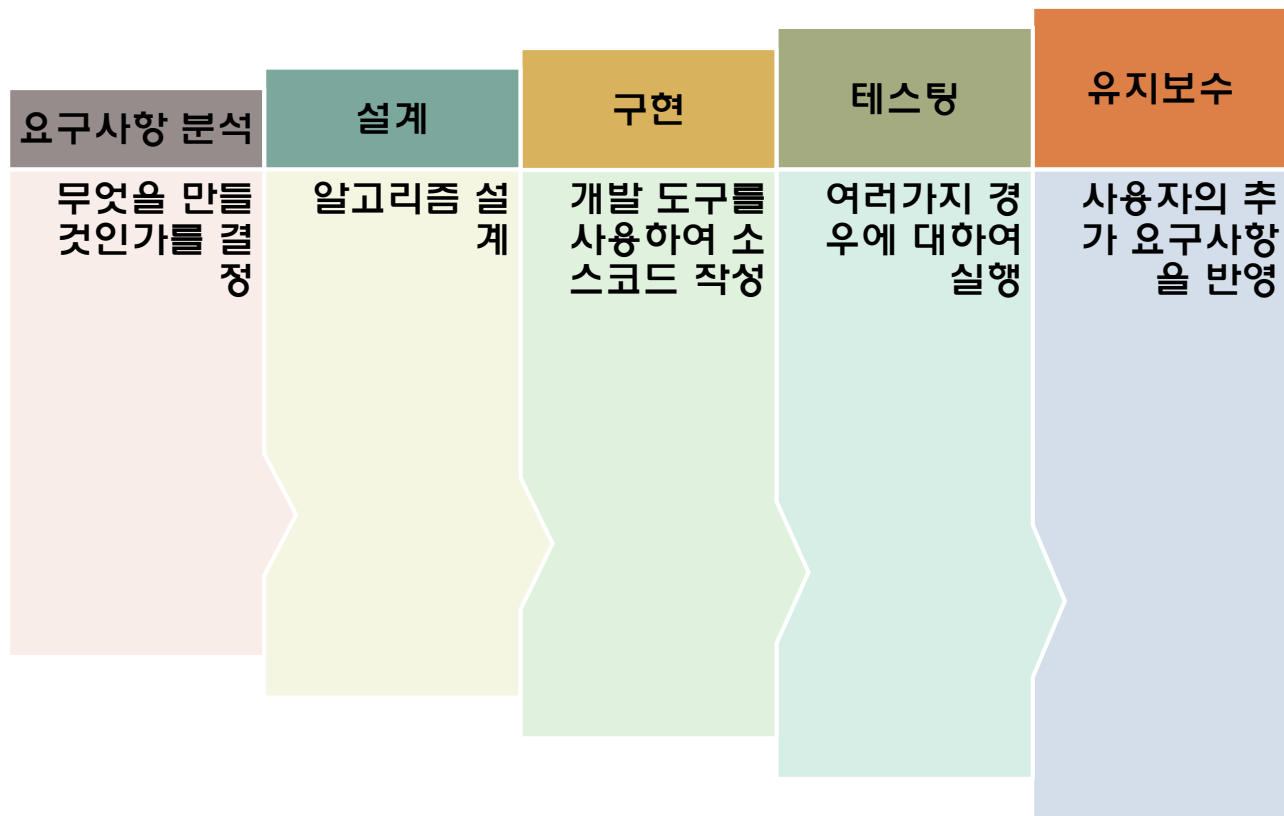
예제 프로그램의 응용

2.7

오류수정

# 프로그램 개발 과정

4



# 요구 사항 분석

5

- 프로그래머는 사용자들의 요구사항을 만족시키기 위하여 프로그램을 작성
- (예) 3년 이상 근무한 직원들의 리스트 출력
  - ▣ 정규직만 or 계약직 포함
  - ▣ 기준이 되는 날짜가 오늘?
- 요구 사항 명세서: 사용자의 요구 조건을 만족하도록 소프트웨어가 갖는 기능 및 제약 조건, 성능 목표 등을 포함

# 설계

6

- 문제를 해결하는 알고리즘 개발하는 단계
- 어떤 단계를 밟아서 어떤 순서로 작업을 처리할 것인지를 설계
- 순서도와 의사 코드를 도구로 사용
- 알고리즘은 프로그래밍 언어와는 무관
- 알고리즘은 원하는 결과를 얻기 위하여 밟아야 하는 단계에 집중적으로 초점을 맞추는 것

# 소스 작성

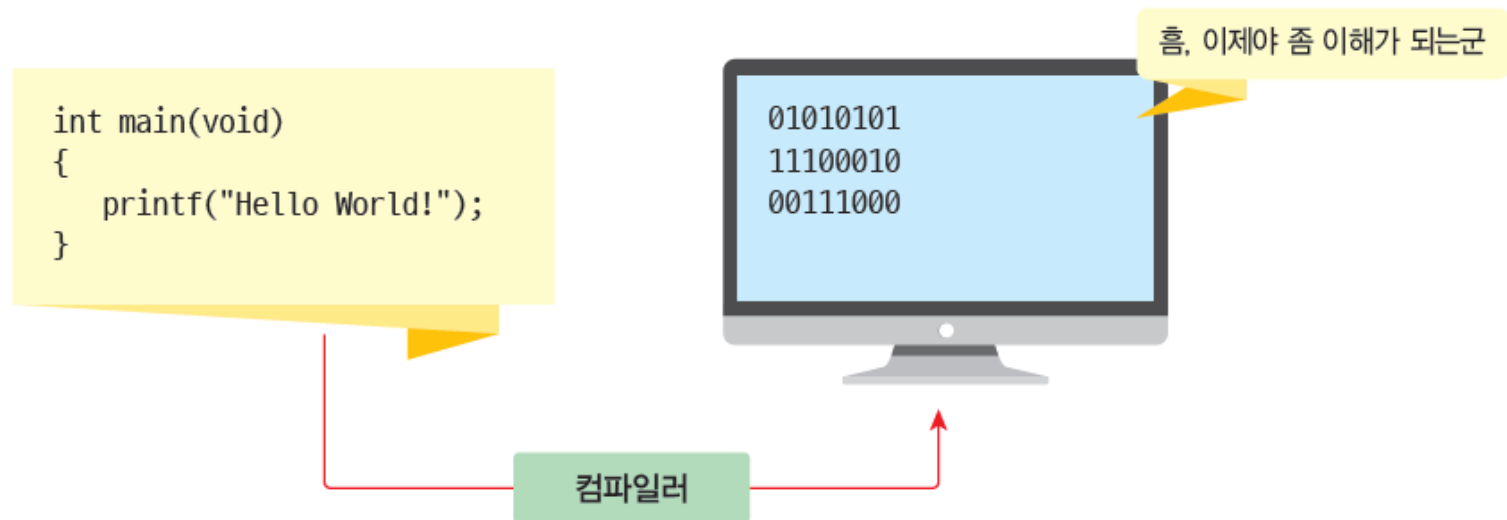
7

- 알고리즘의 각 단계를 프로그래밍 언어를 이용하여 기술
- 알고리즘을 프로그래밍 언어의 문법에 맞추어 기술한 것을 *소스 프로그램(source program)*
- 소스 프로그램은 주로 텍스트 에디터나 통합 개발 환경을 이용하여 작성
- 소스 파일 이름: (예) test.c

# 컴파일

8

- 소스 프로그램을 오브젝트 프로그램으로 변환하는 작업
- 오브젝트 파일 이름: (예) test.obj
- 컴파일러가 수행

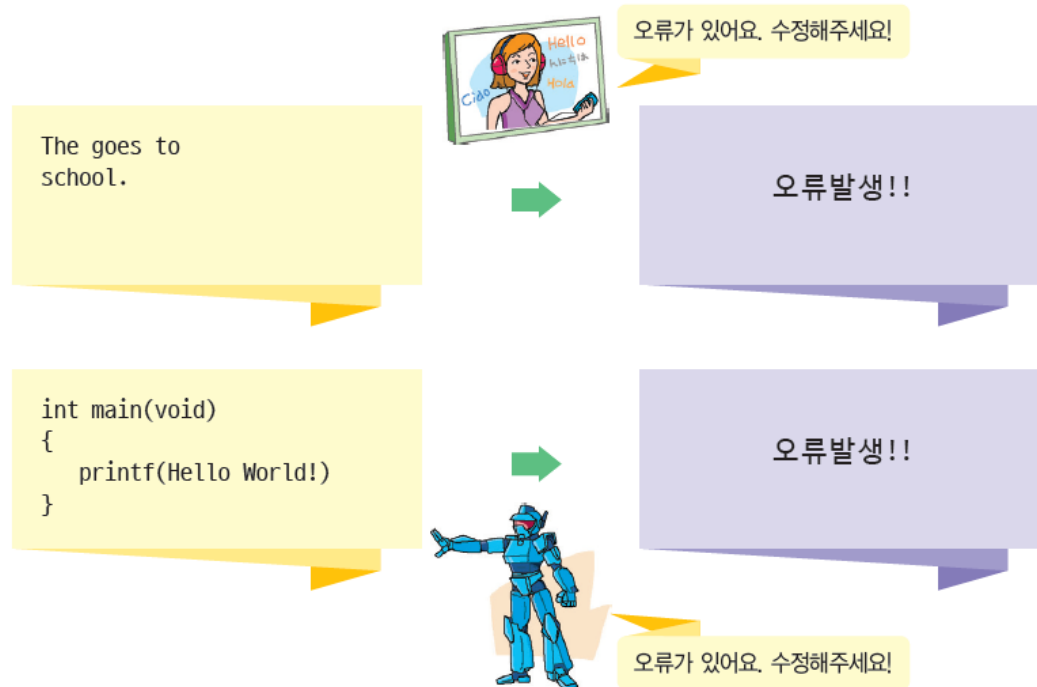




# 컴파일 오류

9

- 컴파일 오류(compile error): 문법 오류
  - ▣ (예) He go to school;
  - ▣ 오류가 발생하면 소스 프로그램을 수정한 후에 다시 컴파일



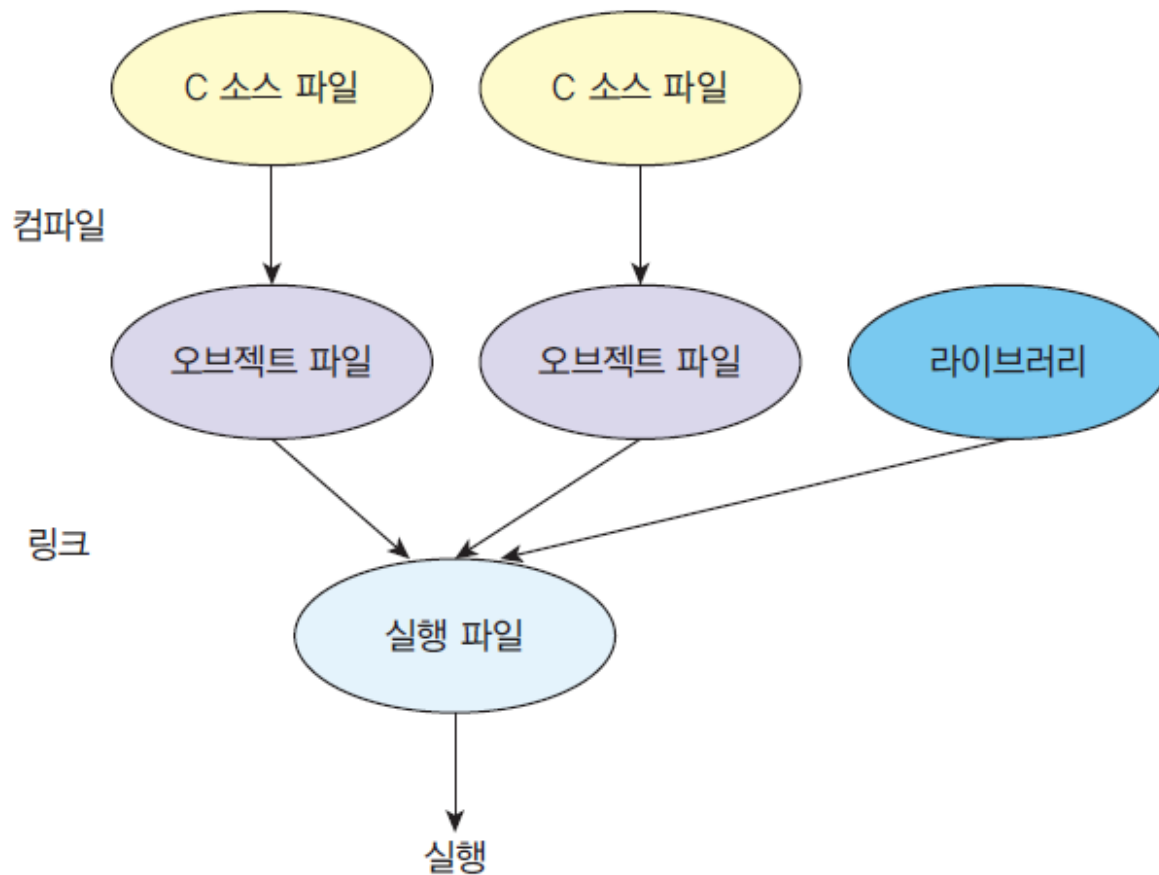
# 링크

10

- 컴파일된 목적 프로그램을 라이브러리와 연결하여 실행 프로그램을 작성하는 것
- 실행 파일 이름: (예) test.exe
- *라이브러리(library)*: 프로그래머들이 많이 사용되는 기능을 미리 작성해 놓은 것
  - ▣ (예) 입출력 기능, 파일 처리, 수학 함수 계산
- 링크를 수행하는 프로그램을 *링커(linker)*라고 한다.

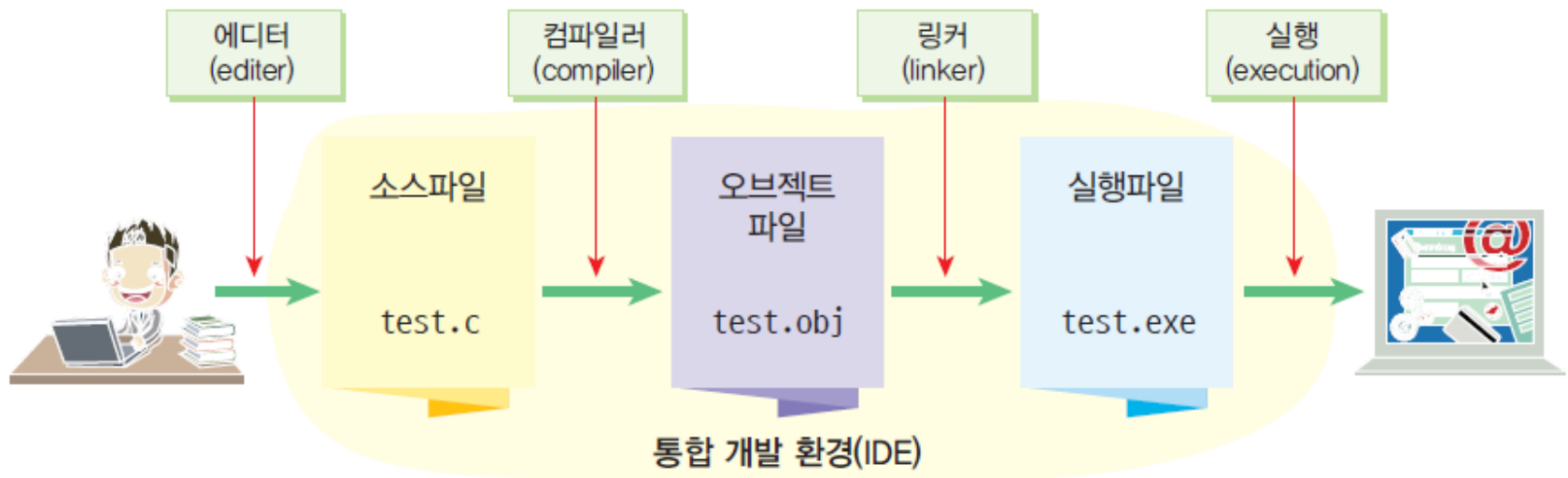
# 링크

11



# 링크

12



# 실행 및 디버깅

13

- 실행 시간 오류(run time error):
  - ▣ (예) 0으로 나누는 것
  - ▣ 잘못된 메모리 주소에 접근하는 것
- 논리 오류(logical error):
  - ▣ 문법은 틀리지 않았으나 논리적으로 정확하지 않는 것
  - ▣ (예)

- ① 그릇1과 그릇2를 준비한다.
- ② 그릇1에 밀가루, 우유, 계란을 넣고 잘 섞는다.
- ③ 그릇2를 오븐에 넣고 30분 동안 350도로 굽는다.



# 디버깅

14

- 소스에 존재하는 오류를 잡는 것



# 디버깅의 유래

15

- 1945년 마크 II 컴퓨터가 릴레이 장치에 날아든 나방 때문에 고장을 일으켰고 이것을 “컴퓨터 버그(bug: 벌레)”라고 불렀다. 여성 컴퓨터 과학자인 그레이스 호퍼가 나방을 채집해 기록에 남기고 이를 “디버깅(debugging)” 작업이라고 보고하였다



# 소프트웨어의 유지 보수

16

- 소프트웨어의 유지 보수가 필요한 이유
  - ▣ 디버깅 후에도 버그가 남아 있을 수 있기 때문
  - ▣ 소프트웨어가 개발된 다음에 사용자의 요구가 추가될 수 있기 때문
- 유지 보수 비용이 전체 비용의 50% 이상을 차지



# Contents

17

2.1

프로그램 개발 과정

2.2

통합 개발 환경

2.3

비주얼 스튜디오 설치

2.4

비주얼 스튜디오 사용하기

2.5

예제 프로그램의 간략한 설명

2.6

예제 프로그램의 응용

2.7

오류수정

# 통합 개발 환경

18

- 통합 개발 환경(IDE: integrated development environment)
  - ▣ 에디터 + 컴파일러 + 디버거



# 통합 개발 환경의 예

19

- 비주얼 스튜디오: 마이크로소프트
- 이클립스(eclipse): 오픈 소스 프로젝트
- Dev-C++: 오픈 소스 프로젝트



# Contents

20

2.1

프로그램 개발 과정

2.2

통합 개발 환경

2.3

비주얼 스튜디오 설치

2.4

비주얼 스튜디오 사용하기

2.5

예제 프로그램의 간략한 설명

2.6

예제 프로그램의 응용

2.7

오류수정

# 비주얼 스튜디오 버전

21

- 커뮤니티(Visual Studio Community) 버전은 “기업 외 응용 프로그램 빌드 개발자를 위한 완벽한 기능의 확장 가능한 무료 도구” 이다.
- 프로페셔널 버전(Visual Studio Professional)은 “개별 개발자 또는 소규모 팀을 위한 전문적인 개발자 도구 및 서비스” 라고 되어 있다.
- 엔터프라이즈 버전(Visual Studio Enterprise)은 “고급 테스트 및 DevOps를 포함해서 어떠한 크기나 복잡한 프로젝트까지 개발 팀을 위한 고급 기능이 포함된 엔터프라이즈급 솔루션” 라고 표시되어 있다.

# 비주얼 C++ 설치

22

□ <http://www.visualstudio.com/ko/>

The screenshot shows the Visual Studio website in Korean. The header includes the Microsoft logo and navigation links for Visual Studio, Downloads, Sign up, Support, and Developer tools. A banner for Mac용 Visual Studio is visible, dated February 24, 2020. The main heading is 'Visual Studio' with the subtitle '모든 개발자를 위한 최상의 도구' (The best tool for all developers). Below this, there are three main sections: Visual Studio, Visual Studio Code, and Visual Studio for Mac. Each section features a screenshot of the IDE, a brief description of its capabilities, and a download link. The Visual Studio section also includes a dropdown menu for different editions: Community 2019, Professional 2019, and Enterprise 2019. The footer mentions '개발자 서비스' (Developer services).

Visual Studio IDE 코드 편집기, +

visualstudio.microsoft.com/ko/

Microsoft | Visual Studio 제품 다운로드 구입 지원 구독자 액세스 무료 Visual Studio Microsoft 전체

Mac용 Visual Studio: 새로 고침(); 2020년 2월 24일

macOS에서 .NET을 사용한 웹, 모바일 및 게임을 개발하는 방법에 관한 종일 라이브 스트리밍에 온라인으로 참여하세요. 자세한 정보 >

일정 알림 >

## Visual Studio

모든 개발자를 위한 최상의 도구

### Visual Studio

코딩, 디버그, 테스트 및 모든 플랫폼에 배포할 수 있는 완전한 기능을 갖춘 IDE

Visual Studio 다운로드

- Community 2019
- Professional 2019
- Enterprise 2019

### Visual Studio Code

모든 OS 편집 및 디버깅  
Visual Studio Code를 사용하면 다음에 동의하는 것입니다. (라이선스 및 개인정보처리방침)

Download Visual Studio Code

자세히 보기 >

### Visual Studio for Mac

.NET을 사용하여 iOS, Android 및 웹 앱 및 게임 개발  
라이선스 활성화에 대한 자세한 정보

Visual Studio for Mac 다운로드

자세히 보기 >

개발자 서비스

# 비주얼 C++ 설치

23

Visual Studio를 다운로드해 주셔서 감사합니다.  
다운로드가 곧 시작됩니다. 다운로드가 시작되지 않은 경우에는 [여기를 클릭하여 다시 시도하세요](#)

## Visual Studio를 사용하여 개발 시작

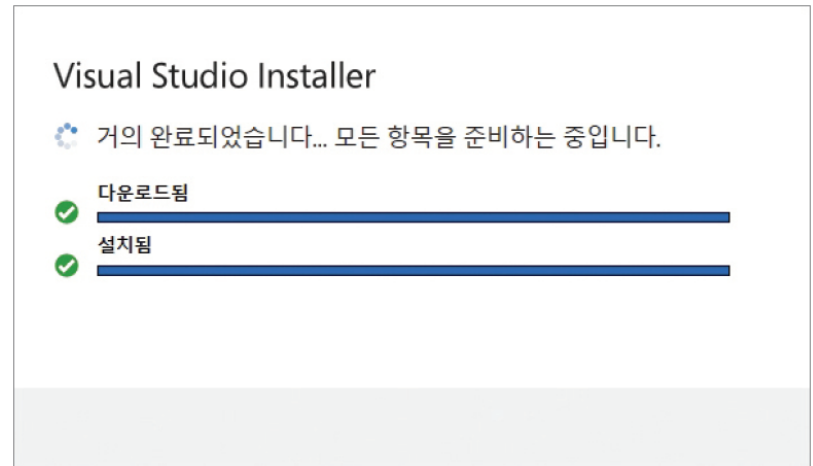
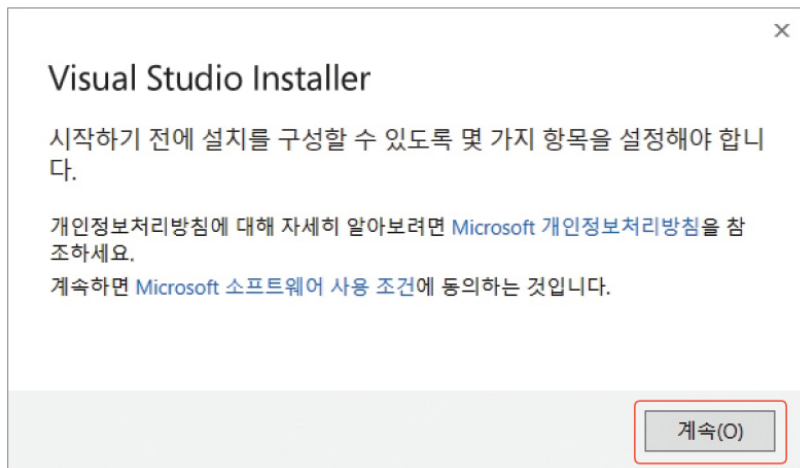
**설치 지침**  
설치에 대한 도움이 필요하세요? Visual Studio 설치의 도움말 설명서를 확인하세요.  
[지침 >](#)

**빠른 시작 가이드**  
C++ 콘솔을 사용하여 첫 번째 애플리케이션을 개발하는 방법에 대한 자습서를 사용해 보세요.  
[계산기 앱 자습서 >](#)

vs\_community\_7...exe

# 비주얼 C++ 설치

24





# 비주얼 C++ 설치


25


Visual Studio Installer


설치 중 — Visual Studio Community 2017 — 15.9.6

워크로드    개별 구성 요소    언어 팩    설치 위치


Windows (3)


 .NET 데스크톱 개발  
C#, Visual Basic 및 F#를 사용하여 WPF, Windows Forms 및 콘솔 응용 프로그램을 빌드합니다.


 C++를 사용한 데스크톱 개발  
Microsoft C++ 도구 집합, ATL 또는 MFC를 사용하여 Windows 데스크톱 응용 프로그램을 빌드합니다.


 유니버설 Windows 플랫폼 개발  
C#, VB, JavaScript 또는 선택적으로 C++를 사용하여 유니버설 Windows 플랫폼용 응용 프로그램을 만듭니다.

웹 및 클라우드 (7)

 ASP.NET 및 웹 개발  
Docker 지원이 포함된 ASP.NET, ASP.NET Core, HTML/JavaScript 및 컨테이너를 사용하여 웹 응용 프로그램...

 Azure 개발  
클라우드 앱 개발, 리소스 생성, Docker 지원 등 컨테이너를 빌드하기 위한 Azure SDK, 도구 및 프로젝트입니다.

 Python 개발  
Python에 대한 편집, 디버깅, 대화형 개발 및 소스 제어입니다.

 Node.js 개발  
비동기 이벤트 구동 JavaScript 런타임인 Node.js를 사용하여 확장 가능한 네트워크 응용 프로그램을 빌드합니다.

위치

C:\Program Files (x86)\Microsoft Visual Studio\2017\Community    변경...

필요한 총 공간    622MB

계속하면 선택한 Visual Studio 버전에 대한 [라이선스](#)에 동의하는 것입니다. Microsoft는 Visual Studio와 함께 다른 소프트웨어를 다운로드할 수 있는 기능도 제공합니다. 이 소프트웨어는 [타사 고지 사항](#) 또는 해당 라이선스에 명시된 대로 별도로 사용이 허가됩니다. 계속하면 이러한 라이선스에도 동의하는 것입니다.

다운로드하는 동안 설치    설치

설치 세부 정보

Visual Studio 핵심 편집기

구문 인식 코드 편집, 소스 코드 제어, 작업 항목 관리 등을 포함하는 Visual Studio 코어 셸 환경입니다.

1.18.1095.110

# Contents

26

2.1

프로그램 개발 과정

2.2

통합 개발 환경

2.3

비주얼 스튜디오 설치

2.4

비주얼 스튜디오 사용하기

2.5

예제 프로그램의 간략한 설명

2.6

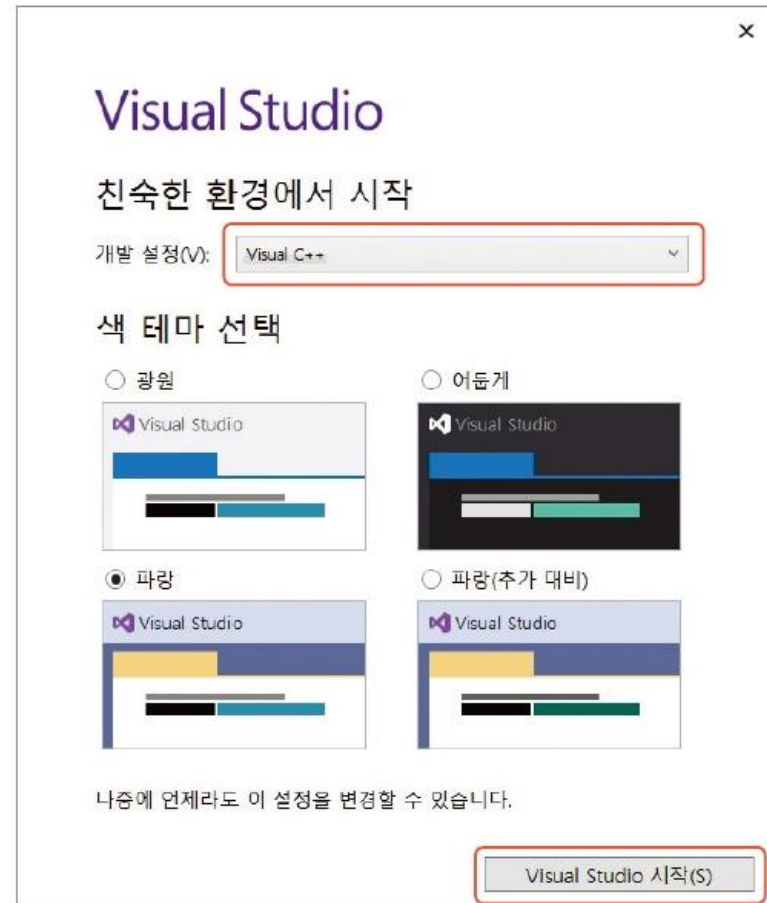
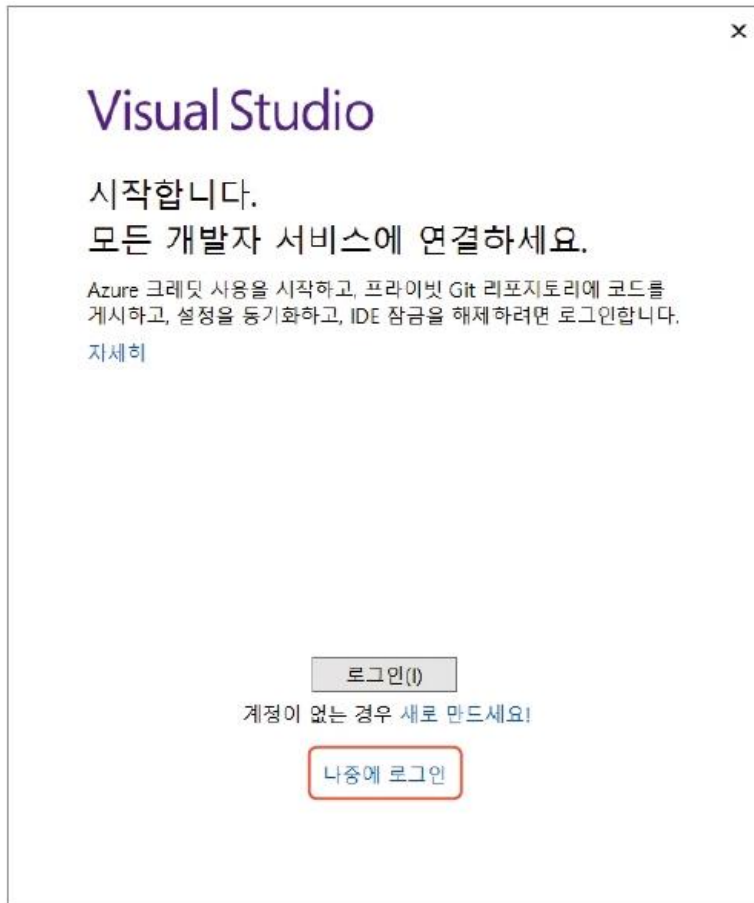
예제 프로그램의 응용

2.7

오류수정

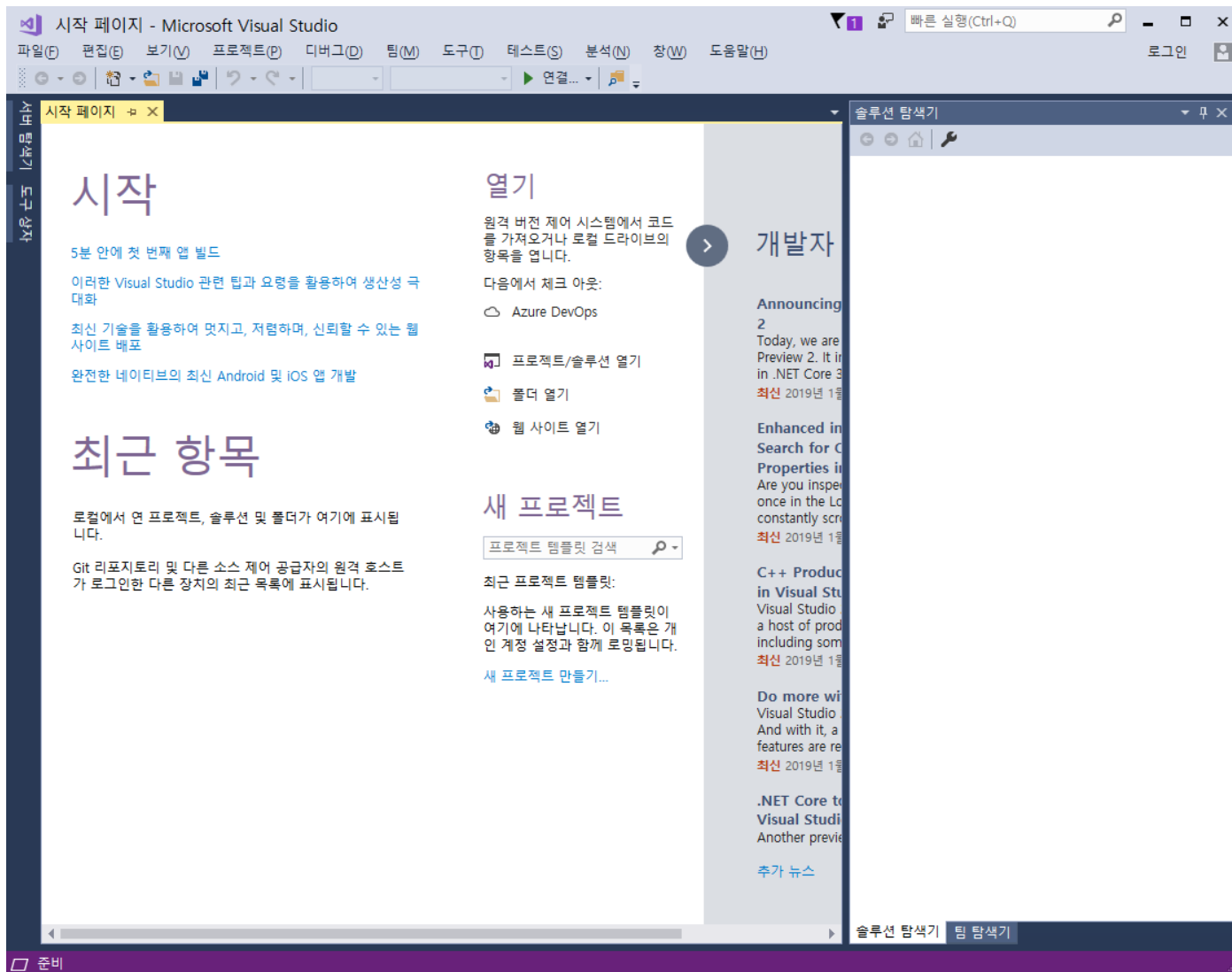
# 비주얼 스튜디오 시작

27



# 비주얼 스튜디오 시작

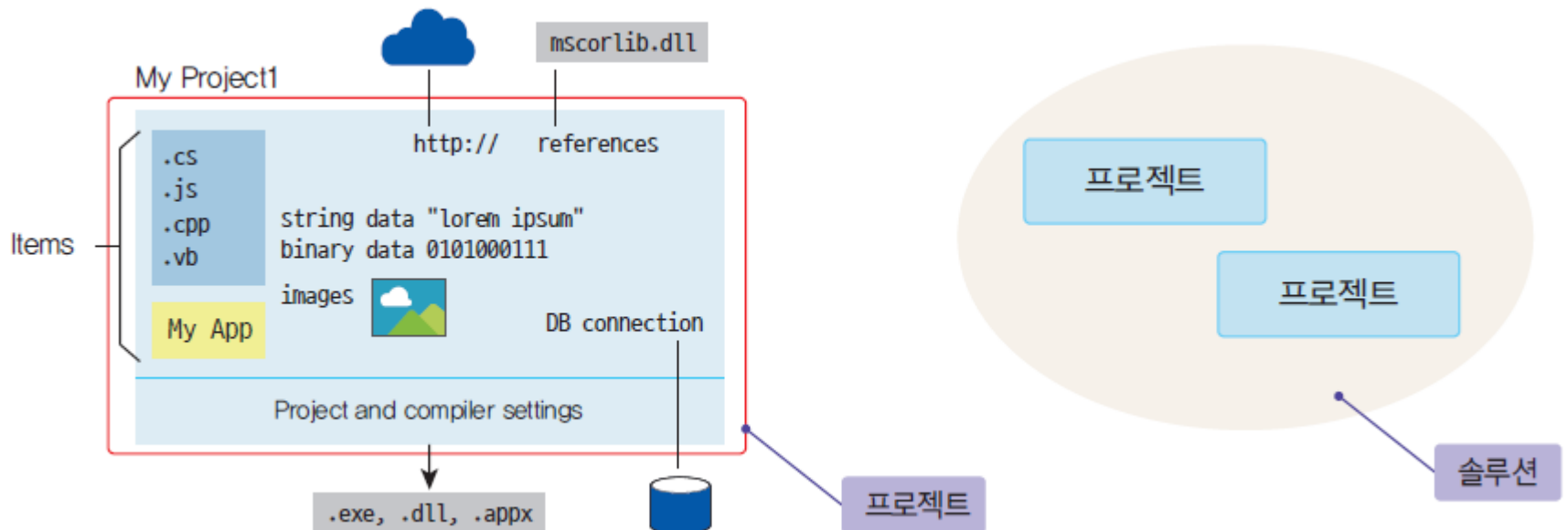
28



# 워크스페이스와 프로젝트

29

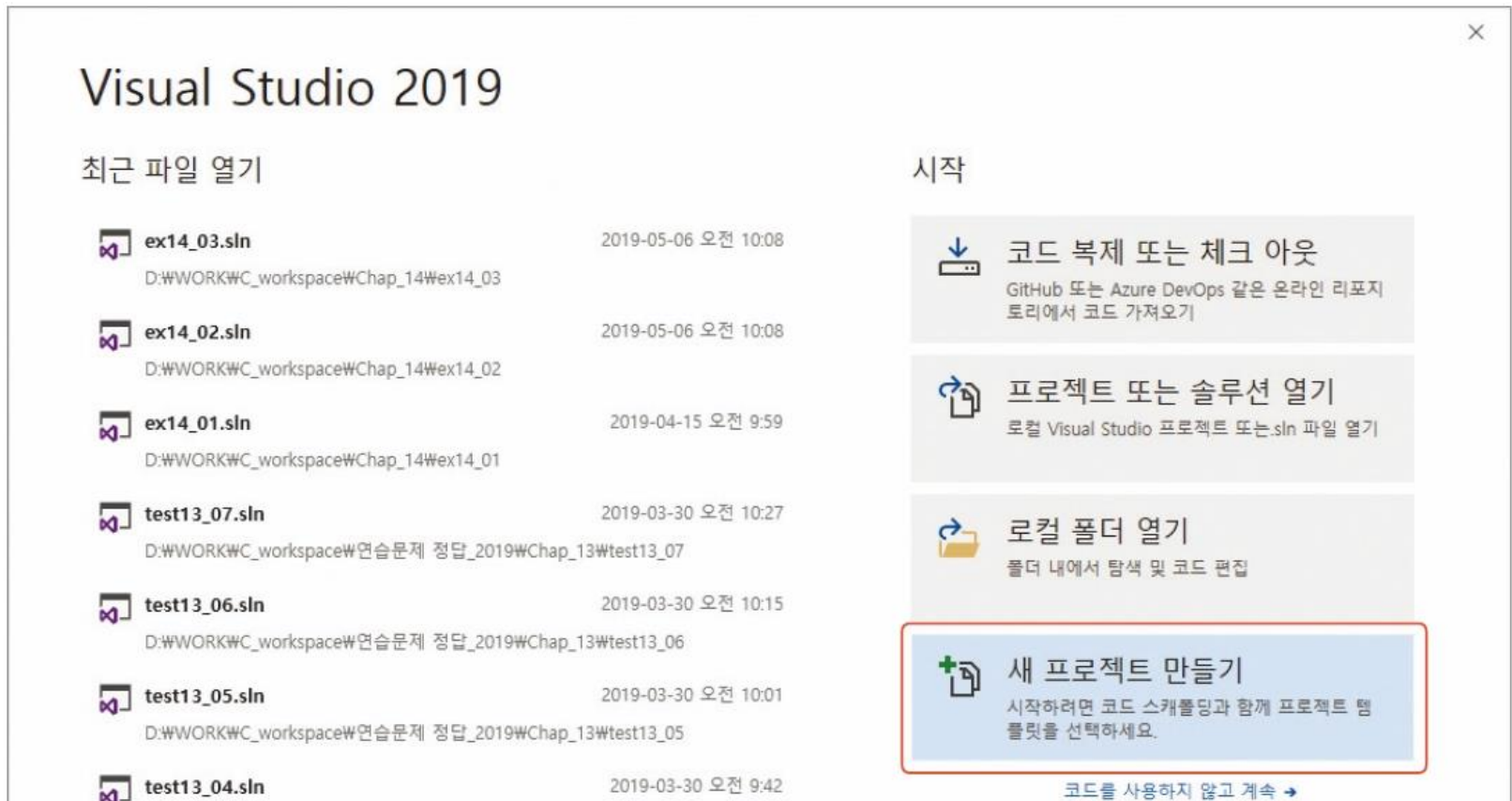
- **솔루션(solution)**; 문제 해결에 필요한 프로젝트가 들어 있는 컨테이너
- **프로젝트(project)**: 하나의 실행 파일을 만드는데 필요한 여러 가지 항목들이 들어 있는 컨테이너



# 프로젝트 생성하기

30

## 1. 화면에서 [새 프로젝트 만들기] 클릭



# 프로젝트 생성하기

31

## 2.[Windows 데스크톱 마법사] 선택



# 프로젝트 생성하기

32

## 3. 프로젝트 이름 입력 후 저장위치 지정

새 프로젝트 구성

Windows 데스크톱 마법사 C++ Windows 데스크톱 콘솔 라이브러리

프로젝트 이름

test 1

위치

D:\WORK\C\_workspace 2

솔루션 이름 1

test

☐ 솔루션 및 프로젝트를 같은 디렉터리에 배치

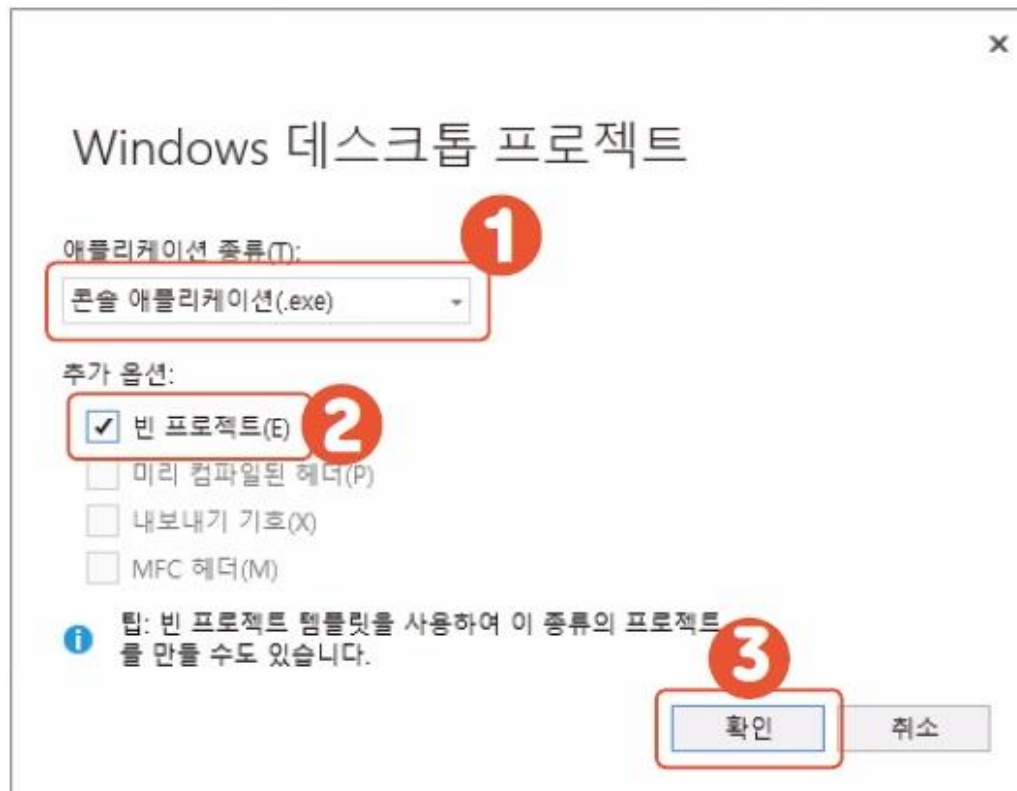
뒤로(B) 3 만들기(C)



# 프로젝트 생성하기

33

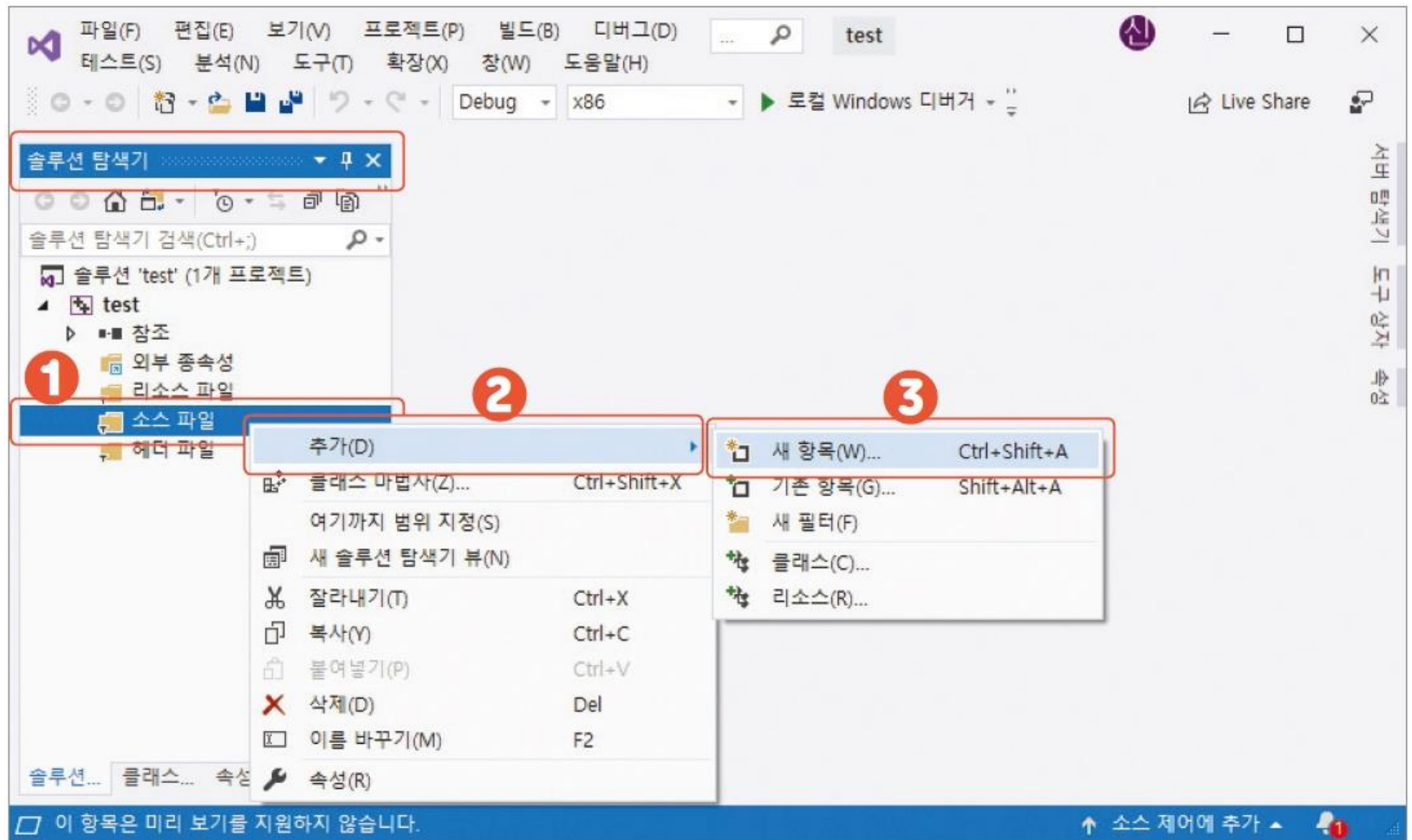
## 4. 콘솔 애플리케이션, 빈 프로젝트 선택



# 소스 파일 생성하기

34

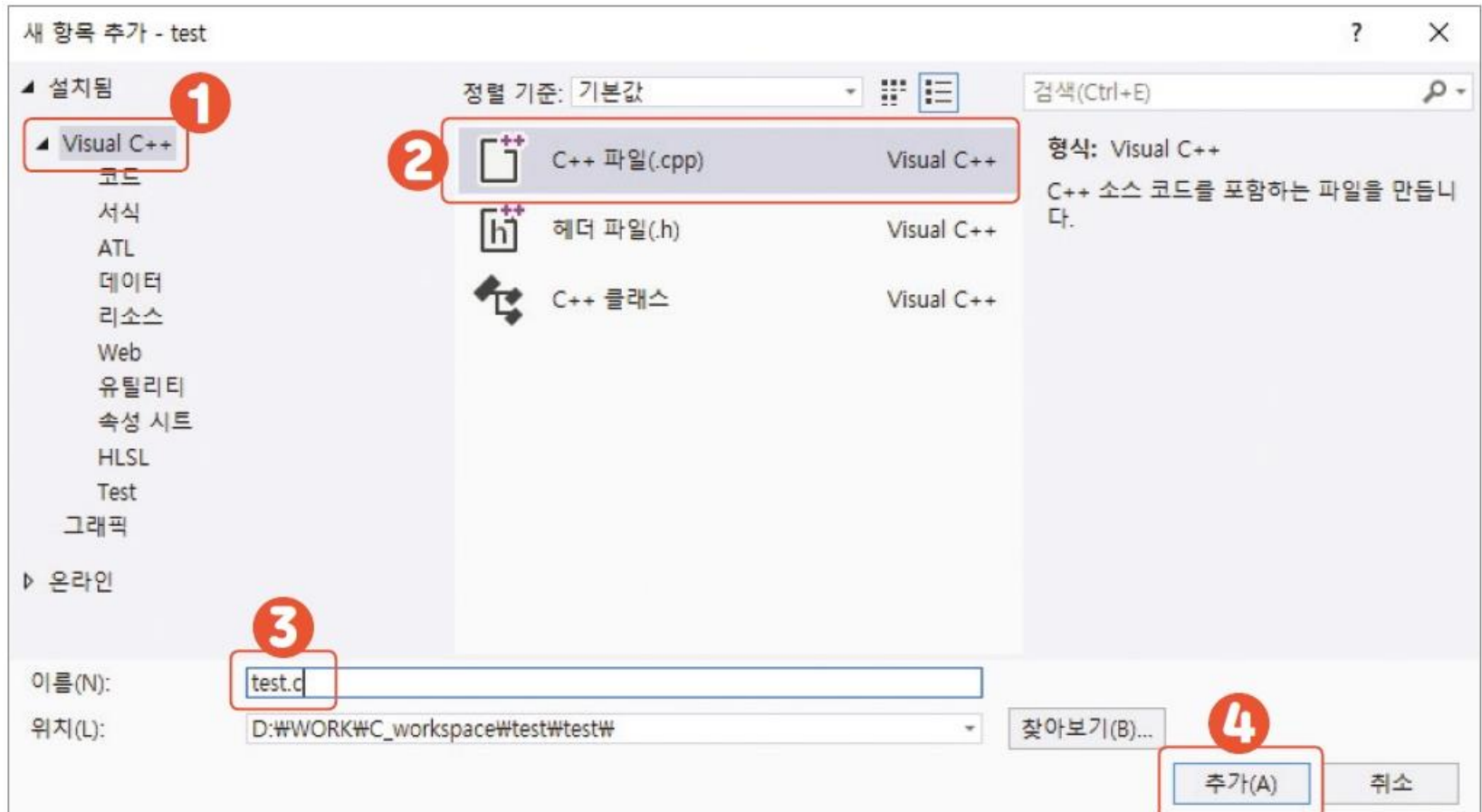
## 1. 소스파일(오른쪽마우스버튼)→ 추가 → 새항목



# 소스 파일 생성하기

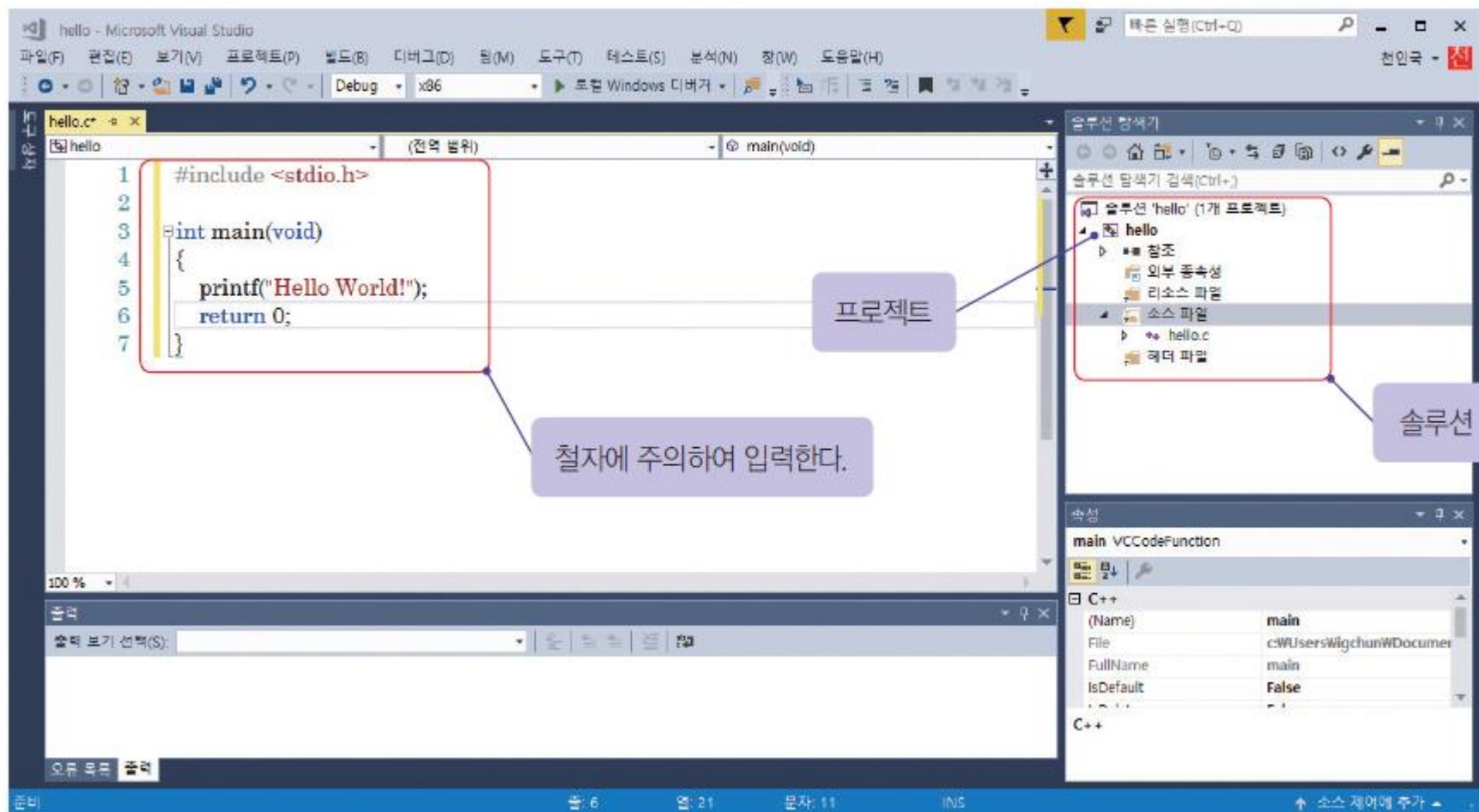
35

2. Visual C++ → C++ 파일(.cpp) → 파일 이름 입력  
→ <추가> 클릭



# 프로그램 입력

36



# 프로그램 입력

include나 stdio는 붙여쓴다.

```

# i n c l u d e < s t d i o . h >

i n t   m a i n ( v o i d )
{
    p r i n t f ( " H e l l o   W o r l d ! " ) ;
    r e t u r n 0 ;
}

```

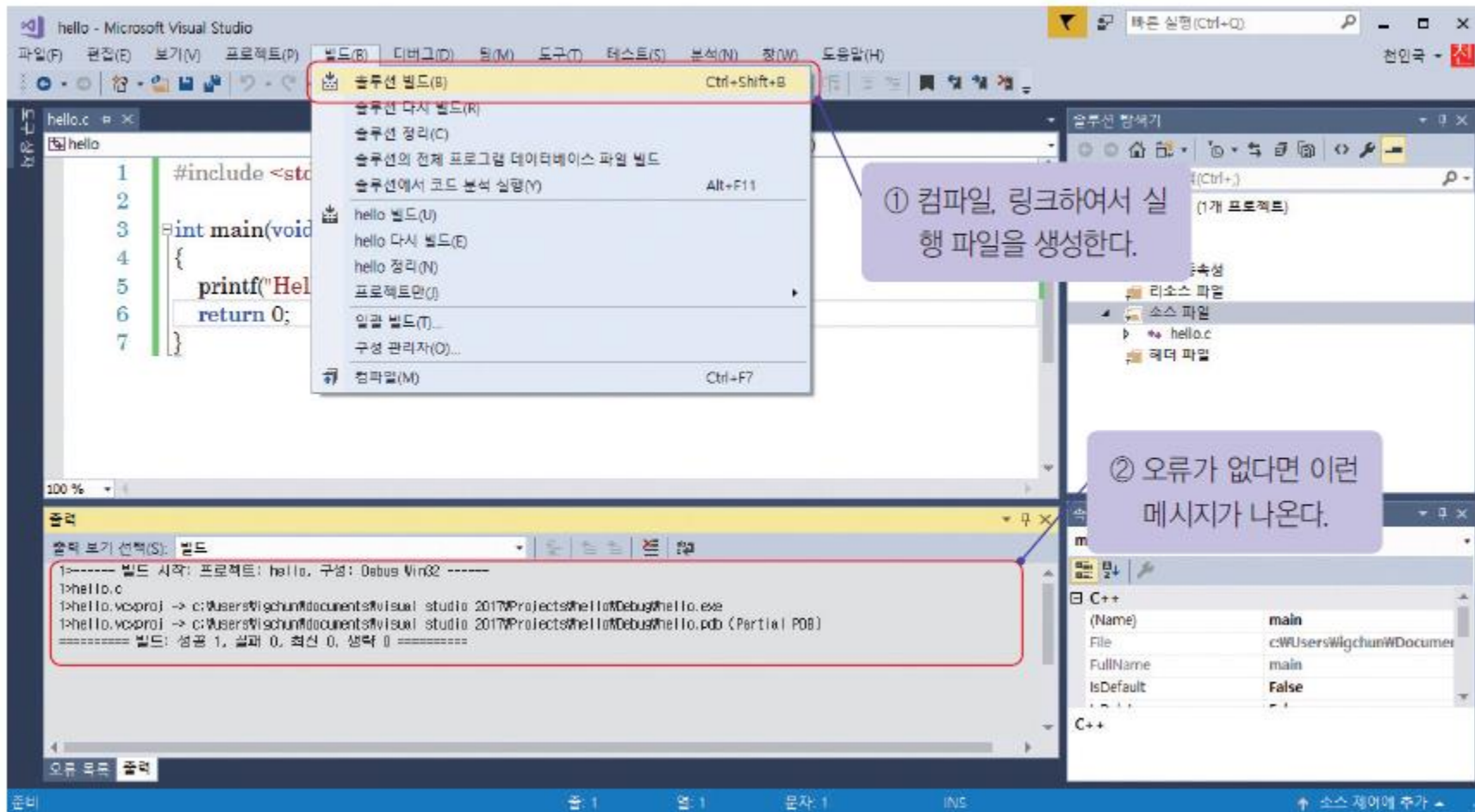
들어쓴다.

문장의 끝에는 세미콜론

문장의 끝에는 세미콜론

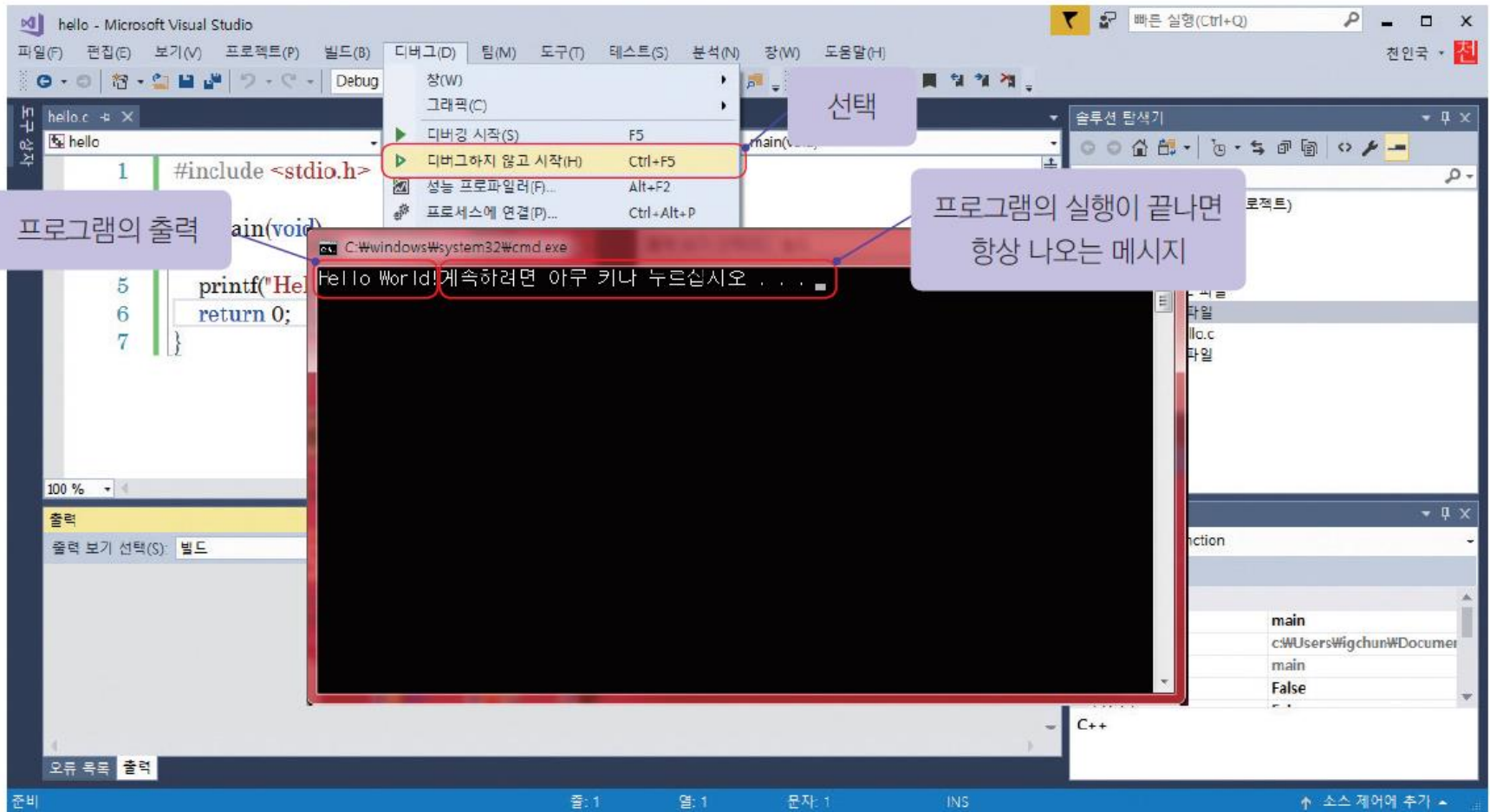
# 컴파일하기

38



# 프로그램 실행 하기

39



# Contents

40

2.1

프로그램 개발 과정

2.2

통합 개발 환경

2.3

비주얼 스튜디오 설치

2.4

비주얼 스튜디오 사용하기

2.5

예제 프로그램의 간략한 설명

2.6

예제 프로그램의 응용

2.7

오류수정



# 첫번째 프로그램의 설명

41

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

```
int main(void)
```

```
{
```

```
    printf("Hello World!");
```

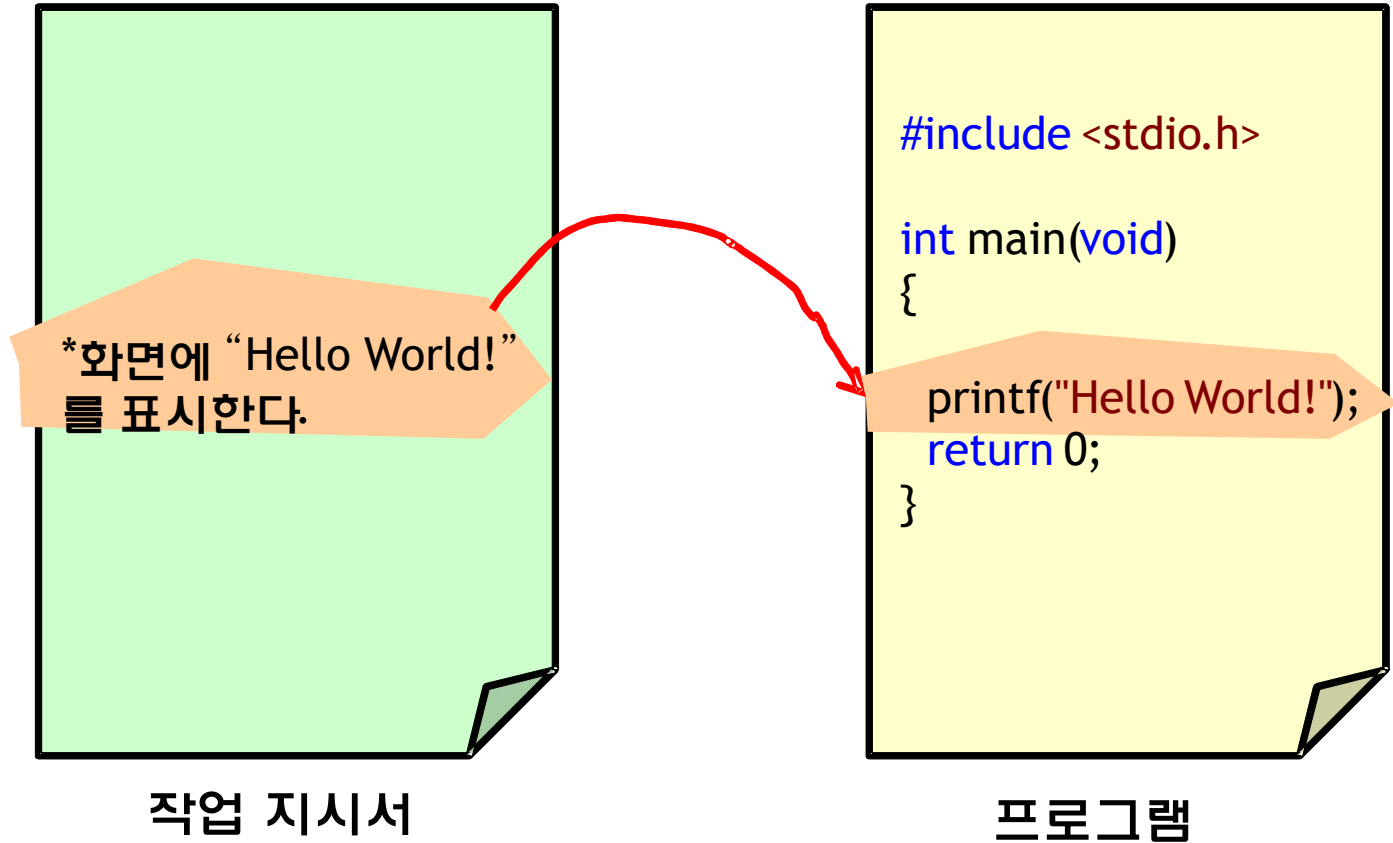
```
    return 0;
```

```
}
```



# 프로그램 == 작업 지시서

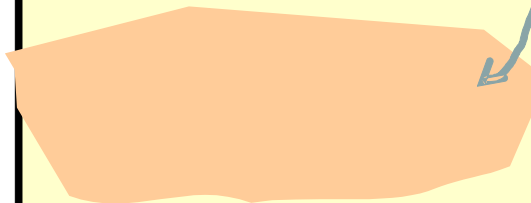
42



# 작업을 적어주는 위치

43

```
#include <stdio.h>

int main(void)
{
    
    return 0;
}
```

여기다가 원하는 작업을 수행하는 문장을 적어준다.

프로그램

# 간략한 소스 설명

44



프로그램

# 헤더 파일 포함

45

- #include는 소스 코드 안에 특정 파일을 현재의 위치에 포함

- 주의!: 전처리기 지시자 문장 끝에는 세미콜론(;)을 붙이면 안 된다.



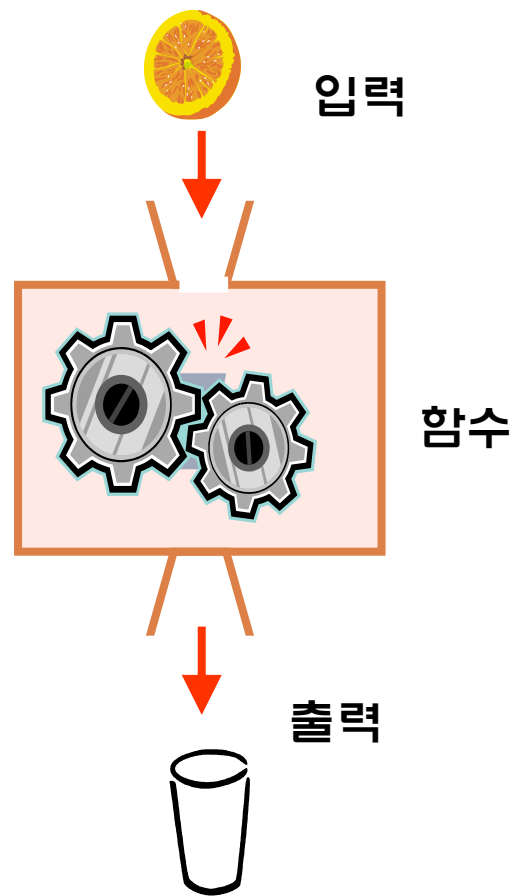
#include <stdio.h>

- 헤더 파일(header file): 컴파일러가 필요로 하는 정보를 가지고 있는 파일
- stdio.h: standard input output header file

# 함수

46

- 함수(function): 특정한 작업을 수행하기 위하여 작성된 독립적인 코드
- (참고) 수학적 함수
- 프로그램 = 함수의 집합  $y = x^2 + 1$
- `main()`은 가장 먼저 수행되는 함수



# main 함수

47

- 함수의 반환 값 타입

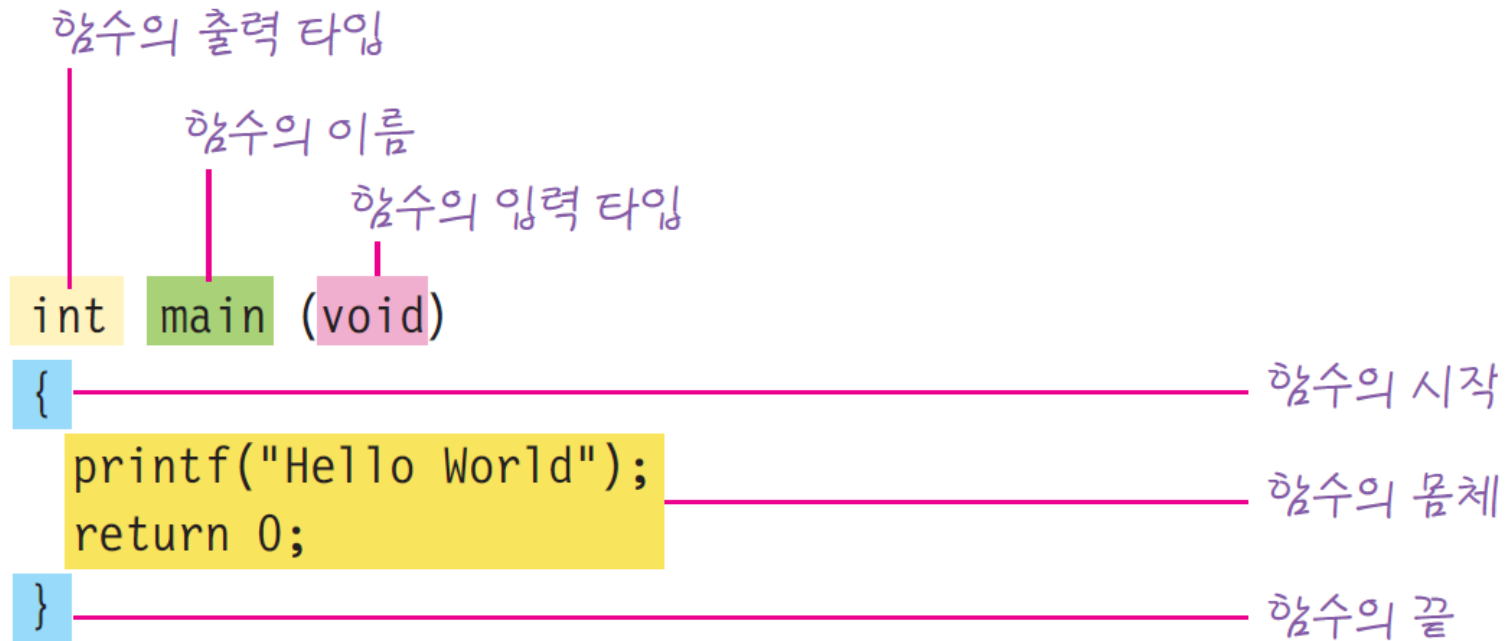
- 함수의 입력 값 타입

**int** main(void)

- 함수의 이름
- main()은 가장 먼저 수행되는 함수

# 함수의 간략한 설명

48

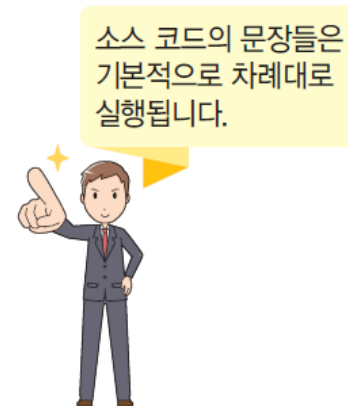
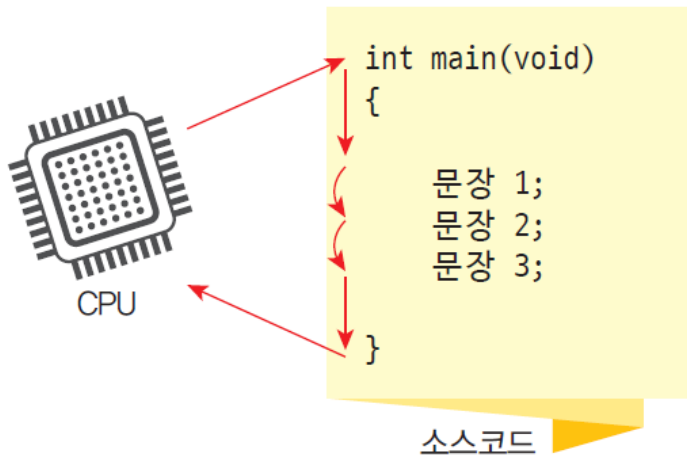




# 문장

49

- 함수는 여러 개의 문장으로 이루어진다.
- 문장들은 순차적으로 실행된다.
- 문장의 끝에는 반드시 ;이 있어야 한다.



# printf() 호출

50

- printf()는 컴파일러가 제공하는 함수로서 출력을 담당합니다.

`printf("Hello World!");`



- 큰따옴표 안의 문자열을 화면에 출력합니다.

# 함수의 반환 값

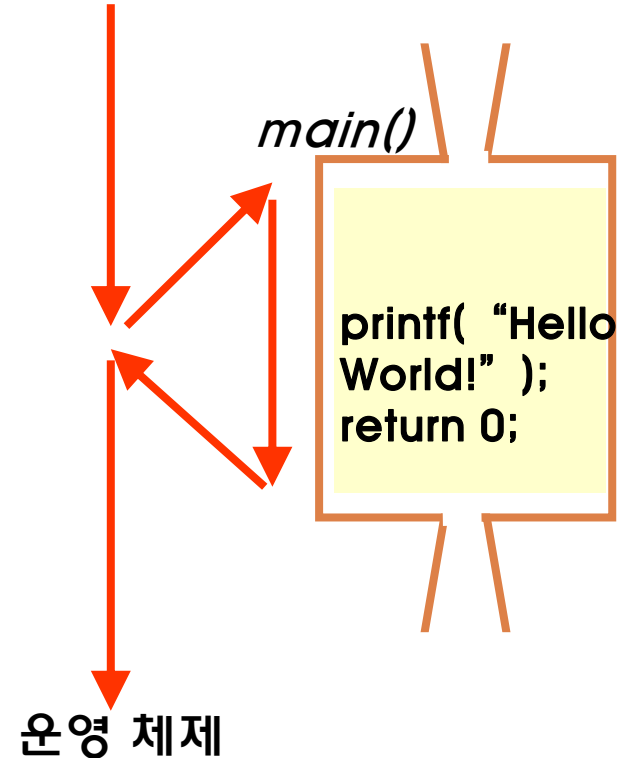
51

- return은 함수의 결과값을 외부로 반환합니다

return 0;

- 반환값은 0

운영 체제



# Contents

52

2.1

프로그램 개발 과정

2.2

통합 개발 환경

2.3

비주얼 스튜디오 설치

2.4

비주얼 스튜디오 사용하기

2.5

예제 프로그램의 간략한 설명

2.6

예제 프로그램의 응용

2.7

오류수정

# 응용 프로그램 #1

53

- 다음과 같은 출력을 가지는 프로그램을 제작하여 보자.



# 첫번째 버전

54

- 문장들은 순차적으로 실행된다는 사실 이용

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

```
int main(void)
```

```
{
```

```
    printf("Hello World!");
```

```
    printf("Kim ChulSoo");
```

```
    return 0;
```

```
}
```

2개의 문장은  
순차적으로 실행된다.

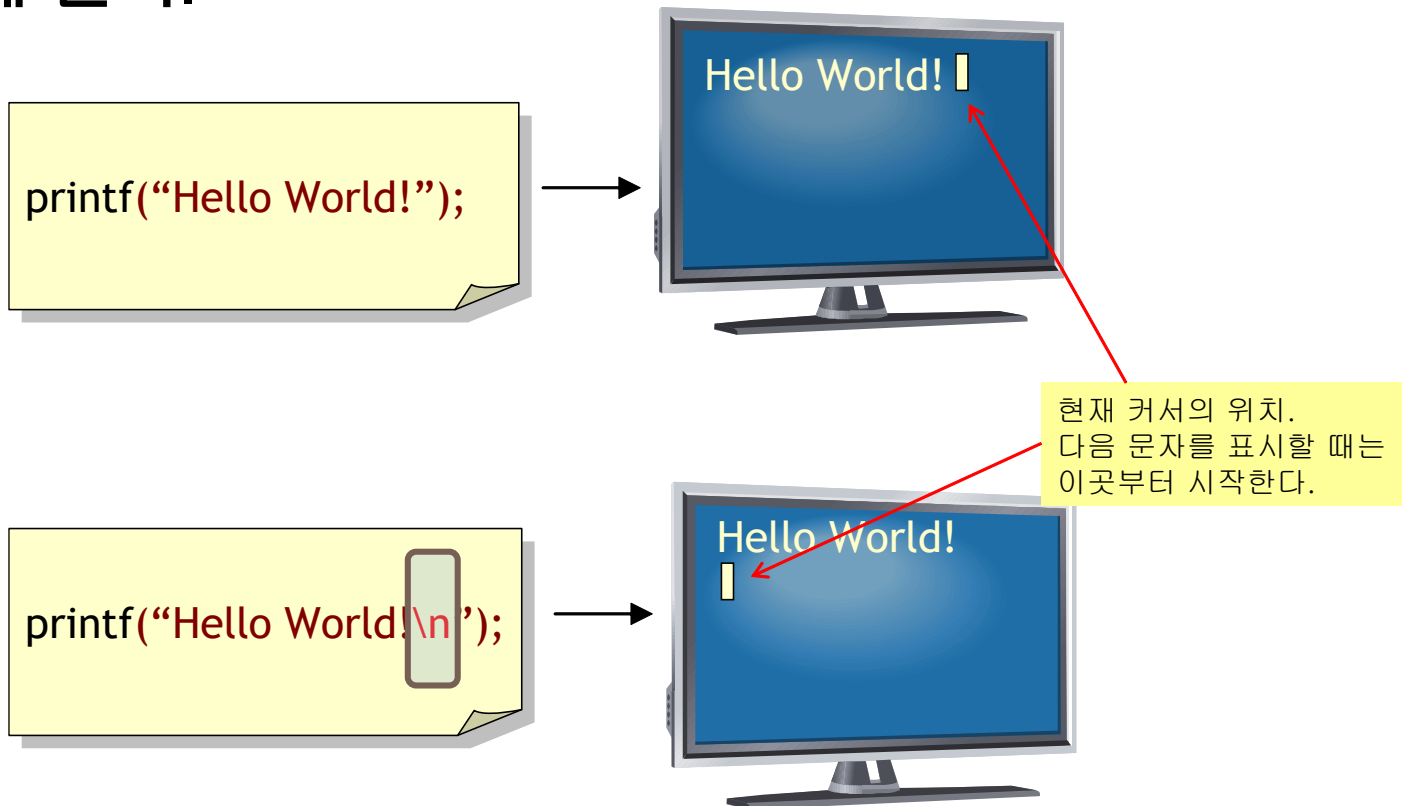


Hello World! Kim ChulSoo

# 줄바꿈 문자 \n

55

- 줄바꿈 문자인 \n은 화면에서 커서는 다음줄로 이동하게 한다.



# 줄바꿈 문자 2개를 사용하면?

56

```
printf("Hello \nWorld! \n");
```





# 변경된 프로그램

57

- 줄바꿈 문자를 포함하면 우리가 원하던 결과가 된다.

```
#include <stdio.h>

int main(void)
{

    printf("Hello World!\n");
    printf("Kim ChulSoo \n");

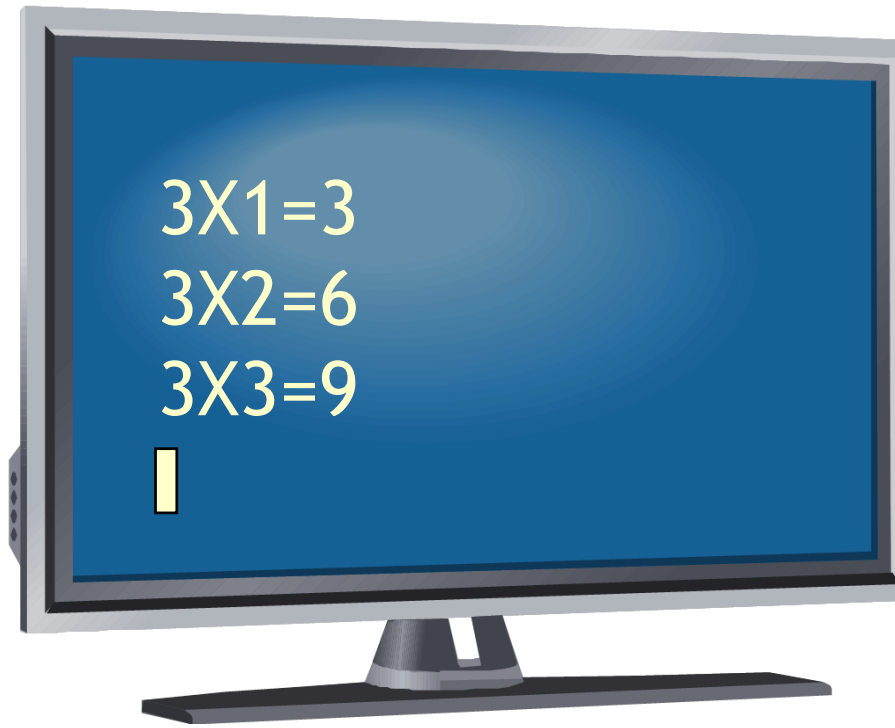
    return 0;
}
```



# 응용 프로그램 #2

58

- 다음과 같은 출력을 가지는 프로그램을 제작하여 보자.



# 응용 프로그램

59

- 역시 문장들은 순차적으로 수행된다는 점을 이용한다.

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

```
int main(void)
```

```
{
```

```
    printf("3 X 1 = 3\n");
```

```
    printf("3 X 2 = 6\n");
```

```
    printf("3 X 3 = 9\n");
```

```
    return 0;
```

```
}
```

3개의 문장은  
순차적으로 실행된다.

# 중간 점검

60

- 화면에 새로운 줄을 만드는데 사용되는 특수한 기호는?
- “사과” , “오렌지” , “포도” 를 한 줄에 하나씩 출력하는 프로그램을 작성하여 보자.
- 구구단 3단 전체를 출력하는 프로그램을 작성하여 보자.



# lab: 간단한 계산을 해보자

61

- 덧셈과 뺄셈, 곱셈, 나눗셈 계산을 하는 프로그램을 작성해보자.

📌 실행결과

$2+5=5$

$2-3=-1$

$2*3=6$

$2/3=0$

# solution

62

```
#include <stdio.h>

int main(void)
{
    printf("2+5=%d\n", 2 + 3);
    printf("2-3=%d\n", 2 - 3);
    printf("2*3=%d\n", 2 * 3);
    printf("2/3=%d\n", 2 / 3);
    return 0;
}
```

# Contents

63

2.1

프로그램 개발 과정

2.2

통합 개발 환경

2.3

비주얼 스튜디오 설치

2.4

비주얼 스튜디오 사용하기

2.5

예제 프로그램의 간략한 설명

2.6

예제 프로그램의 응용

2.7

오류수정

# 오류 수정 및 디버깅

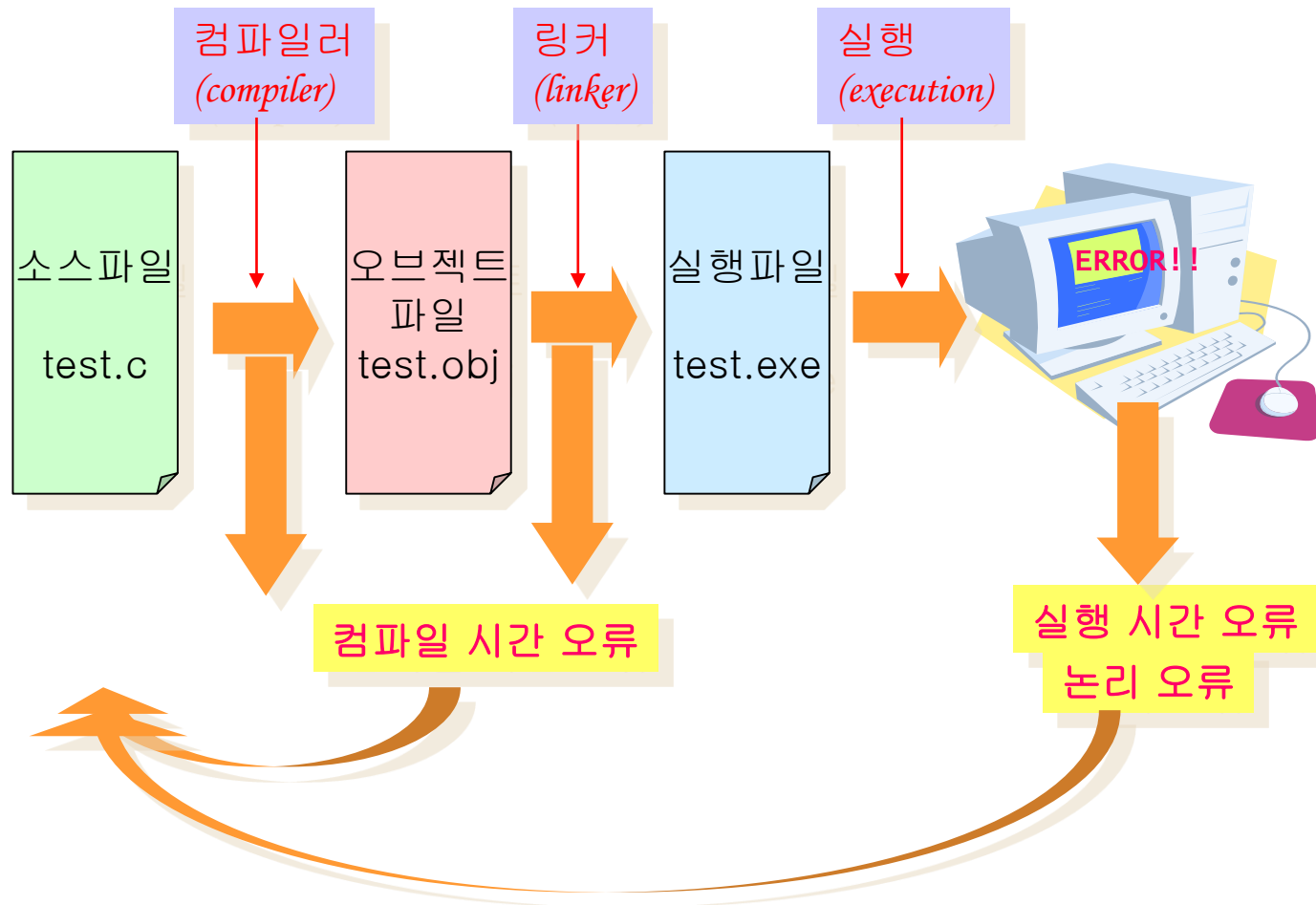
64

- 컴파일이나 실행 시에 오류가 발생할 수 있다.
- 에러와 경고
  - ▣ 에러(error): 심각한 오류
  - ▣ 경고(warning): 경미한 오류
- 오류의 종류
  - ▣ 컴파일 시간 오류: 대부분 문법적인 오류
  - ▣ 실행 시간 오류: 실행 중에 0으로 나누는 연산 같은 오류
  - ▣ 논리 오류: 논리적으로 잘못되어서 결과가 의도했던 대로 나오지 않는 오류



# 오류 수정 과정

65



# 오류 #1

66

Visual Studio Hello - Microsoft Visual Studio

파일(F) 편집(E) 보기(V) 프로젝트(P) 빌드(B) 디버그(D) 팀(M) 도구(T) 테스트(S) 분석(N) 창(W) 도움말(H) 로그인

Debug x86 로컬 Windows 디버거

Hello.c (전역 범위) main(vodi)

```
1 #include<stdio.h>
2
3 int main(vodi)
4 {
5     printf("Hello World!");
6     return 0;
7 }
```

; 이 생략되었다

오류 목록

전체 솔루션 2 오류 0 경고 0 메시지 빌드 + IntelliSense

검색 오류 목록

번호	코드	설명	프로젝트	파일	줄
E0065		';'가 필요합니다.	Hello	Hello.c	6
C2143		구문 오류: ';'이(가) 'return' 앞에 없습니다.	Hello	hello.c	6

오류가 발견된 소스 파일

오류가 발견된 줄 번호

속성

빌드 실패 소스 제어에 추가

# 오류 #2

67

문자열을 표시할 때 “를 누락

오류 목록

전체 솔루션 3 오류 0 경고 0 메시지 빌드 + IntelliSense

코드	설명	프로젝트	파일	줄
E0020	식별자 "Hello"이(가) 정의되어 있지 않습니다.	Hello	Hello.c	5
E0018	'/'가 필요합니다.	Hello	Hello.c	5
C2143	구문 오류: ';'이(가) 'return' 앞에 없습니다.	Hello	hello.c	6

# 오류 #3

68

Visual Studio interface showing a C program in `Hello.c` with the following code:

```
1 #include<stdio.h>
2
3 int main(vodi)
4 {
5     print("Hello World!");
6     return 0;
7 }
```

A red box highlights the `print` function call, with a callout bubble stating: **printf이어야한다** (It should be printf).

The **오류 목록** (Error List) pane at the bottom shows two errors:

- LNK2019**: print 외부 기호(참조 위치: \_main 함수)에서 확인하지 못했습니다. (The external symbol 'print' (reference location: \_main function) could not be found.)
- LNK1120**: 1개의 확인할 수 없는 외부 참조입니다. (There is 1 unresolved external reference.)

A red box highlights the **LNK2019** error, with a callout bubble stating: **함수를 찾지 못했음** (Could not find the function).

The **솔루션 탐색기** (Solution Explorer) on the right shows the project structure:

- 참조 (References)
- 외부 종속성 (External Dependencies)
- 리소스 파일 (Resource Files)
- 소스 파일 (Source Files)
- Hello.c
- 헤더 파일 (Header Files)

The status bar at the bottom indicates **빌드 실패** (Build Failed).

# 논리 오류

69

- 다음과 같은 출력을 가지는 프로그램을 작성하여 보자.



# 논리 오류가 존재하는 프로그램

70

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

```
int main(void)
```

```
{
```

```
    printf("Hey!");
```

```
    printf("Good Morning");
```

```
    return 0;
```

```
}
```

줄이 바뀌지  
않았음!

Hey!Good Morning

|

# 논리 오류가 수정된 프로그램

71

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

```
int main(void)
```

```
{
```

```
    printf("Hey! \n");
```

```
    printf("Good Morning \n");
```

```
    return 0;
```

```
}
```

논리 오류  
수정!!



Hey!  
Good Morning  
|



# 디버깅

72

## □ 디버깅: 논리 오류를 찾는 과정

아무래도 이부분이 수상해..



프로그램의 실행결과

논리 에러를 발견하는 것은  
수사관이 범죄 흔적을 이용하여  
범인을 찾는 것과 같습니다.





# 디버거(debugger)

73

hello (디버깅) - Microsoft Visual Studio

파일(F) 편집(E) 보기(V) 프로젝트(P) 빌드(B) 디버그(D) 팀(M) 도구(T) 테스트(S) 분석(N) 창(W) 도움말(H)

프로세스: [144720] hello.exe

hello.c

```
1 #include <stdio.h>
2
3 int main(void)
4 {
5     printf("Hello World!");
6     printf("Good Mornin");
7     return 0;
8 }
```

현재 실행되고 있는 위치

디버그(D) 메뉴:

- 장(W)
- 그래픽(C)
- 계속(C) F5
- 모두 중단(K) Ctrl+Alt+Break
- 디버깅 중지(E) Shift+F5
- 모두 분리(L)
- 모두 종료(M)
- 다시 시작(R) Ctrl+Shift+F5
- 코드 변경 내용 적용(A) Alt+F10
- 성능 프로파일러(F...) Alt+F2
- 프로세스에 연결(P...) Ctrl+Alt+P
- 기타 디버그 대상(H)
- 프로파일러
- 하 단계씩 코드 실행(I) F11
- 프로세서 단위 실행(O) F10**
- 프로세서 다가기(T) Shift+F11
- 간략한 조사식(Q...) Shift+F9
- 중단점 설정/해제(G) F9
- 새 중단점(B)
- 모든 중단점 삭제(A) Ctrl+Shift+F9
- 모든 DataTips 지우기(A)
- DataTips 내보내기(X)...
- DataTips 가져오기(I)...
- 다른 이름으로 덤프 저장(V)...
- 옵션(O)...
- hello 속성...

한 문장 단위로 실행한다.

진단 도구

진단 및 세션: 0 초(178 ms)(7) 선택됨

CPU (모든 프로세서에 대한 비율(%))

요약 이벤트 메모리 사용량 CPU 사용량

이벤트

표시(1/1)

스냅숏 만들기

스택

중단점 예외 설정 디버깅 옵션 성능 프로파일러

준비

소스 제어에 추가

# 디버거의 명령어 정의

74

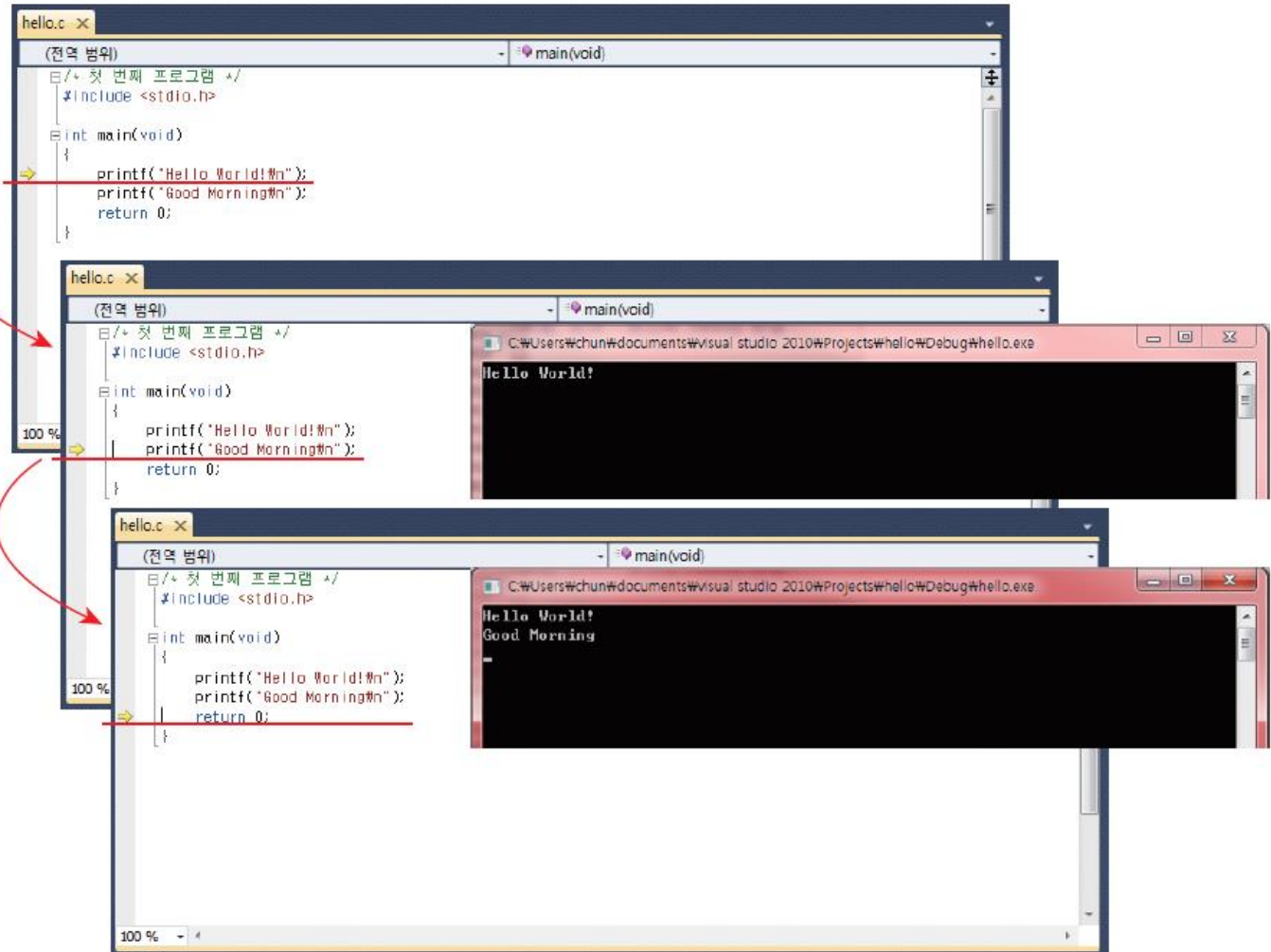
- F5 (Go): 실행
- F10 (Step Over): 한 문장씩 실행(함수도 하나의 문장 취급)
- F11 (Step Into): 한 문장씩 실행(함수 안으로 진입)
- F9 (Breakpoint): 현재 문장에 중단점을 설정

# 디버거의 실행 과정

75

F10을 누를 때마다  
한 문장씩 실행된다.

F10을 누를 때마다  
한 문장씩 실행된다.



# mini project

76

## □ 오류를 수정해보자!

```
#include <stdio.h>

int Main(void)
(
    printf(안녕하세요?\n);
    printf(이번 코드에는 많은 오류가 있다네요\n)
    print(제가 다 고쳐보겠습니다.\n);
    return 0;
)
```

## bug.c

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

```
2
```

```
3  int Main(void)
```

```
4  (
```

```
5      printf(안녕하세요? \n);
```

```
6      printf(이번 코드에는 많은 오류가 있다네요 \n)
```

```
7      print(제가 다 고쳐보겠습니다.\n);
```

```
8      return 0;
```

```
9  )
```

main

(가 아니라 {이어야 한다.

문장의 끝에는 ;가 있어야 한다.

print가 아니고 printf이어야 한다.

문자열에는 따옴표를 붙인다.

오늘은 어제보다 반드시 좋은 날이 될 것입니다.

**Thank  
You!**