

-Part1-

# 제1장 C 언어의 소개와 프로그램 작성 방법

(중요도: 상)

# 학습목차

1.1 C언어란 무엇인가 ?

1.2 컴파일러란 무엇인가?

1.3 프로그램 작성 4단계- 이론 & 실습

1.4 오류의 종류

1.5 디버깅 실습

1.5 C언어의 특징

1.6 C언어의 학습방식

1.7 Quiz1

# 1.1 C언어란 무엇인가? (1/2)

## ▶ C언어란?

- ✓ 인간과 컴퓨터 사이의 의사소통을 위한 **프로그래밍 언어**

## ▶ 프로그래밍 언어의 종류

- ✓ C언어 , C++언어, C#언어, Java언어 ...

## ▶ 프로그램

- ✓ 프로그래밍 언어로 프로그래밍한 작업 결과
  - C프로그램, C++프로그램, C#프로그램, Java프로그램 ...

## ▶ 프로그래머

- ✓ 프로그래밍을 하는 사람
  - C프로그래머, C++프로그래머, C#프로그래머, Java 프로그래머 ...

# 1.1 C언어란 무엇인가? (2/2)

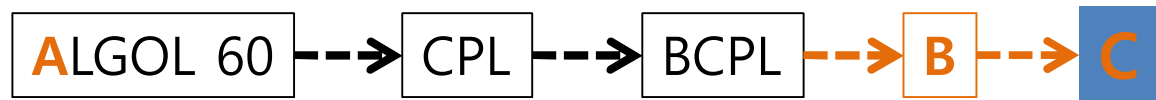
## ▶ C언어의 탄생

### ✓ 누가 만들었나?

- 미국 AT&T사의 벨(Bell) 연구소의 연구원들이 만들
- 켄 톰슨, 데니스 리치

### ✓ 왜 만들었나?

- 프로그램의 이식성을 높이기 위해...



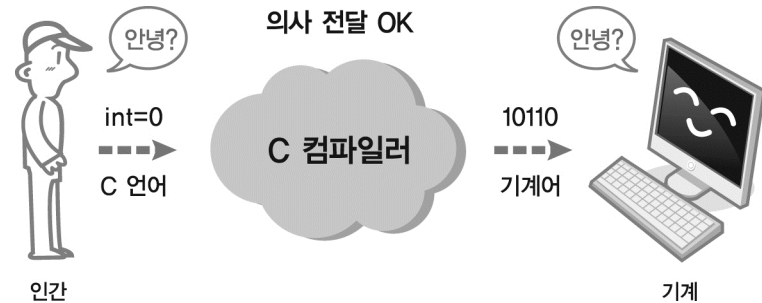
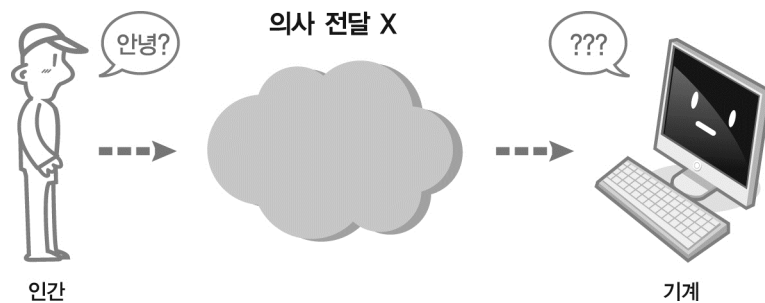
## 1.2 컴파일러란 무엇인가?

### ▶ 컴파일러란?

- ✓ 인간이 만든 프로그램을 기계가 이해 하도록 **기계어로 변환**하는 변환기
- ✓ 통역관의 역할

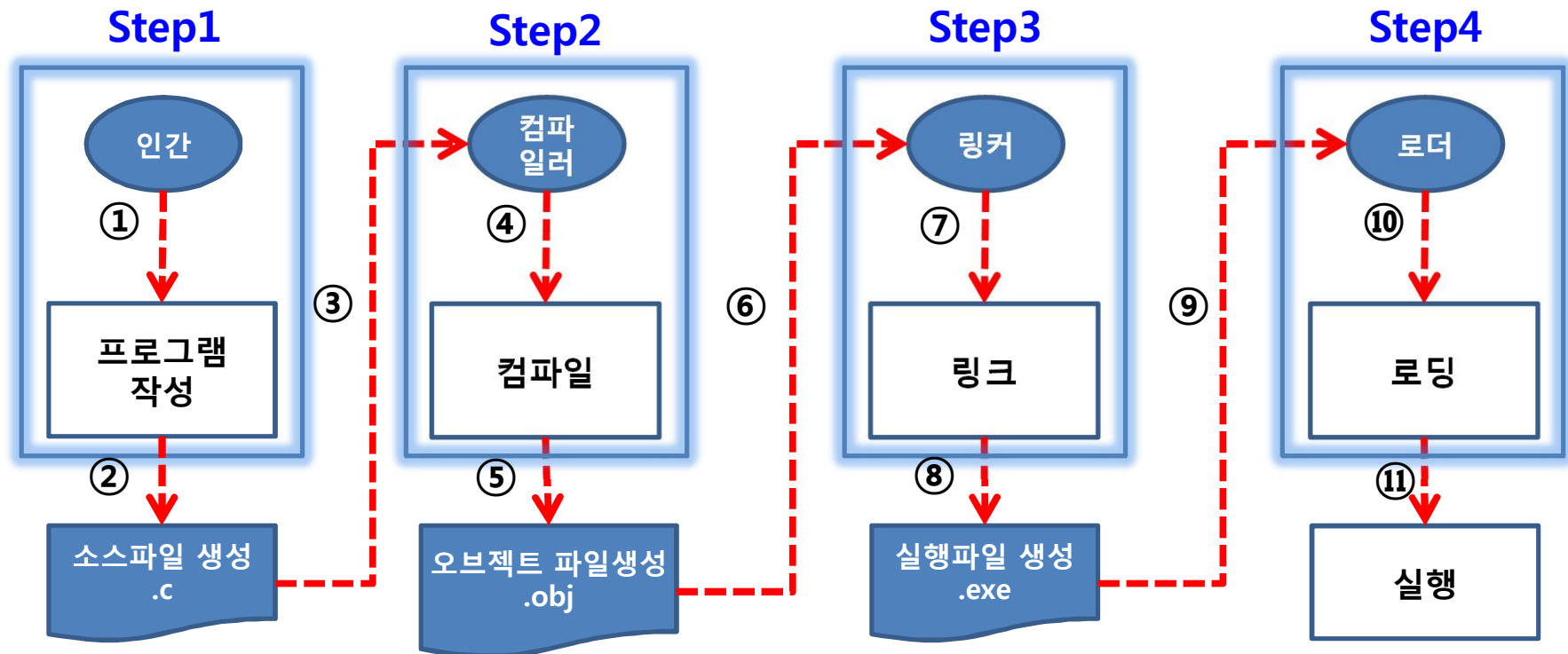
### ▶ 기계어란?

- ✓ 기계가 이해하는 **2진 숫자(0과 1)**로 작성된 언어



## 1.3 프로그램 작성 방법 4단계 - 이론 & 실습 (1/4)

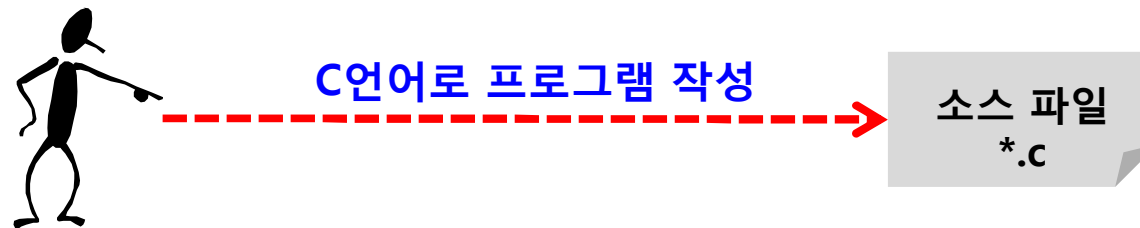
### ▶ 프로그램 작성 방법 4 단계 - 이론



## 1.3 프로그램 작성 방법 4단계 - 이론 & 실습 (2/4)

### ▶ 프로그램 작성 방법 4 단계 - 이론

#### ✓ Step1



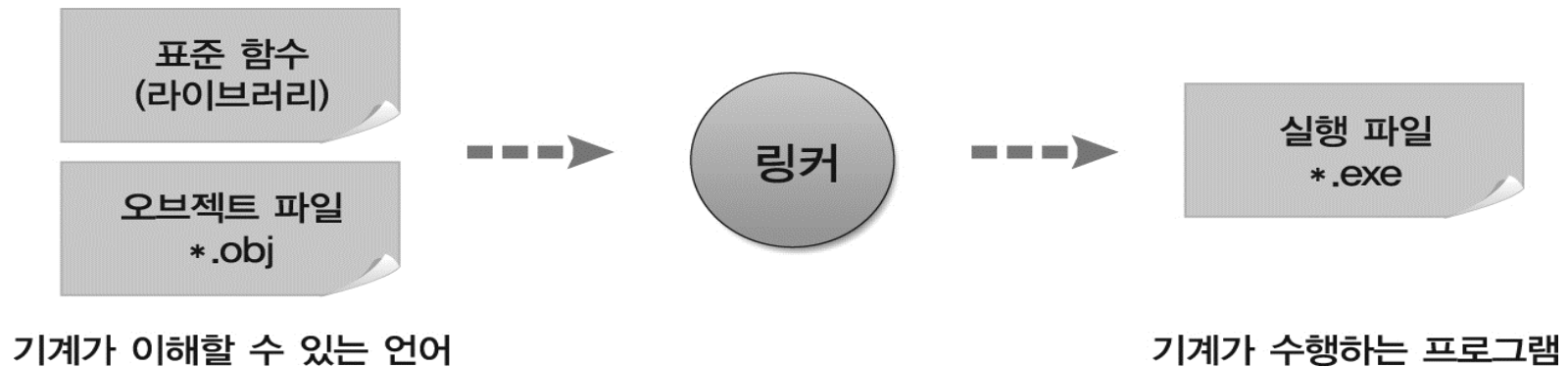
#### ✓ Step2



## 1.3 프로그램 작성 방법 4단계 - 이론 & 실습 (3/4)

### ▶ 프로그램 작성 방법 4 단계 - 이론

#### ✓ Step3

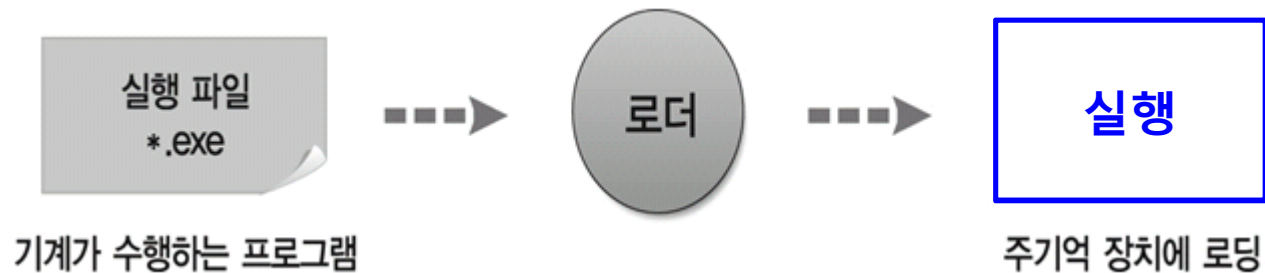




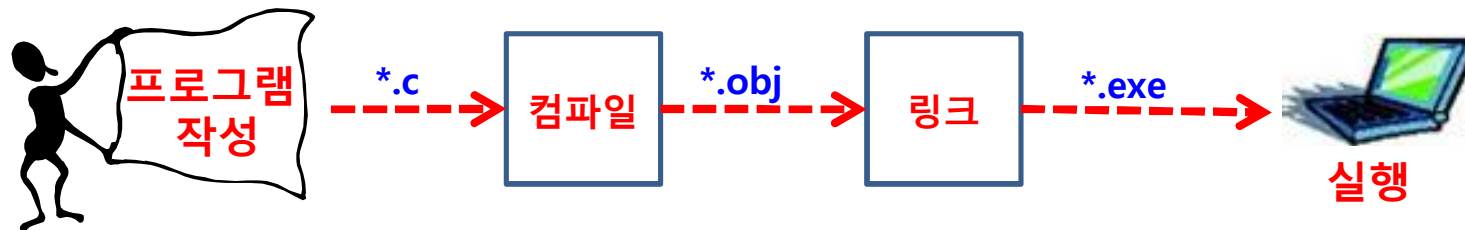
## 1.3 프로그램 작성 방법 4단계 - 이론 & 실습 (4/4)

### ▶ 프로그램 작성 방법 4 단계 - 이론

#### ✓ Step4

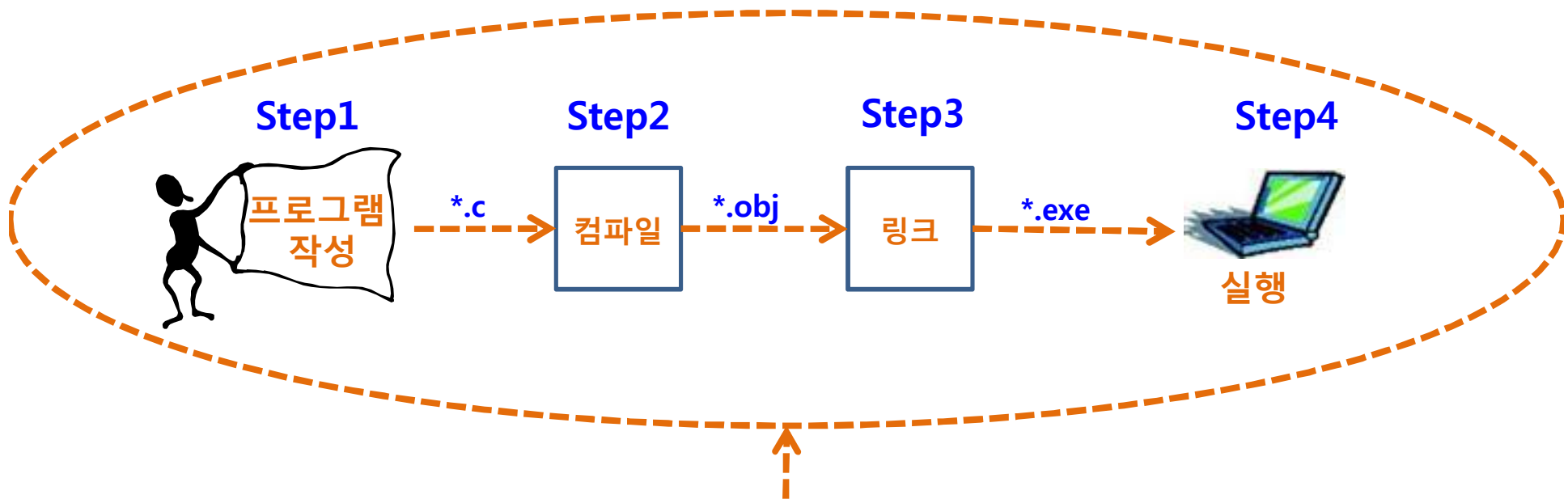


#### ✓ 프로그램 작성 4단계



## 1.3 프로그램 작성 방법 4단계 - 이론 & 실습

### ▶ 프로그램 작성 방법 4 단계 - 실습



Step1 ~ Step4 까지의 작업을 도와주는 소프트웨어를  
'통합 개발 소프트웨어'라고 한다.

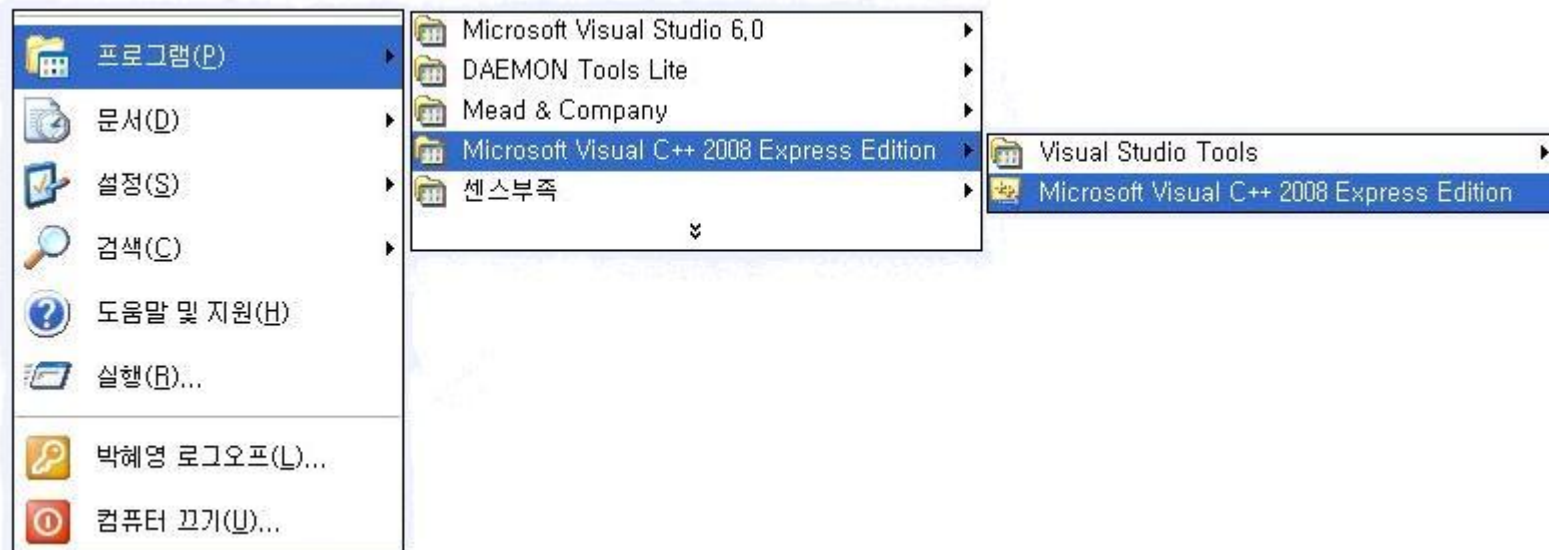
## 1.3 프로그램 작성 방법 4단계 실습 – Step0 (1/2)

### ▶ 통합 개발 소프트웨어 설치

#### ✓ Visual C++ 2008 Express Edition 설치

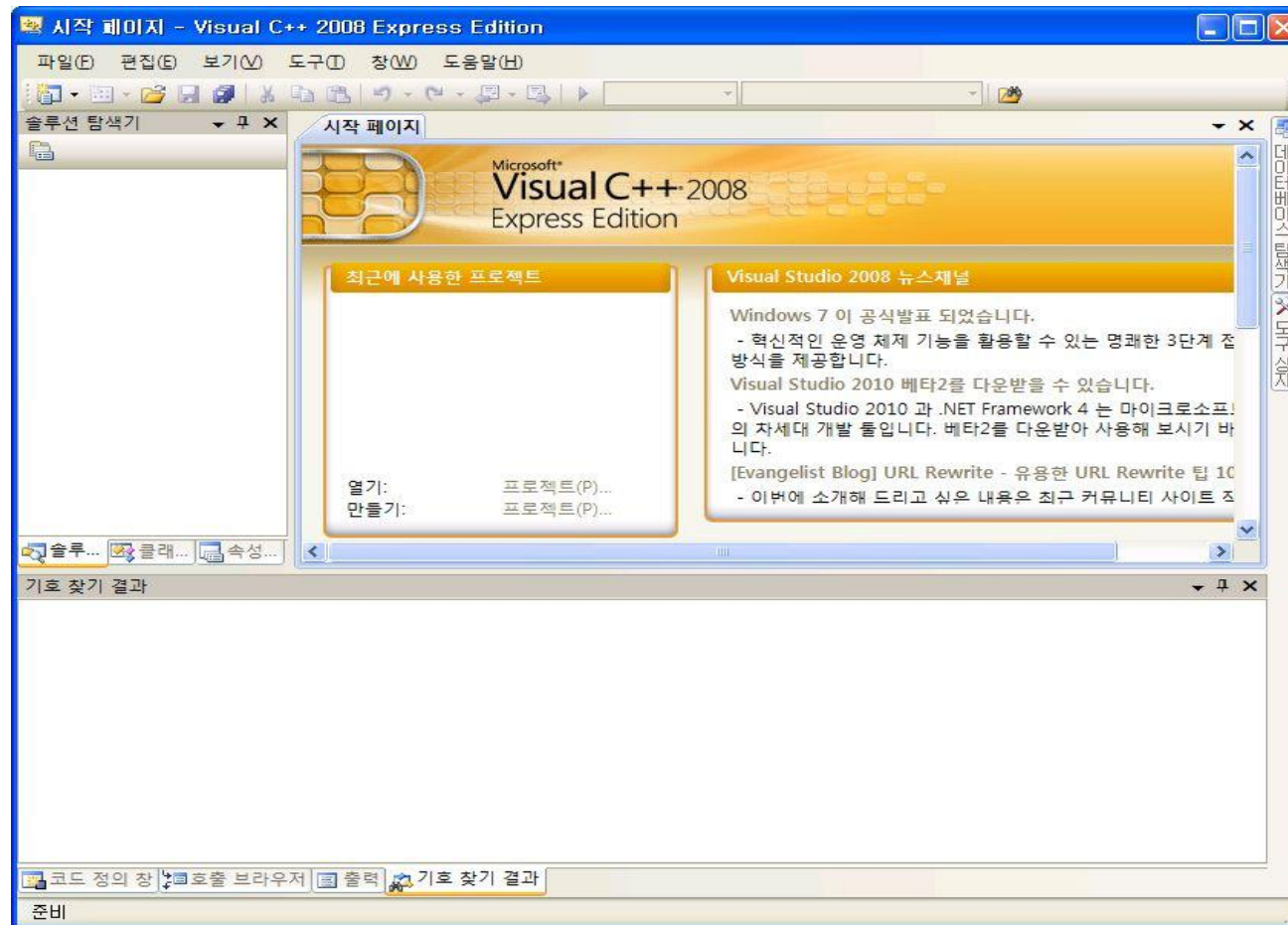
– 방법1: e-class에서 설치 파일 및 설치 매뉴얼 다운로드

### ▶ Visual C++ 2008 Express Edition 실행



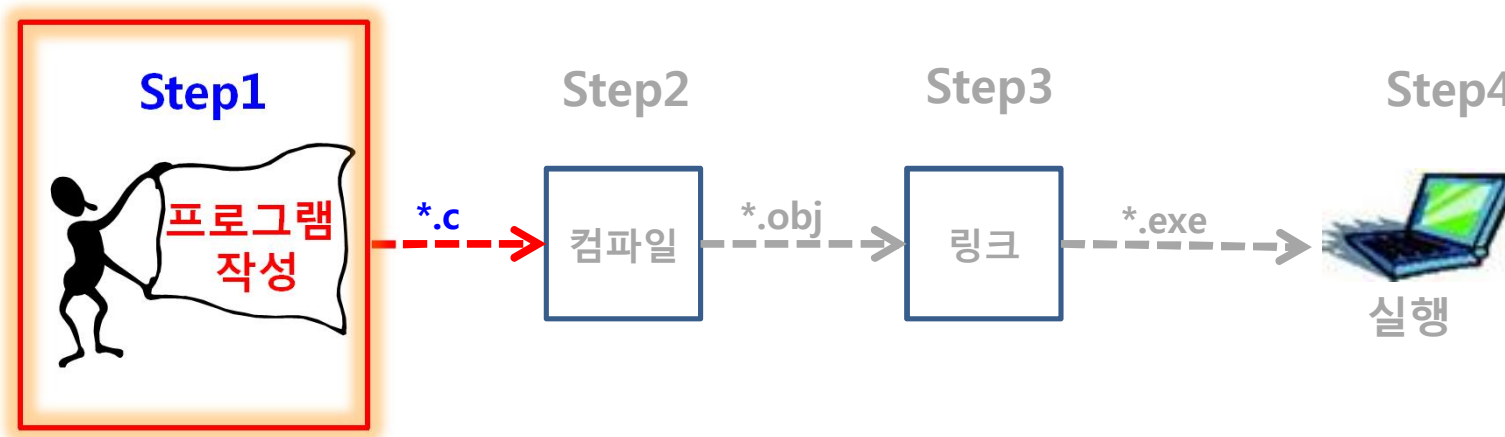
## 1.3 프로그램 작성 방법 4단계 실습 – Step0 (2/2)

### ▶ Visual C++ 2008 Express Edition 실행



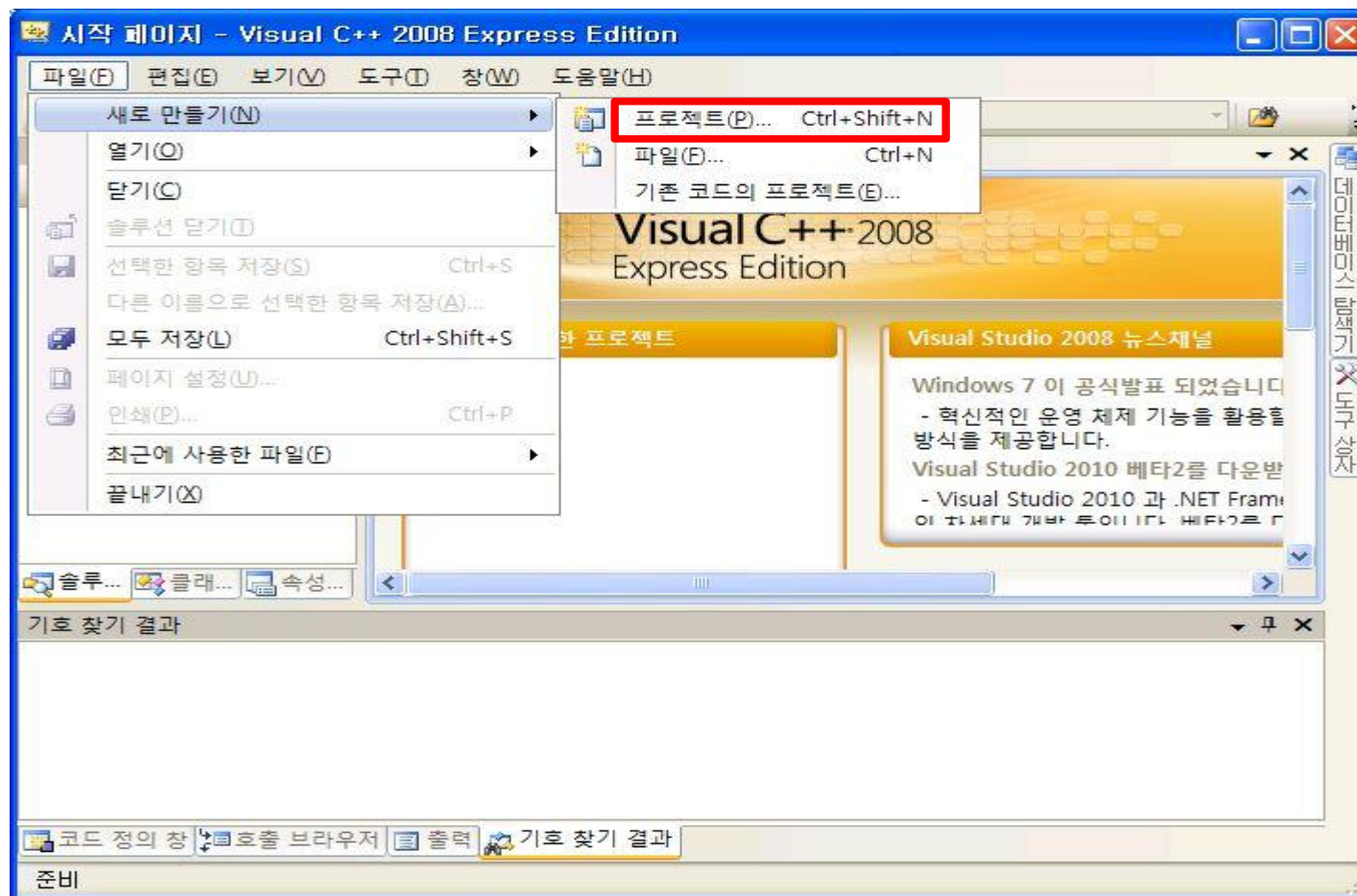
## 1.3 프로그램 작성 방법 4단계 - 이론 & 실습

### ▶ 프로그램 작성 방법 4 단계 - 실습



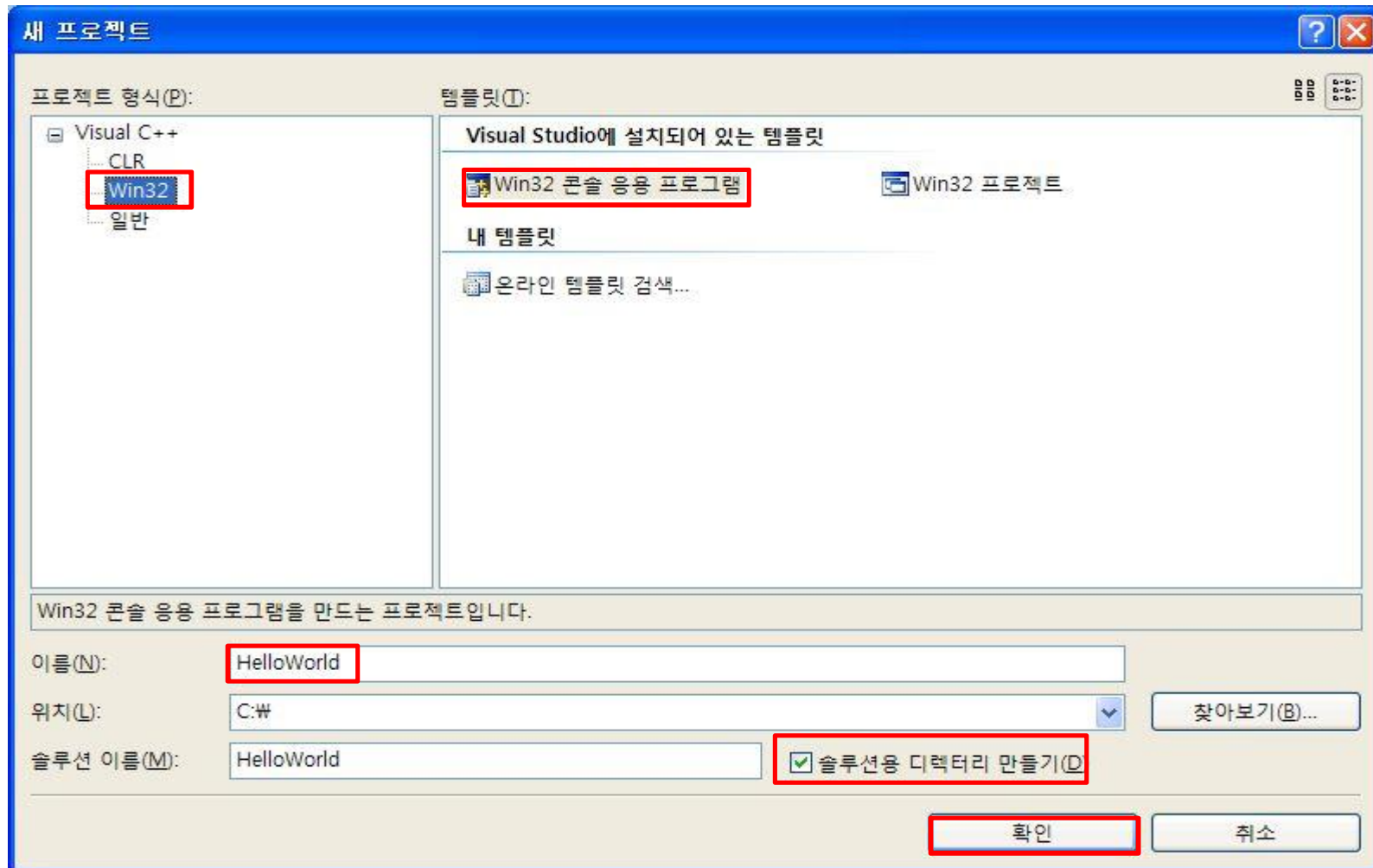
## 1.3 프로그램 작성 방법 4단계 실습 – Step1 (1/9)

▶ 프로그램 작성 단계 – 소스파일 (Hello.c)을 만들어 보자!



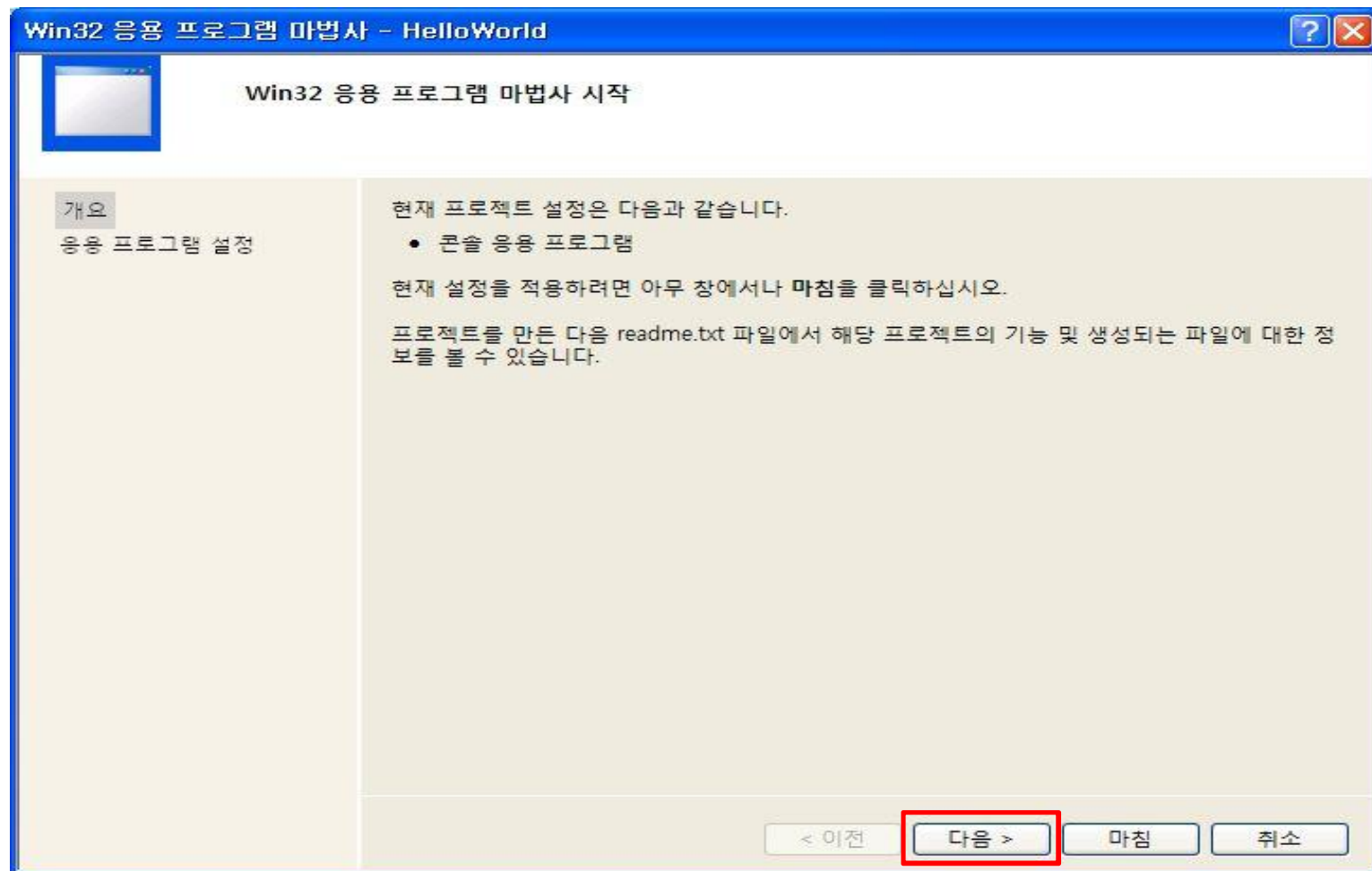
## 1.3 프로그램 작성 방법 4단계 실습 – Step1 (2/9)

▶ 프로그램 작성 단계 – 소스파일 (Hello.c)을 만들어 보자!



## 1.3 프로그램 작성 방법 4단계 실습 – Step1 (3/9)

▶ 프로그램 작성 단계 – 소스파일 (Hello.c)을 만들어 보자!





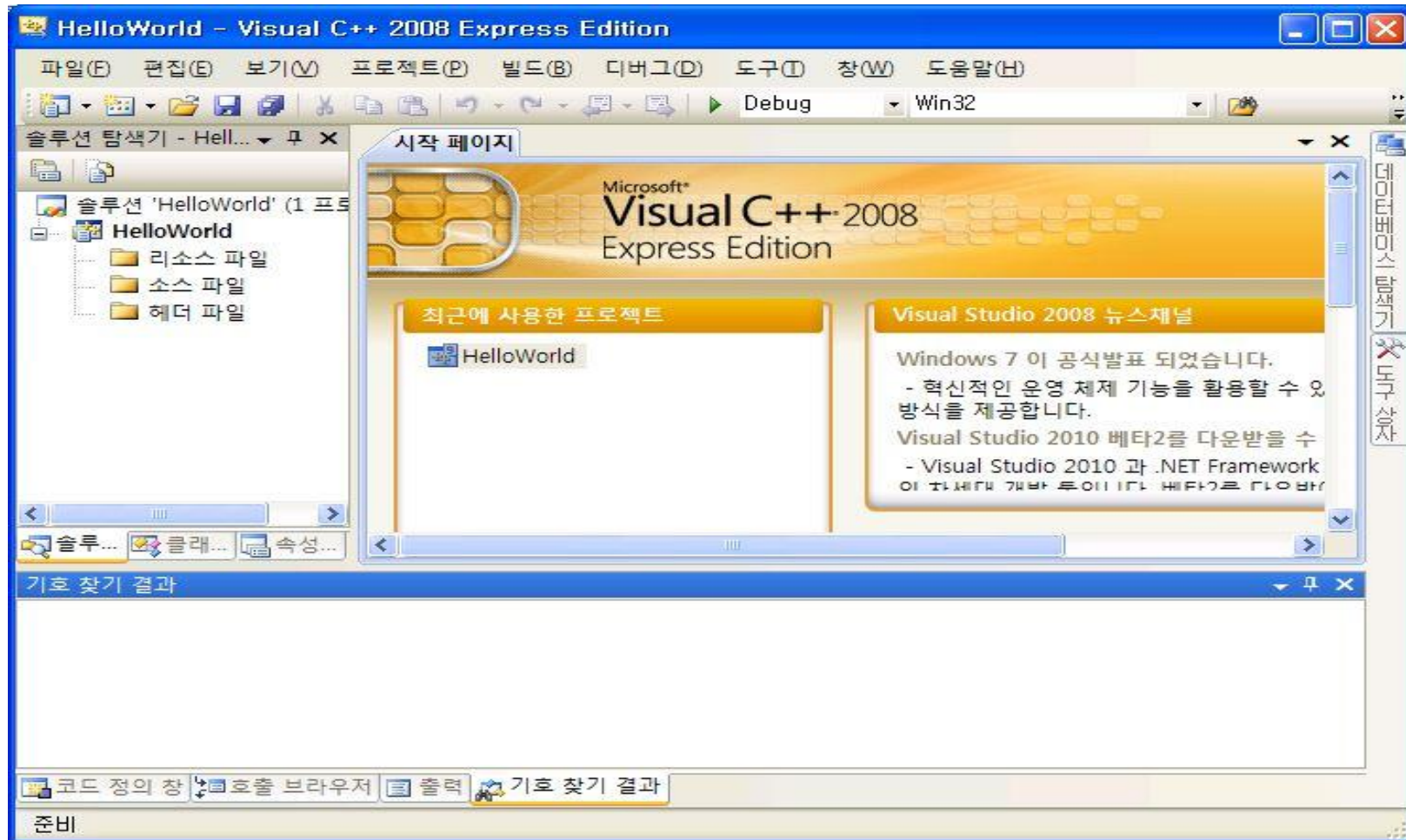
## 1.3 프로그램 작성 방법 4단계 실습 – Step1 (4/9)

▶ 프로그램 작성 단계 – 소스파일 (Hello.c)을 만들어 보자!



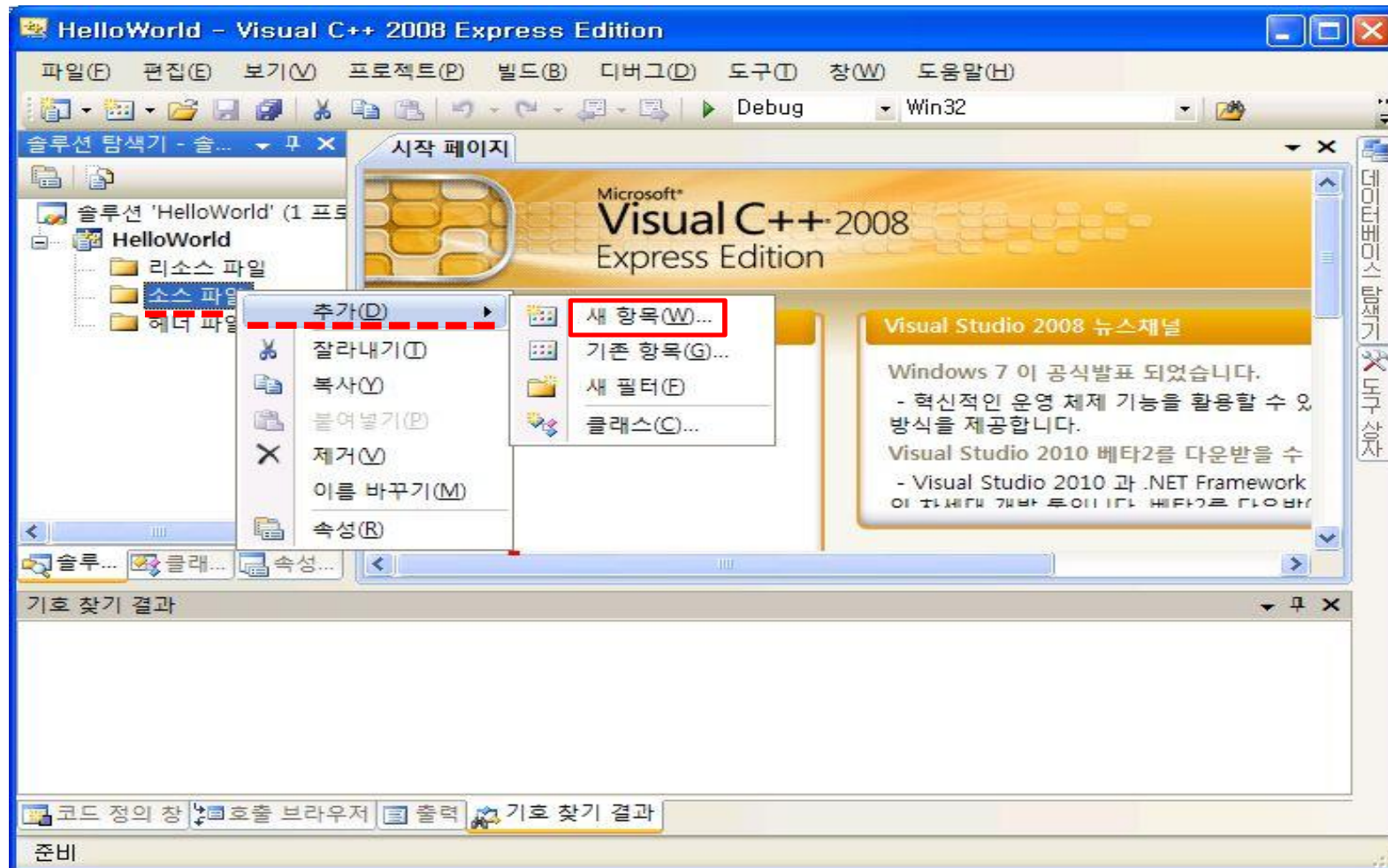
## 1.3 프로그램 작성 방법 4단계 실습 – Step1 (5/9)

▶ 프로그램 작성 단계 – 소스파일 (Hello.c)을 만들어 보자!



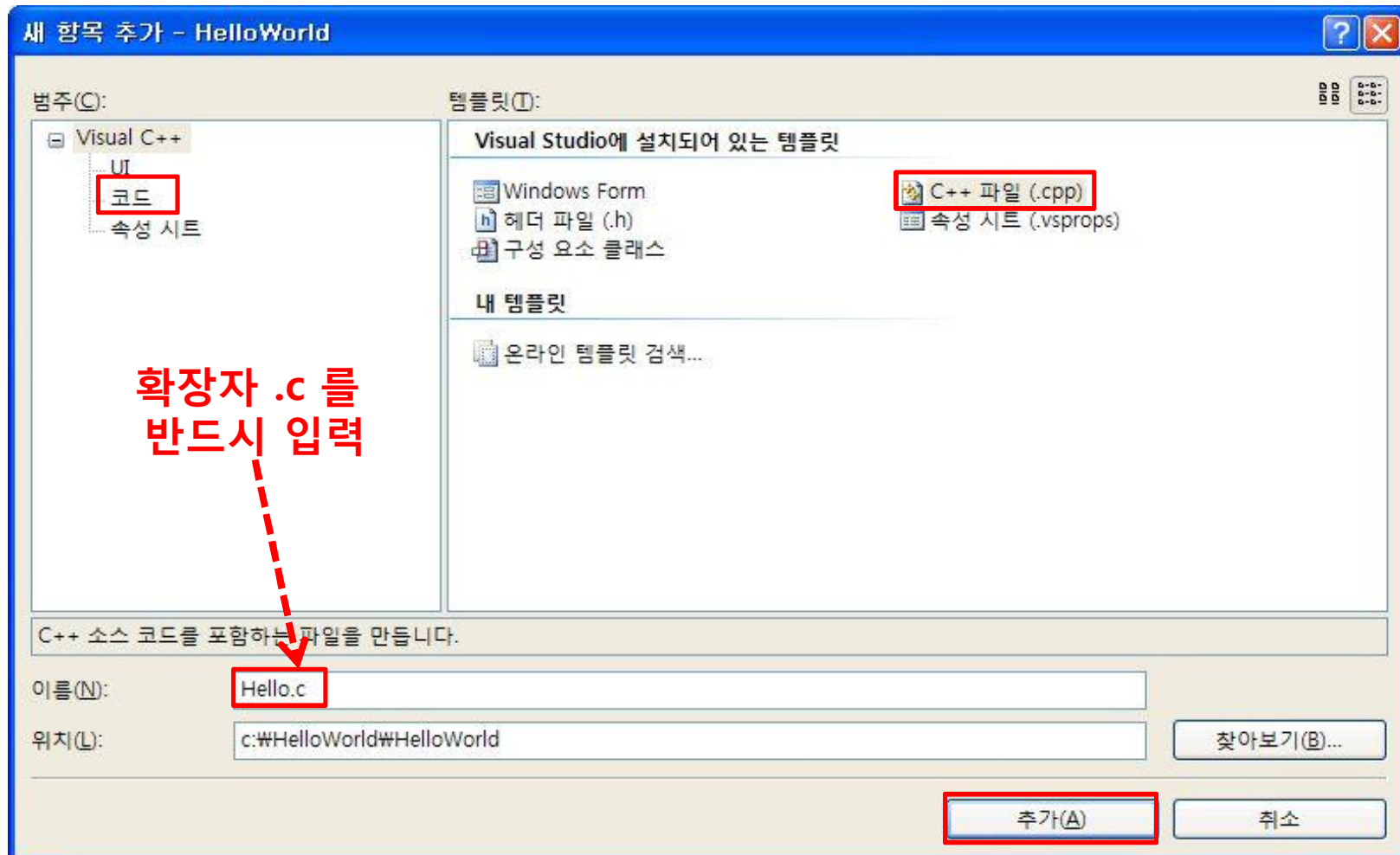
## 1.3 프로그램 작성 방법 4단계 실습 – Step1 (6/9)

▶ 프로그램 작성 단계 – 소스파일 (Hello.c)을 만들어 보자!



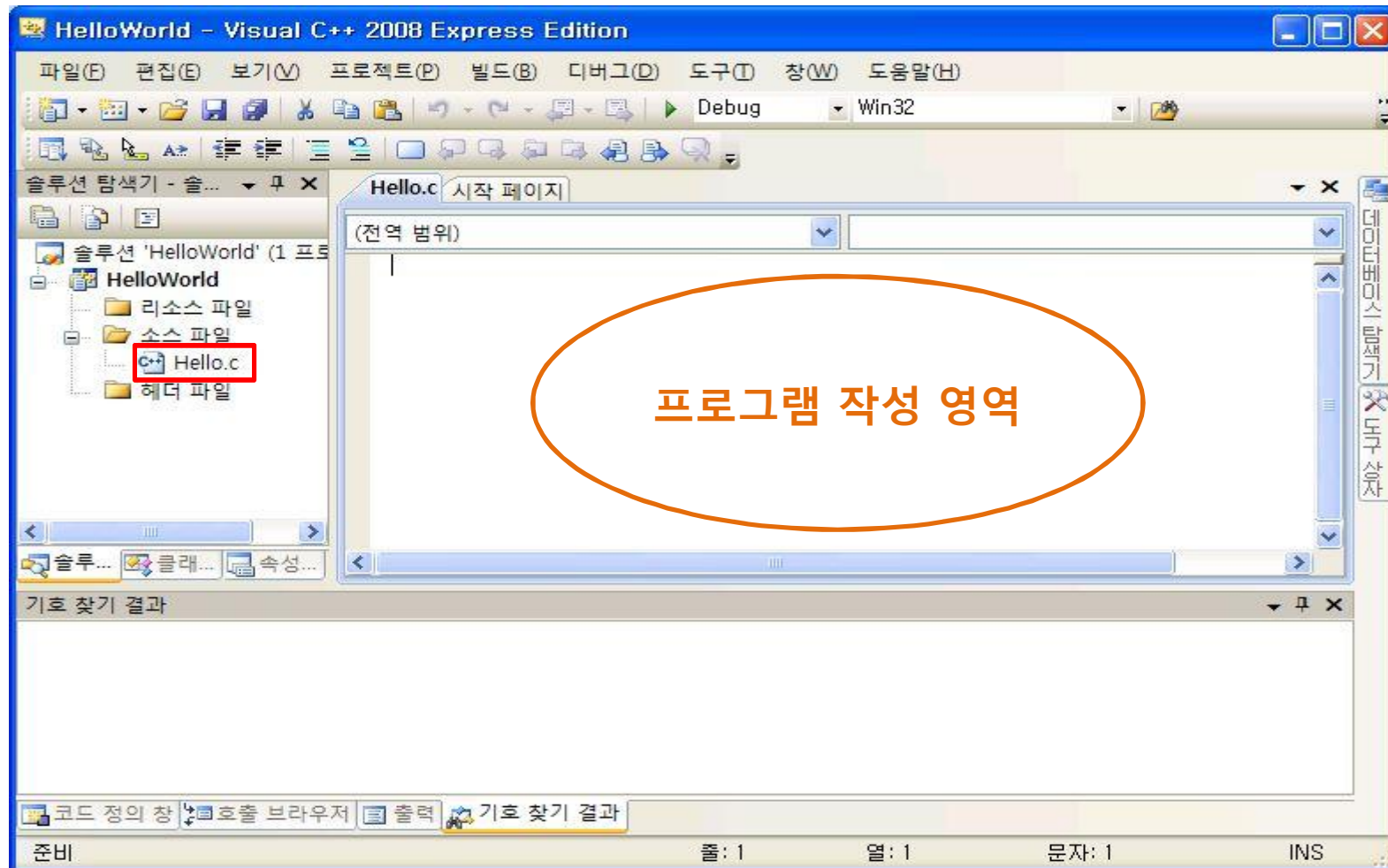
## 1.3 프로그램 작성 방법 4단계 실습 – Step1 (7/9)

▶ 프로그램 작성 단계 – 소스파일 (Hello.c)을 만들어 보자!



## 1.3 프로그램 작성 방법 4단계 실습 – Step1 (8/9)

▶ 프로그램 작성 단계 – 소스파일 (Hello.c)을 만들어 보자!



## 1.3 프로그램 작성 방법 4단계 실습 – Step1 (9/9)

▶ 프로그램 작성 단계 – 소스파일 (Hello.c)을 만들어 보자!

```
#include<stdio.h>
int main(void)
{
    printf("Hello C world \n");

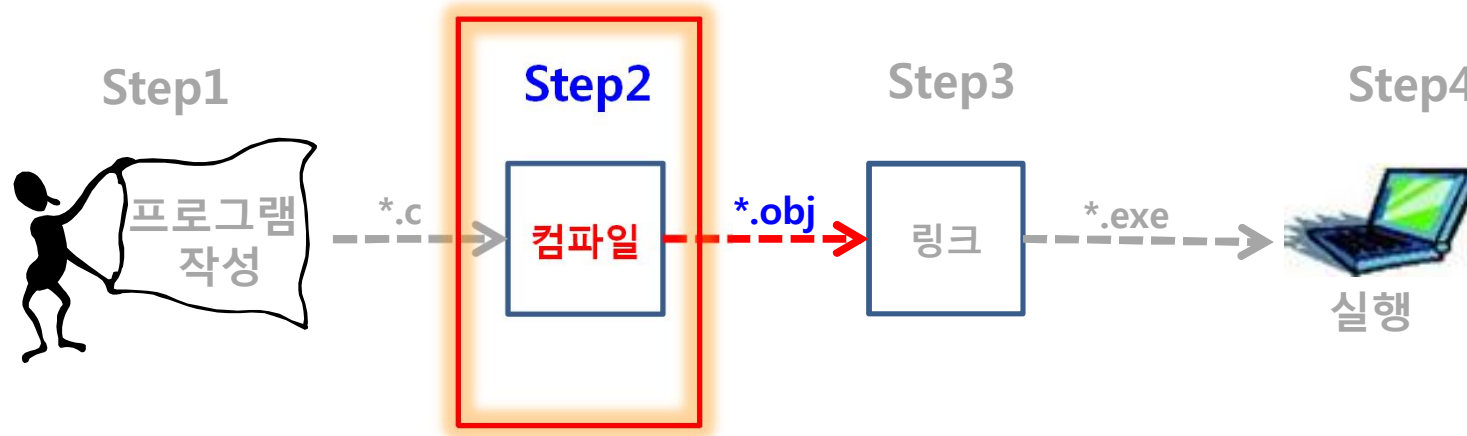
    return 0;
}
```



– 프로그램 작성 영역에 입력

## 1.3 프로그램 작성 방법 4단계 - 이론 & 실습

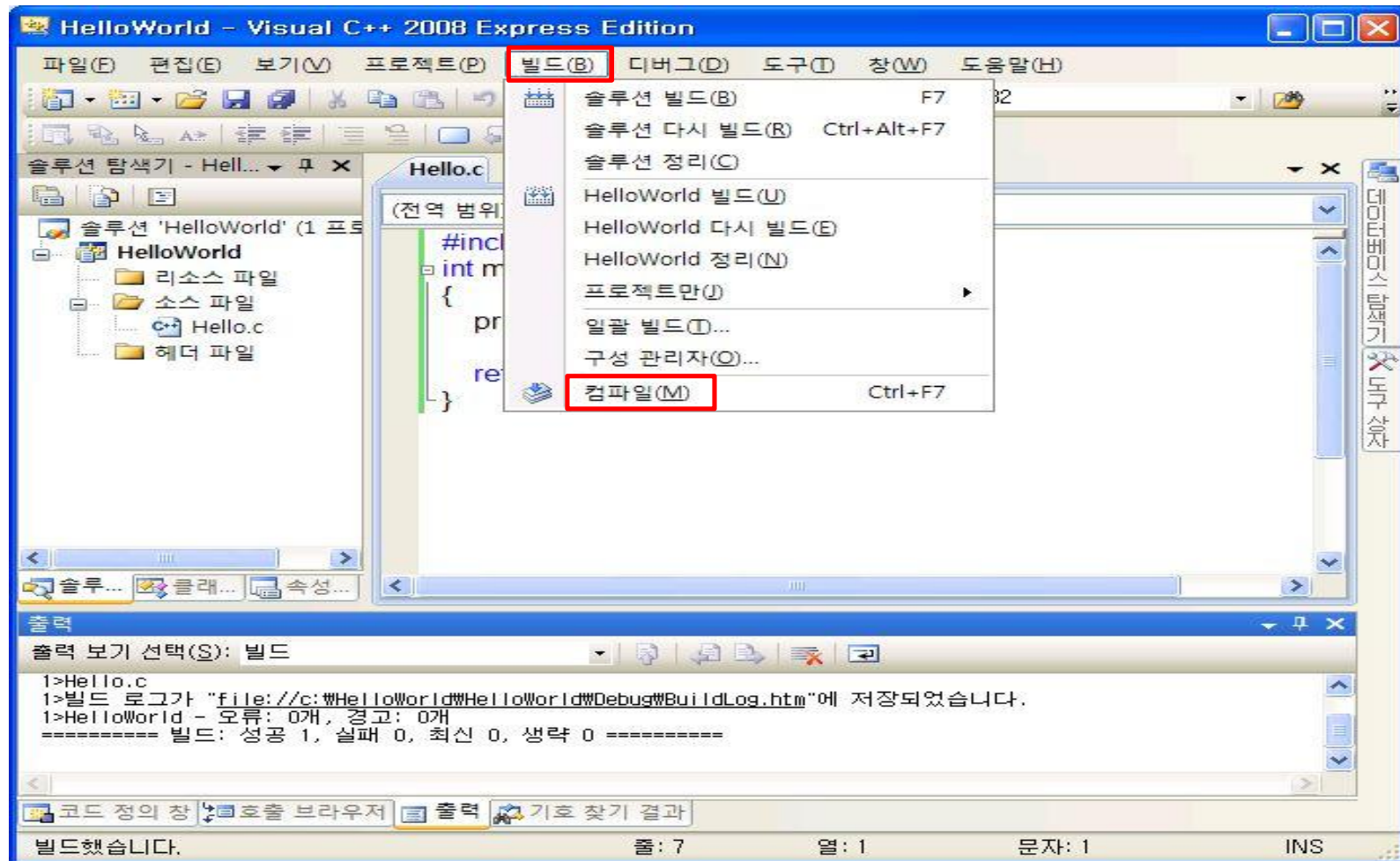
### ▶ 프로그램 작성 방법 4 단계 - 실습





## 1.3 프로그램 작성 방법 4단계 실습 – Step2 (1/2)

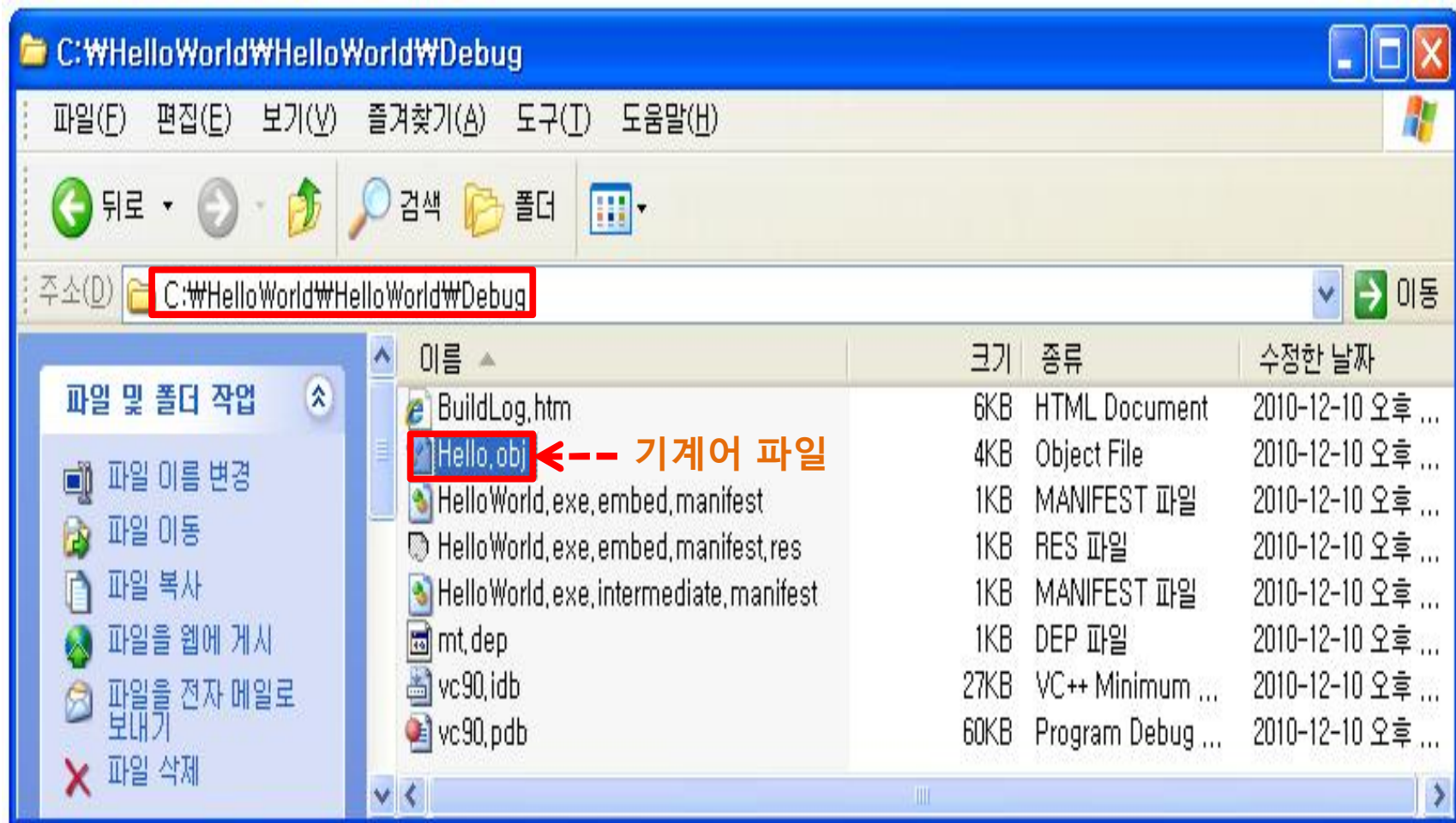
▶ 컴파일 단계 – 오브젝트 파일 (Hello.obj)을 만들어 보자!





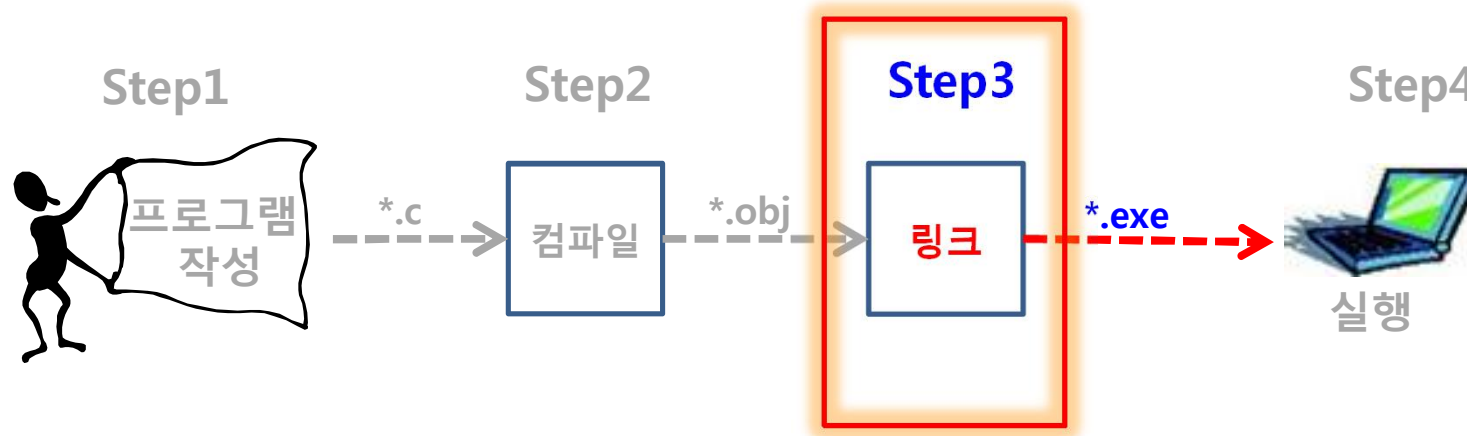
## 1.3 프로그램 작성 방법 4단계 실습 – Step2 (2/2)

▶ 컴파일 단계 – 오브젝트 파일 (Hello.obj)을 만들어 보자!



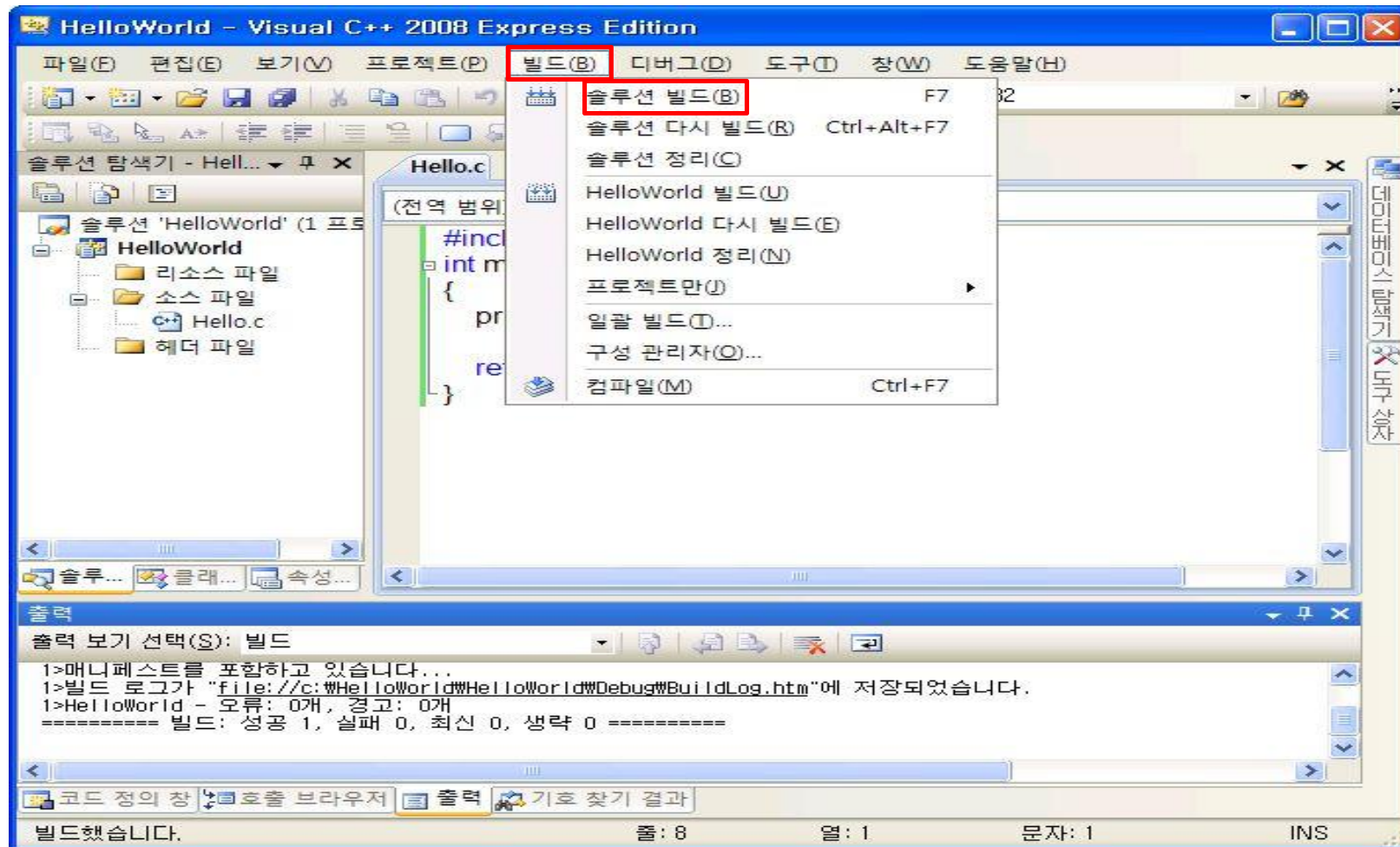
## 1.3 프로그램 작성 방법 4단계 - 이론 & 실습

### ▶ 프로그램 작성 방법 4 단계 - 실습



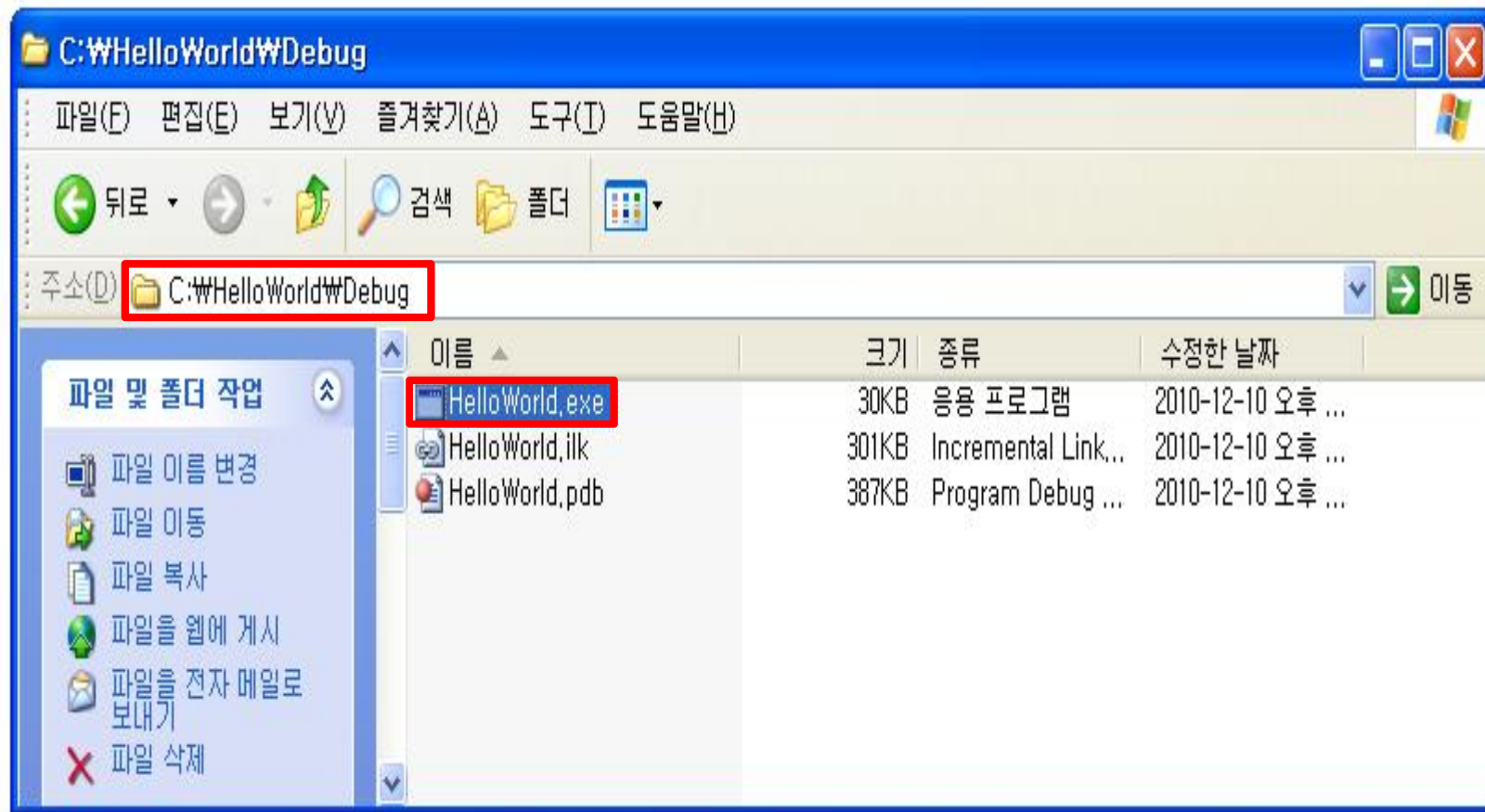
## 1.3 프로그램 작성 방법 4단계 실습 – Step3 (1/2)

▶ 링크 단계 – 실행 파일 (HelloWorld.exe)을 만들어 보자!



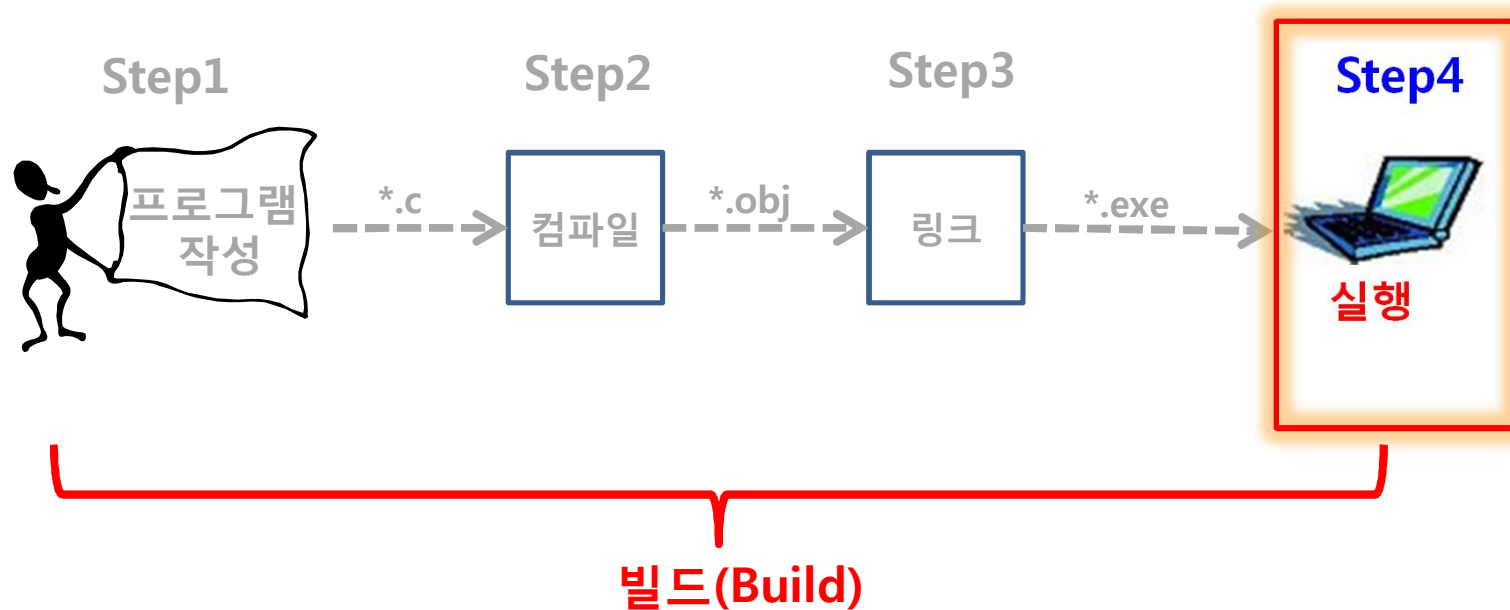
## 1.3 프로그램 작성 방법 4단계 실습 – Step3 (2/2)

▶ 링크 단계 – 실행 파일 (HelloWorld.exe)을 만들어 보자!



## 1.3 프로그램 작성 방법 4단계 - 이론 & 실습

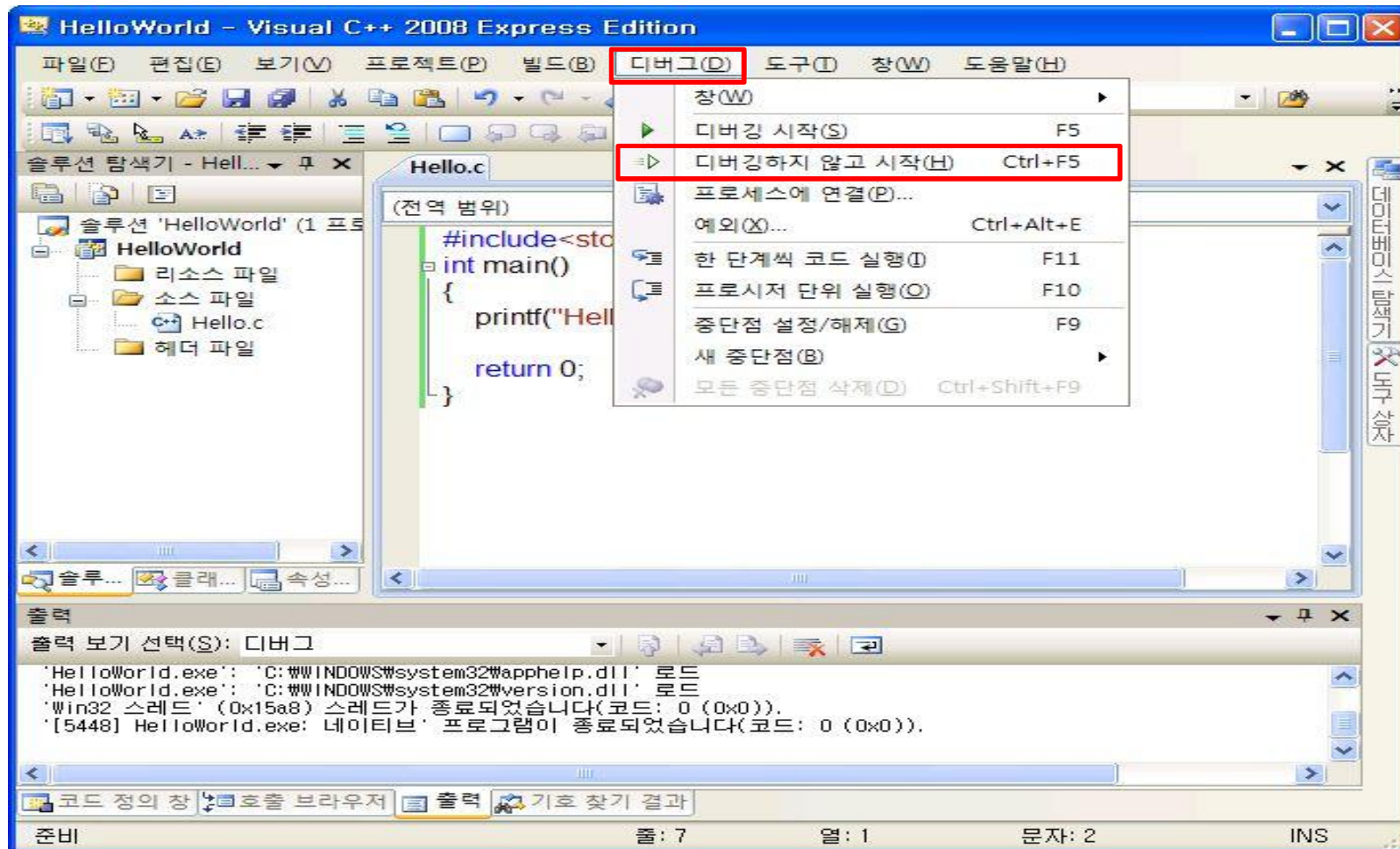
### ▶ 프로그램 작성 방법 4 단계 - 실습





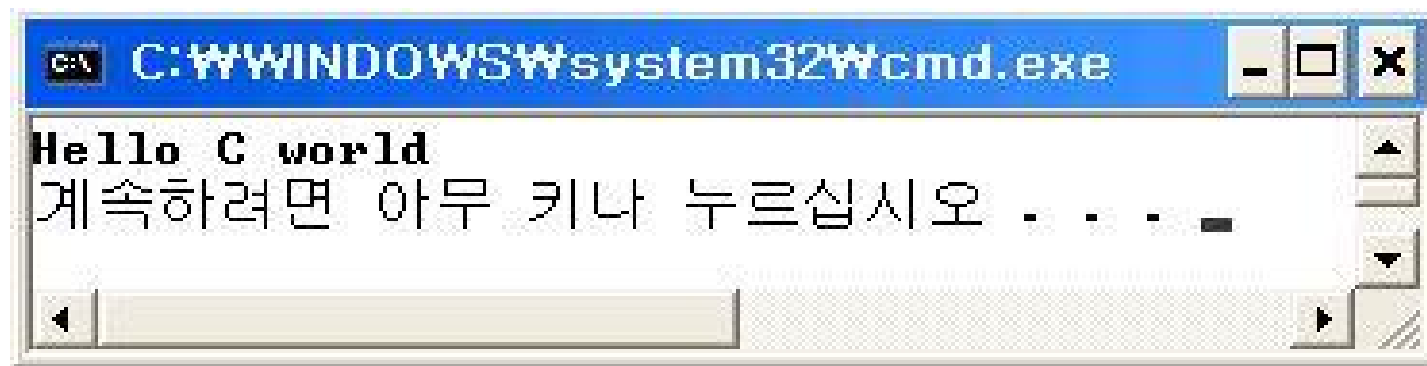
## 1.3 프로그램 작성 방법 4단계 실습 – Step4 (1/2)

▶ 실행 단계 – 실행 파일 (HelloWorld.exe)을 실행시키자!



## 1.3 프로그램 작성 방법 4단계 **실습** - **Step4** (2/2)

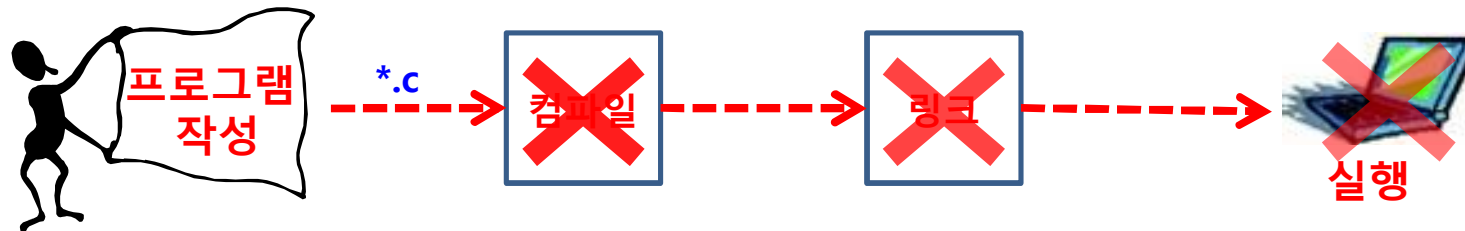
▶ **실행** 단계 - **실행** 파일 (HelloWorld.exe)을 실행시키자!



## 1.4 오류의 종류 (컴파일오류 & 런타임오류) - (1/4)

### ▶ 컴파일(compile) 오류 - 이론

- ✓ 구문(syntax) 오류
  - 통합개발SW에서 **자동으로 감지**
  - **[F4]**키를 이용하여 '확인'



**[F4] 키로 '에러'를 확인하자**




## 1.4 오류의 종류 (컴파일오류 & 런타임오류) - (2/4)

---

### ▶ 컴파일(compile) 오류 - 실습

```
#include<stdio.h>
int main(void)
{
    printf("Hello C world \n");
    return 0;
}
```



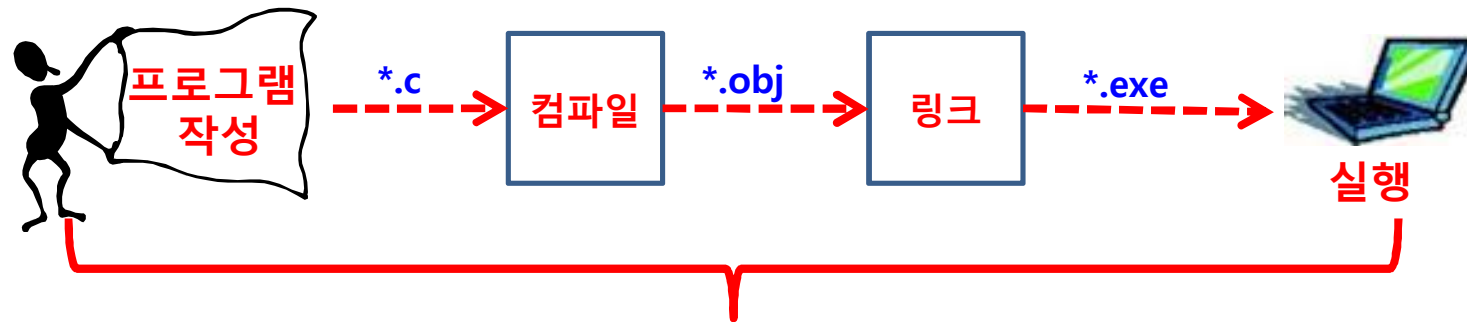
1. 세미콜론을 삭제하고 컴파일 해보자
2. [F4]를 눌러 확인해보자

## 1.4 오류의 종류 (컴파일오류 & 런타임오류) - (3/4)

### ▶ 런타임(run-time) 오류 - 이론

✓ 실행오류

- 통합개발 SW에서 자동 감지하지 못함- 수동으로 개발자가 해야 함



빌드(Build)가능, 그러나 비정상적인 실행



∴ 디버깅모드 필요

## 1.4 오류의 종류 (컴파일오류 & 런타임오류) - (4/4)

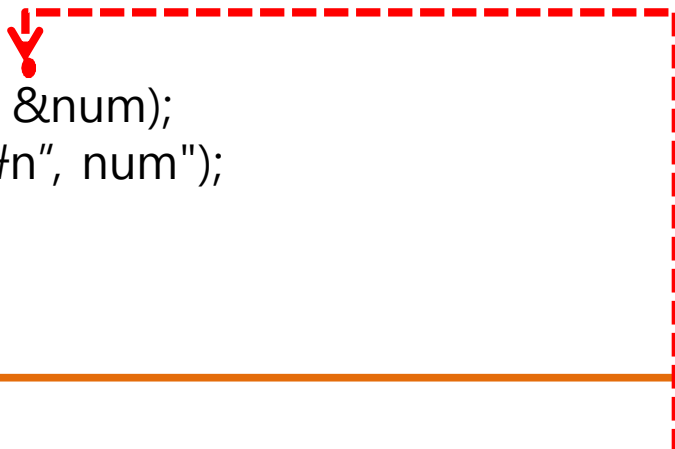
---

### ▶ 런타임(run-time) 오류 - 실습

- 디버깅모드
  - 메뉴에서 [디버그]/[디버깅 시작]

```
#include <stdio.h>
int main(void)
{
    int num;
    scanf("%d", &num);
    printf("%d\n", num);

    return 0;
}
```



1. **&를 삭제**하고 컴파일&빌드 (비정상적 실행)
2. **[디버그]/[디버깅 시작]**을 눌러서 확인

## 1.5 디버깅 실습 - (1/6)

---

### ▶ 디버깅(debugging)이란?

- ✓ **실세계:** 벌레(bug)를 잡는 것
- ✓ **컴세계:** 오류(fault)를 잡는 것
- ✓ **목적**
  - 런타임 오류 검사
  - 복잡한 코드의 변수 값이나 주소 검사

### ▶ 디버깅 모드의 바로 가기 키

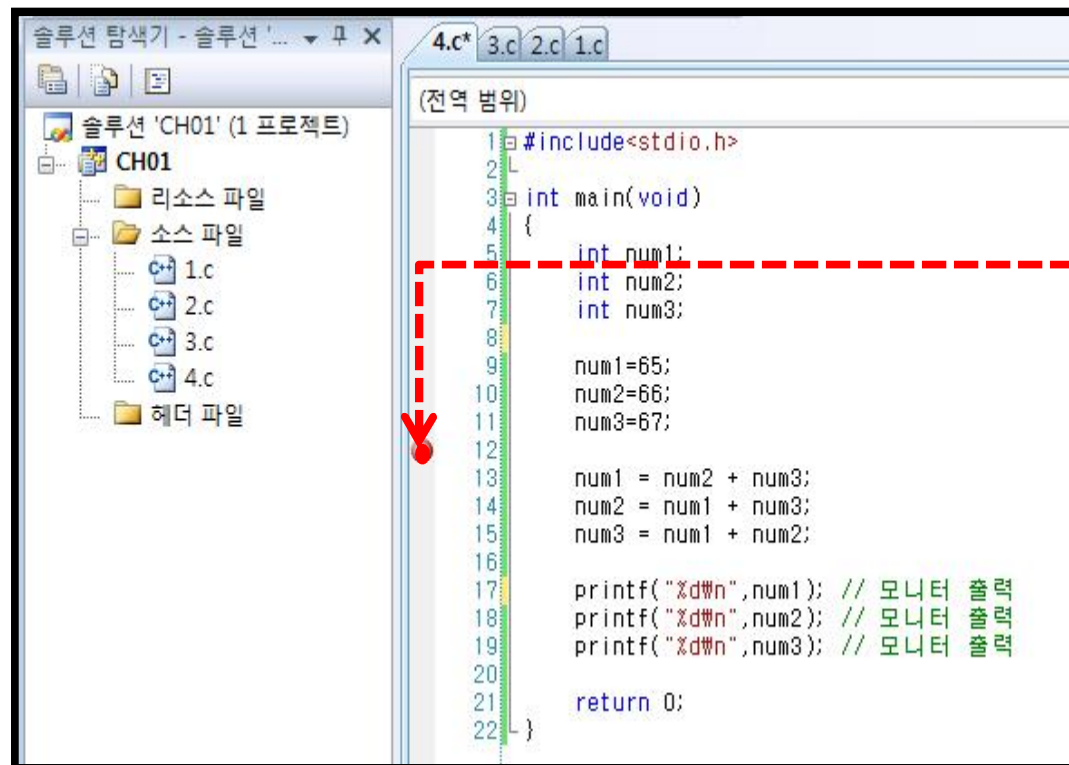
- ✓ 디버깅모드 [F5]
- ✓ 디버깅중지 [Shift + F5]
- ✓ 중단점 설정 및 해제 [F9]
- ✓ 한줄씩 실행 [F10]
- ✓ 함수 안으로 진입 [F11]

### ▶ 디버깅 모드에서 변수 값이나 주소 검사

- ✓ [메뉴]-[디버그]-[창]-[자동]
- ✓ [메뉴]-[디버그]-[창]-[조사식]
- ✓ [메뉴]-[디버그]-[창]-[메모리]

## 1.5 디버깅 실습 - (2/6)

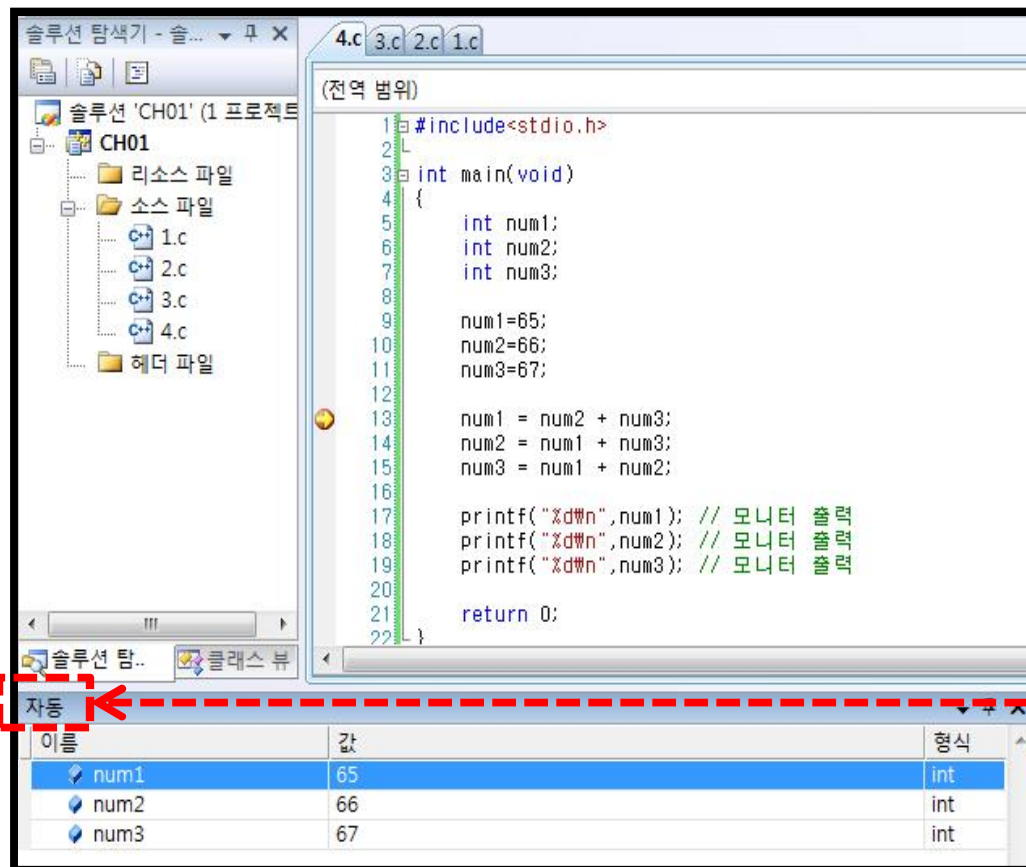
### ▶ 디버깅 모드에서 변수 값 검사



1. 12행에서 중단점 설정 - [F9]
2. [메뉴]-[디버그]-[디버깅시작]

## 1.5 디버깅 실습 - (3/6)

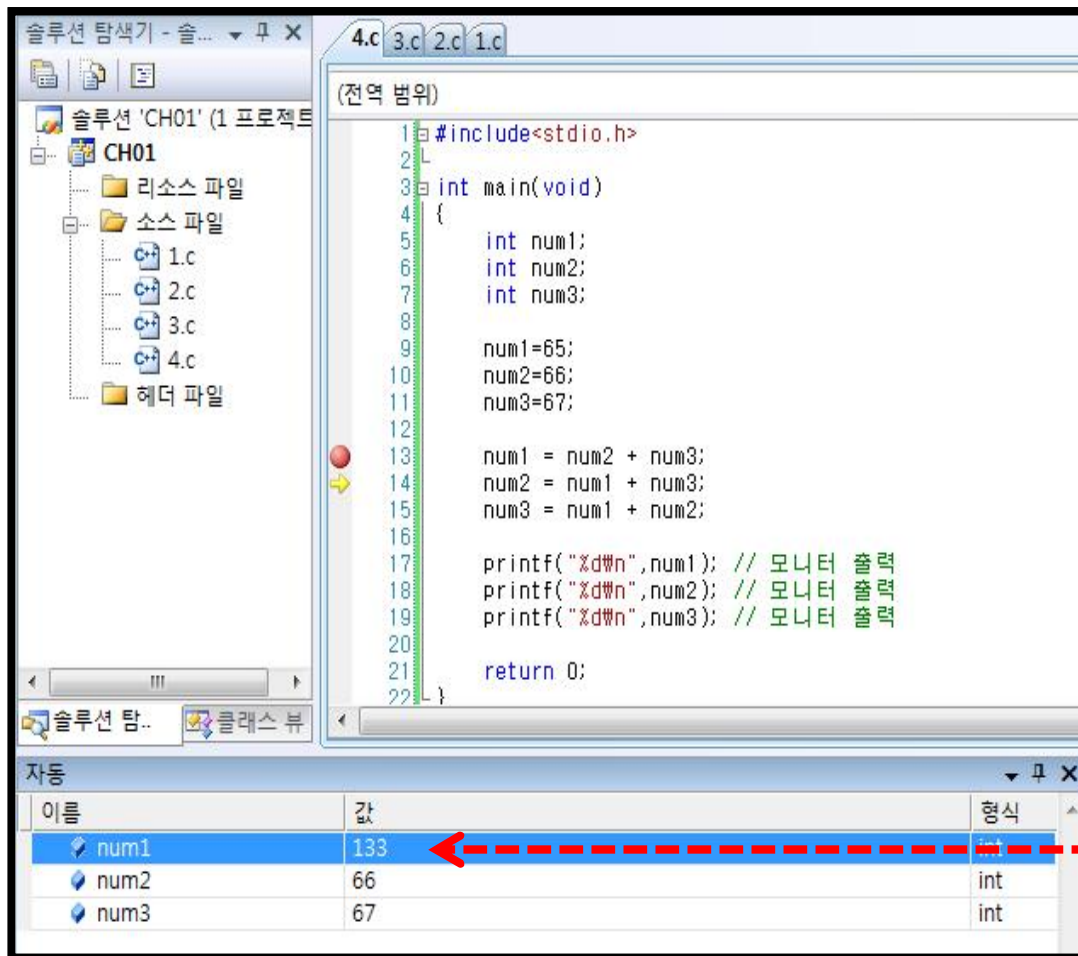
### ▶ 디버깅 모드에서 변수 값 검사



3. [메뉴]-[디버그]-[창]-[자동]

## 1.5 디버깅 실습 - (4/6)

### ▶ 디버깅 모드에서 변수 값 검사



The screenshot shows a C++ IDE with a project named 'CH01'. The source code is displayed in the main editor window, and the variable watch window is open at the bottom. The code defines three integers: num1, num2, and num3. num1 is initialized to 65, num2 to 66, and num3 to 67. The program then calculates num1 = num2 + num3, num2 = num1 + num3, and num3 = num1 + num2. Finally, it prints the values of num1, num2, and num3. The variable watch window shows the current values of the variables: num1 is 133, num2 is 66, and num3 is 67. A red arrow points to the value of num1 in the watch window.

```
1 #include<stdio.h>
2
3 int main(void)
4 {
5     int num1;
6     int num2;
7     int num3;
8
9     num1=65;
10    num2=66;
11    num3=67;
12
13    num1 = num2 + num3;
14    num2 = num1 + num3;
15    num3 = num1 + num2;
16
17    printf("%d\n",num1); // 모니터 출력
18    printf("%d\n",num2); // 모니터 출력
19    printf("%d\n",num3); // 모니터 출력
20
21    return 0;
22 }
```

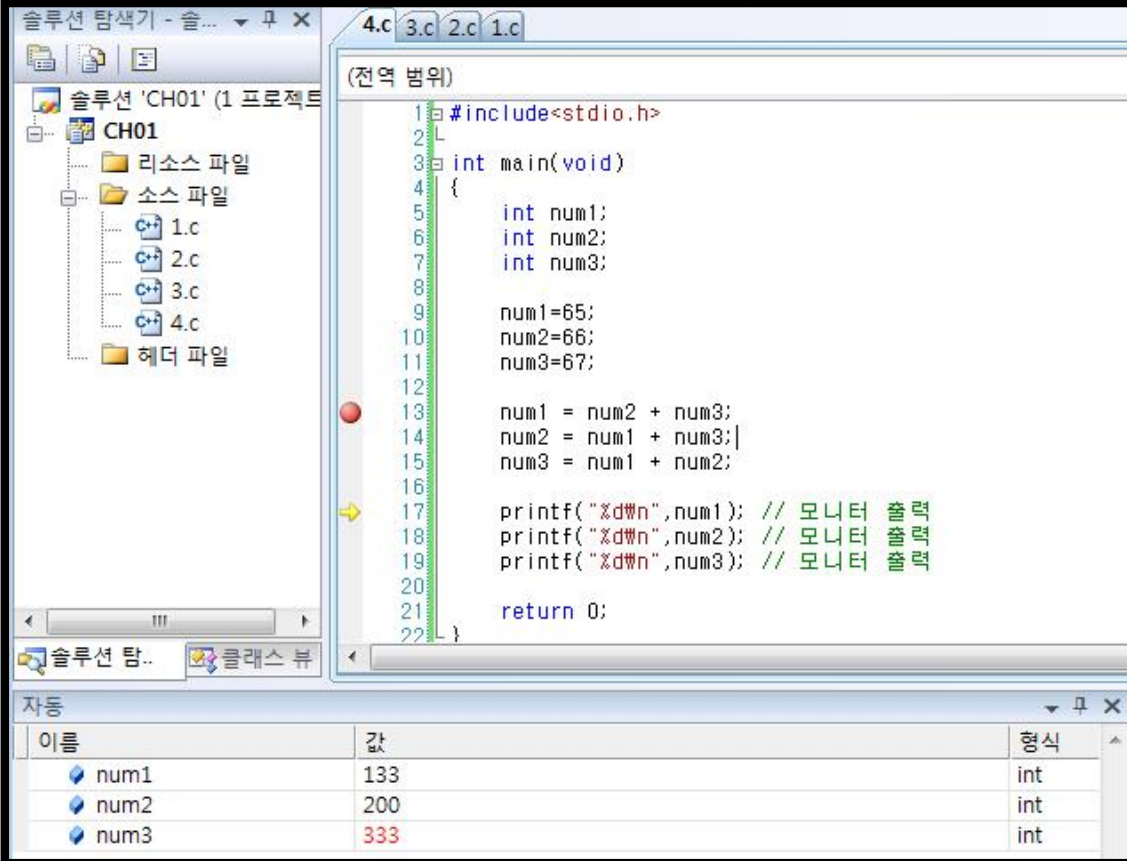
이름	값	형식
num1	133	int
num2	66	int
num3	67	int

4. 한줄씩 실행 - [F10]

num1의 값이 변경됨

## 1.5 디버깅 실습 - (5/6)

### ▶ 디버깅 모드에서 변수 값 검사



The screenshot shows a C++ IDE with a project named 'CH01'. The source code is displayed in the main editor, and the 'Variables' window is open at the bottom. The code is as follows:

```
(전역 범위)
1 #include<stdio.h>
2
3 int main(void)
4 {
5     int num1;
6     int num2;
7     int num3;
8
9     num1=65;
10    num2=66;
11    num3=67;
12
13    num1 = num2 + num3;
14    num2 = num1 + num3;
15    num3 = num1 + num2;
16
17    printf("%d\n",num1); // 모니터 출력
18    printf("%d\n",num2); // 모니터 출력
19    printf("%d\n",num3); // 모니터 출력
20
21    return 0;
22 }
```

The 'Variables' window at the bottom shows the current values of the variables:

이름	값	형식
num1	133	int
num2	200	int
num3	333	int

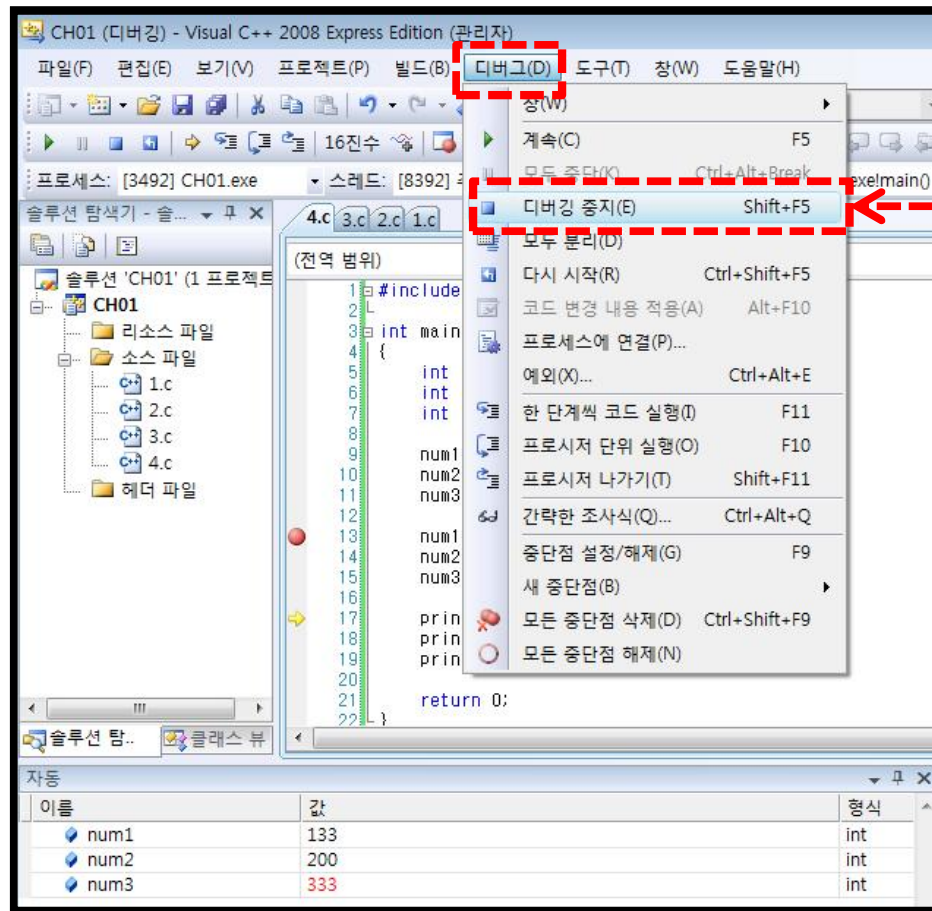
5. 계속 한줄씩 실행 - [F10]

num1,num2, num3의 값을 확인할 수 있음



## 1.5 디버깅 실습 - (6/6)

### ▶ 디버깅 모드에서 변수 값 검사



6. 디버깅모드 종료

## 1.6 C언어의 특징

- ▶ 이식성과 효율성이 높음
- ▶ 다른 프로그래밍 언어를 배우기에 좋음
- ▶ 지능형 서비스 로봇 제어
- ▶ 절차지향 언어

## 1.6 C언어의 학습 방식

- ▶ **1단계:** 교재에 있는 코드를 분석
- ▶ **2단계:** 분석된 내용을 통해 교재를 참고하지 않고 코드 작성
- ▶ **3단계:** 나만의 코드 작성 후, 교재의 코드와 비교

# 공부한 내용 떠올리기

- ▶ C언어의 의미와 C언어의 탄생
- ▶ 컴파일러란 무엇인가
- ▶ 프로그램 작성 방법 4단계의 이론
- ▶ 프로그램 작성 방법 4단계의 실습
- ▶ 오류의 종류 (컴파일오류와 런타임오류)
- ▶ 디버깅 실습

# Quiz 1.

---

0. 프로그램 작성 4단계 이론을 쓰세요.
1. 새로운 프로젝트를 만들고, 프로그램 작성 4단계를 실습하세요.
2. 프로젝트에 여러 개의 .c파일을 생성하세요. (예, 1.c 부터 10.c 까지)
3. 강의를 통해 배운 실습파일을 이용하여, 디버깅 모드를 연습하세요.
4. 프로젝트에 1.c, 2.c, 3.c 파일이 있다고 가정하자. 현재 3.c 파일을 작성하고 있는데 1.c 와 2.c 파일에 main() 함수가 있으면 어떤 문제가 생기나요?
5. 4번의 문제를 해결하려면 어떻게 해야 하나요?
6. 프로젝트를 만들어놓은 폴더위치를 알아야 하는 것이 왜 중요한가?
7. 바탕화면에서 test.c 파일을 만들고, 기존 프로젝트에 추가하세요.
8. 파일에 확장자가 보이지 않는 경우, 이를 해결하세요.
9. 코드의 줄번호를 보이게 하려면 어떻게 해야 하는가?

---

복습은 그날 그날... ^^