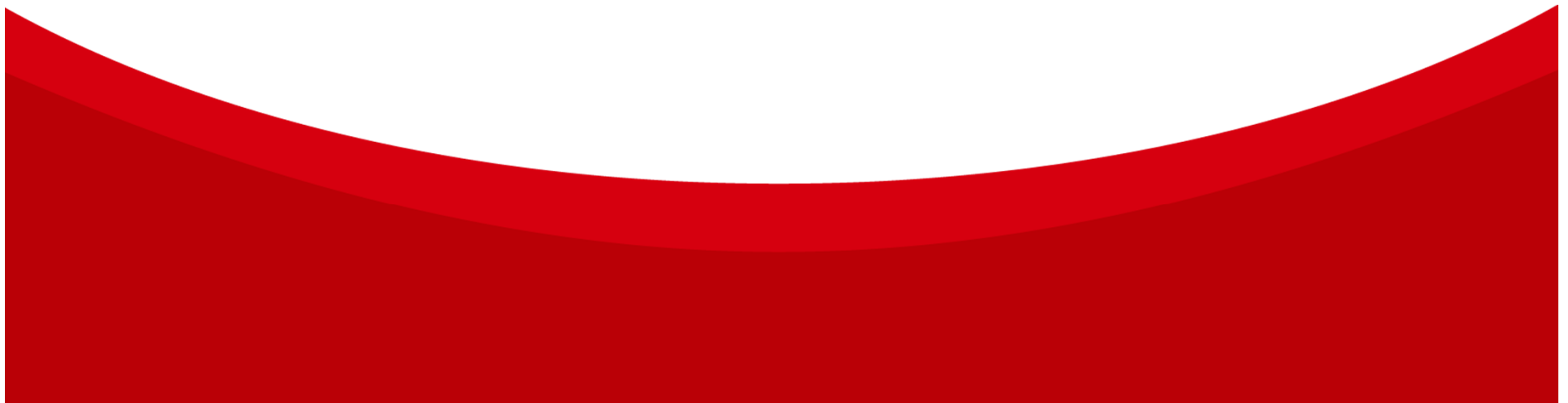


Chapter 4

주석문과 디버깅



01

Question

실수형 변수 두 개를 입력받아서 두 수의 합을 정수로 변환한 결과값과 두 수를 각각 정수로 변환하여 합을 구한 결과값을 출력하는 프로그램을 작성하고 프로그램 내용에 관한 설명을 주석으로 표시하십시오

입력 예 3.4 5.65

출력 예 9 8

소스

```
/*
이 프로그램은 주석의 사용에 대해 학습하기 위해 작성한 것입니다.
주석은 프로그램 작성자가 프로그램을 이해하기 쉽도록 하기 위해 작성하는 것입니다.
주석으로 표시된 부분은 프로그램을 컴파일 할 때 무시하고 처리합니다.
따라서 실제 프로그램에는 없는 것과 마찬가지입니다.
*/

#include <stdio.h>

int main()
{
    int a, b; /* 정수형 변수 두 개를 선언한다. */
    double d, e; /* 실수형 변수 두 개를 선언한다. */

    scanf("%lf %lf", &d, &e); /* 실수형 변수 두 개를 입력받는다. */

    a = d + e; /* 실수 두개를 더해서 정수 변수에 대입한다. */
}
```

Chapter 4

주석문과 디버깅

```
b = (int)d + (int)e; /* 두개의 실수를 각각 정수로 변환하여 정수 변수에 대입한다. */
printf("%d %d \n", a, b); /* 결과값을 출력하고 비교해 본다. */

return 0;
}

/*****
여러 줄을 주석 처리하는 방법입니다.
*****/
```

설명

- `/* 프로그램이나 코드에 대한 설명 */`

↑
주석의 시작

↑
주석의 끝

주석의 시작부분부터 끝부분 사이의 모든 문자열을 주석이라고 하며 이 부분은 실제 프로그램의 실행과는 전혀 상관이 없이 사용자가 프로그램의 구조나 의미등을 알아보기 쉽도록 표시하는 것이다. 여러 줄에 걸쳐 주석이 필요할 때에도 시작과 끝만 표시되면 모두 주석으로 인식한다.

- `a = d + e;`

실수를 정수 변수에 대입을 하면 자료형을 변환하여 소수 부분을 버리고 정수 부분만 저장된다. d와 e는 실수이므로 입력의 예에서 먼저 합을 구하면 9.05가 되며 정수형 변수인 a에 저장을 하면 형변환을 하여 9가 저장된다.

- `b = (int)d + (int)e;`

`(int)d`는 실수 d의 값을 정수로 변환하는 형변환 연산으로 d의 값 3.4에서 정수부분 3을 반환한다. 3 + 5의 값 8이 정수형 변수 b에 저장된다.

자가진단 1

2개의 실수를 입력 받아서 두 수의 곱을 정수로 변환한 결과값과 두 수를 각각 정수로 변환하여 곱을 구한 결과값을 출력하는 프로그램을 작성하고 프로그램 내용에 관한 설명을 주석으로 표시하시오.

입력 예 3.4 5.65

출력 예 19 15

02

Question

한 개의 정수를 입력받아 정수 4로 나눈 몫과 실수 4.0으로 나눈 결과값을 각각 출력하는 프로그램을 작성하고 프로그램 내용에 관한 설명을 주석으로 표시하시오. (단, 실수는 반올림하여 소수 둘째자리 까지 출력한다.)

입력 예

5

출력 예

$5 / 4 = 1$

$5 / 4.0 = 1.25$

소스

```
// 이것은 한줄을 주석처리하는 방법입니다.  
// 주석처리 하고자 하는 줄마다 "//" 표시를 해주어야 합니다.  
  
#include <stdio.h>  
  
int main()  
{  
    int a; // 정수형 변수 한 개를 선언한다.  
    scanf("%d", &a); // 정수를 입력받는다.  
  
    printf("%d / 4 = %d \n", a, a / 4);  
    // 입력받은 정수를 정수 4로 나눈 몫을 출력한다.  
  
    printf("%d / 4.0 = %.2lf \n", a, a / 4.0);  
    // 입력받은 정수를 실수 4.0으로 나눈 결과값을 출력한다.  
  
    return 0;  
}
```


설명

- // 주석내용

한 줄만 주석처리 할 때 사용하는 방법이며 “//” 표시부터 줄의 끝까지가 주석이 된다. 여러 줄의 주석을 처리하려면 줄마다 “//” 표시를 해야 한다.

- `printf("%d / 4.0 = %lf \n", a, a / 4.0);`

정수와 실수가 혼합된 연산을 실행하면 정수는 자동으로 실수로 형변환을 하여 연산이 된다. 예를 들어 a의 값이 5라고 하면 `5.0 / 4.0`을 하여 1.25가 된다.

`(double)a / 4`와 같이 작성을 해도 똑같은 결과가 된다.

자가진단 2

2개의 정수를 입력받아서 첫 번째 수를 두 번째 수로 나눈 몫을 출력하고 첫 번째 수를 double로 변환하여 두 번째 수로 나눈 값을 구한 후 반올림하여 소수 둘째자리까지 출력하는 프로그램을 작성하고 프로그램 내용에 관한 설명을 주석으로 표시하십시오.

입력 예 11 3

출력 예 3 3.67

03

Question

디버깅

아래 소스를 입력하고 한 줄씩 실행하면서 디버깅을 해 보시오.

Chapter 4

주석문과 디버깅

소스

```
/* ****
디버깅은 프로그램에서 논리적인 에러를 찾아내기 위해 사용합니다.
**** */


#include <stdio.h>

int main()
{
    int a = 10;
    int b;
    int c;
    scanf("%d", &b);
    c = a + b;
    printf("%d \n", c);
    return 0;
}
```

설명


- 프로그램에서 문법적인 에러는 컴파일 과정에서 에러가 발생하므로 즉시 확인하고 수정할 수 있지만 논리적인 에러는 쉽게 찾아내기 어렵다.
- 문법적인 에러란 프로그램 형식이 잘못 되어 아예 컴파일 과정에서 에러가 발생하여 프로그램이 실행되지 않는 경우를 말하며
- 논리적인 에러란 사용자가 부호를 잘못 쓰거나 순서를 잘못 정하는 등의 실수로 프로그램은 잘 실행되지만 원하는 결과값이 나오지 않는 경우를 말한다.
- 이러한 논리적 에러를 찾아내어 수정하는 작업을 디버깅이라 한다.

따라하기 <Code::blocks>

1 메뉴에서 “Debug” – “Step into” 또는 Debugger Toolbars 에서  단추를 누른다.

※ Debugger Toolbars 모양  (12.11)



그리고 “Debug” – “Debugging windows” – “Watches”를 체크하거나  단추를 누르고 “Watches”를 선택하면 “Watches” 창이 생기는데 이를 드래그하여 적당한 위치에 배치하고 변수 이름을 적어준다.

Chapter 4

주석문과 디버깅

The screenshot shows a C++ IDE with a file named `main.cpp`. The code is as follows:

```
1  #include <stdio.h>
2
3  int main()
4  {
5      int a = 10;
6      int b;
7      int c;
8      scanf("%d", &b);
9      c = a + b;
10     printf("%d \n", c);
11     return 0;
12 }
13
```

A yellow arrow points to line 5, indicating the current execution point. The `Watches` panel on the right shows the state of variables `a`, `b`, and `c`:

Variable	Value	Type
<code>a</code>	134513769	int
<code>b</code>	0	int
<code>c</code>	-1208193036	int

2 노란 화살표가 있는 부분이 다음에 실행하려는 위치이다.

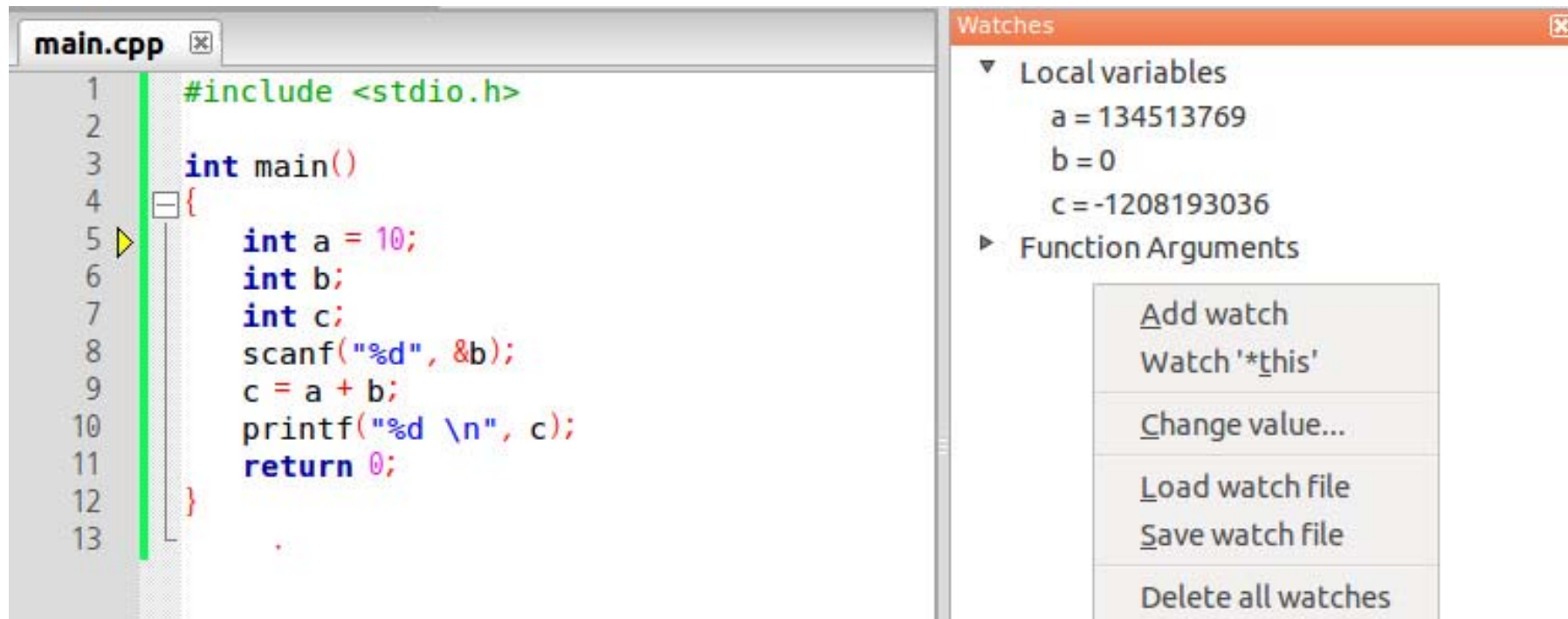
아직 아무것도 실행이 되지 않았으므로 변수의 값은 아직 쓰레기값이다.

쓰레기값이란 아무런 의미 없는 수를 의미하며 0이 되는 경우도 있기는 하지만 전혀 쓸모없는 값으로 저장되어 있는 것이다.

코드블록 10.05 버전에서는 지역변수가 기본적으로 나타난다. 만약 추가로 알고 싶은 것이 있으면 “Watches”창에서 마우스 오른쪽을 클릭하고 “Add watch”를 누르면 변수를 입력할 수 있다.

Chapter 4

주석문과 디버깅





The screenshot shows a C++ IDE with a file named `main.cpp`. The code is as follows:

```
1  #include <stdio.h>
2
3  int main()
4  {
5      int a = 10;
6      int b;
7      int c;
8      scanf("%d", &b);
9      c = a + b;
10     printf("%d \n", c);
11     return 0;
12 }
13
```

The debugger is active, with a yellow arrow pointing to line 5. The **Watches** window on the right displays the following:

- Local variables
 - a = 134513769
 - b = 0
 - c = -1208193036
- Function Arguments
 - Add watch
 - Watch '*this'
 - Change value...
 - Load watch file
 - Save watch file
 - Delete all watches

- 3 계속해서 "Debug" - "Step into" 또는 단축키 **Shift+F7** 또는  단추를 누르면 $a = 10$; 이 실행되어 a 의 값이 바뀌고 화살표는 다음 실행위치로 이동한 것을 알 수 있다. ("Step into" 대신 "Next line" 또는 단축키 **F7** 또는  단추를 눌러도 동일한 결과를 수행한다. "Step into"와 "Next line"은 여기에서는 같지만 함수를 호출할 때에는 서로 다른 동작을 한다. 이 부분은 함수 단원에서 다룰 것이다.)

Chapter 4

주석문과 디버깅

The screenshot shows a C++ IDE with a file named `main.cpp`. The code is as follows:

```
1  #include <stdio.h>
2
3  int main()
4  {
5      int a = 10;
6      int b;
7      int c;
8      scanf("%d", &b);
9      c = a + b;
10     printf("%d \n", c);
11     return 0;
12 }
13
```

A yellow arrow points to line 8, indicating the current execution point. The program has been executed, and the `Watches` window on the right shows the state of variables `a`, `b`, and `c`:

Variable	Value	Type
<code>a</code>	10	int
<code>b</code>	0	int
<code>c</code>	-1208193036	int

The value of `c` is incorrect because `b` was not assigned a value before being used in the calculation `c = a + b`.

Chapter 4

주석문과 디버깅

- 4 같은 키를 다시 한 번 누르면 입력창에서 커서가 깜박이며 입력을 기다린다. 값을 입력하면 b의 값이 입력된 값으로 바뀌고 화살표가 이동한다.



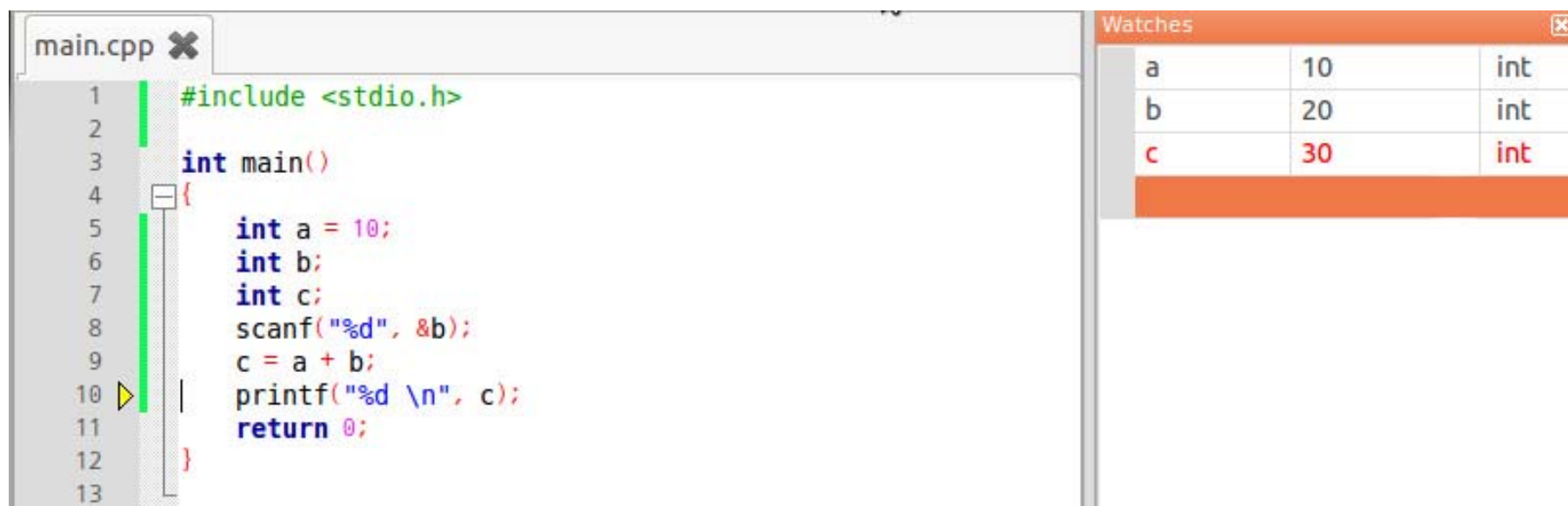
The screenshot shows a C++ IDE with a file named `main.cpp`. The code is as follows:

```
1 #include <stdio.h>
2
3 int main()
4 {
5     int a = 10;
6     int b;
7     int c;
8     scanf("%d", &b);
9     c = a + b;
10    printf("%d \n", c);
11    return 0;
12 }
13
```

The cursor is positioned at line 9. To the right, the **Watches** window displays the following data:

Variable	Value	Type
a	10	int
b	20	int
c	-1208193036	int

- 5 다시 키를 누르면 “ $c = a + b$;”를 실행하여 c 의 값이 a 와 b 의 합으로 바뀌는 것을 확인할 수 있다.





The screenshot shows a C++ IDE with a file named `main.cpp`. The code is as follows:

```
1 #include <stdio.h>
2
3 int main()
4 {
5     int a = 10;
6     int b;
7     int c;
8     scanf("%d", &b);
9     c = a + b;
10    printf("%d \n", c);
11    return 0;
12 }
13
```

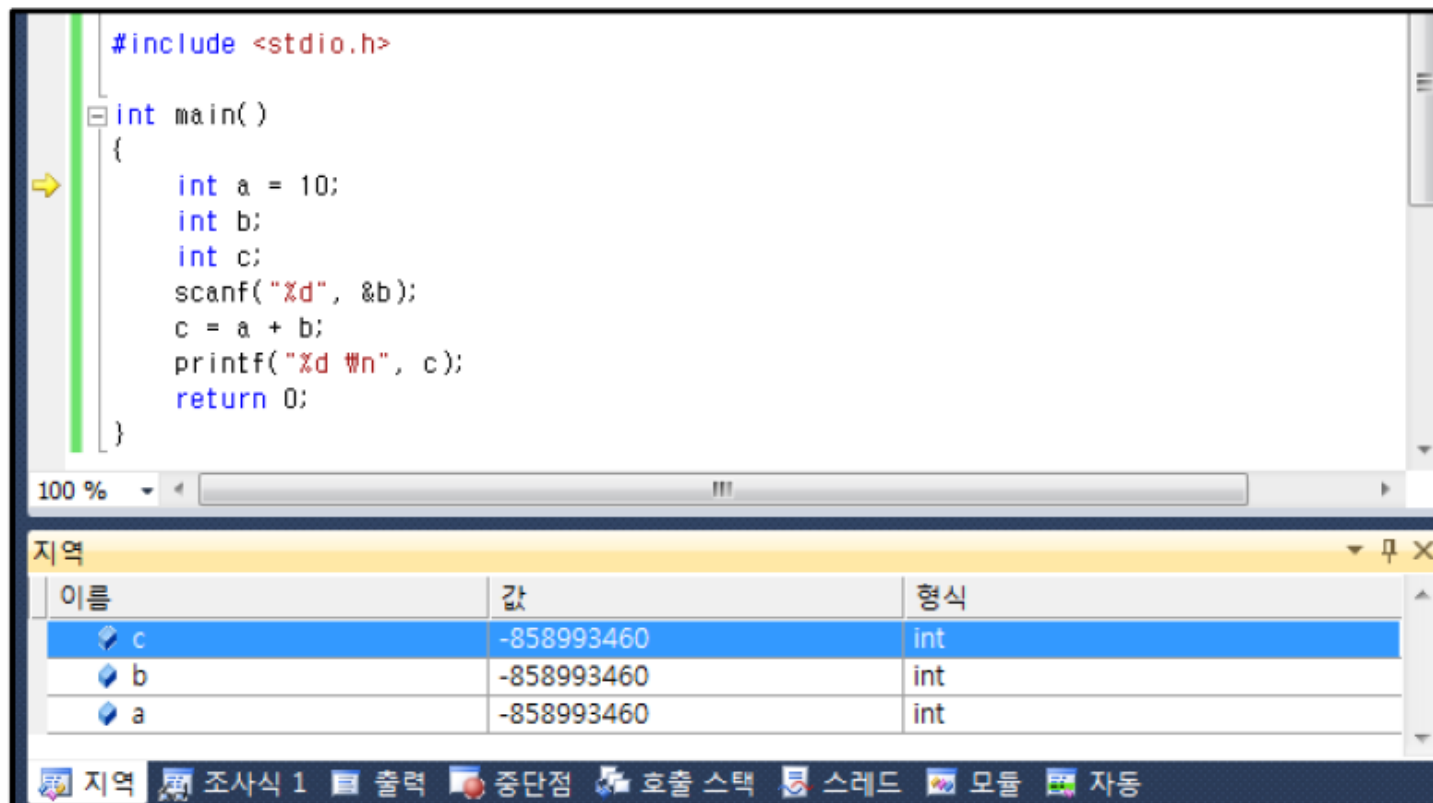
The cursor is positioned at line 10. To the right, the 'Watches' panel displays the current values of variables `a`, `b`, and `c`:

Variable	Value	Type
<code>a</code>	10	int
<code>b</code>	20	int
<code>c</code>	30	int

- 6 “Run to cursor” 또는 `F4` 또는  단추를 누르면 커서 이전까지의 행이 실행되고 커서가 있는 위치로 화살표가 이동한다.
- 7 “Stop debugger” 또는 `Shift+F8` 또는  단추를 누르면 디버깅 모드가 끝난다.

따라하기 <Visual Studio>

- 1 F10을 두 번 누른다. 화살표가 다음에 실행할 행에 위치해 있고 지역창에 a, b, c의 값이 표시된다. 현재까지는 각 변수의 값이 정해지지 않았으므로 쓰레기값이 들어있다.



Chapter 4

주석문과 디버깅

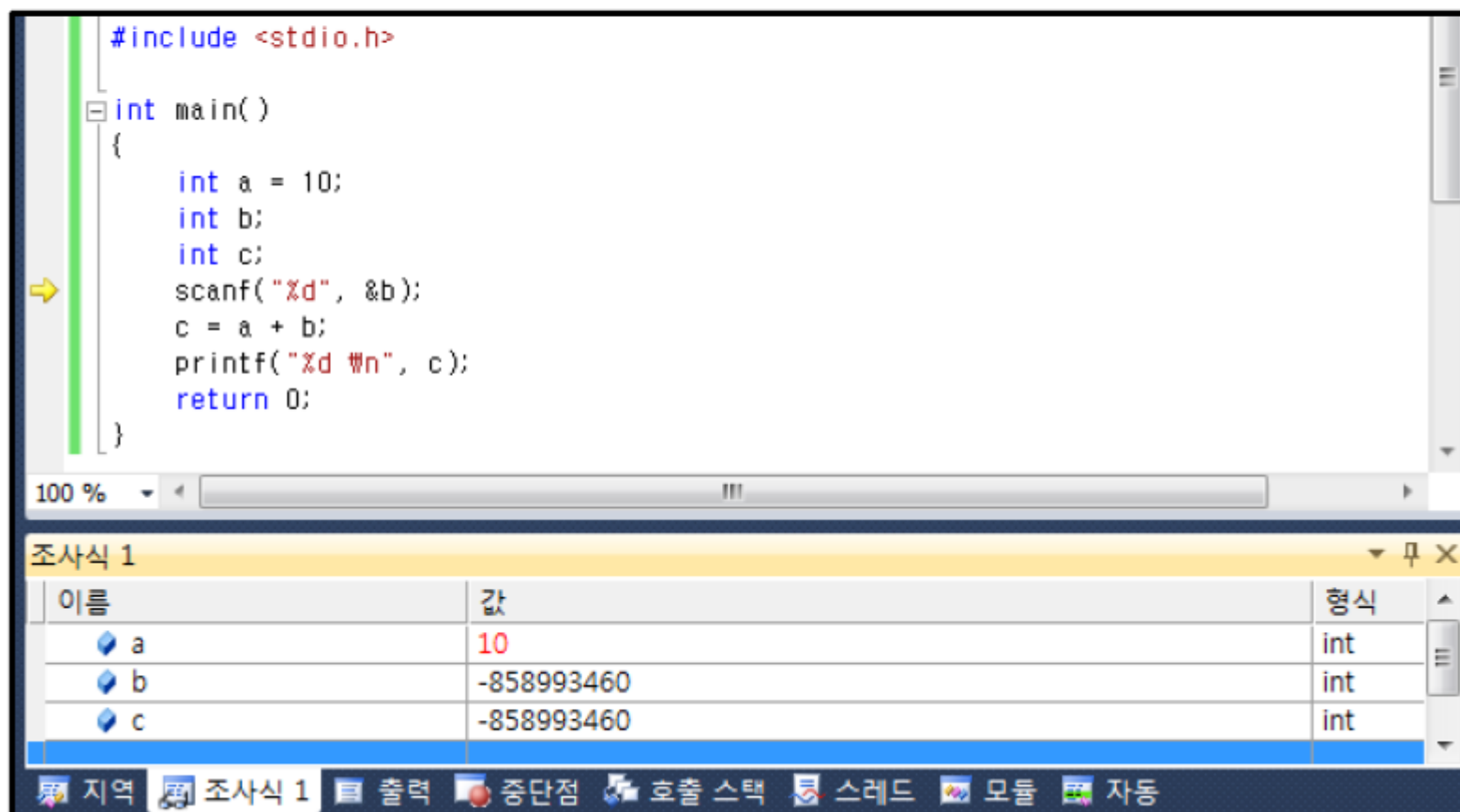
- 2 조사식 창에 a, b, c를 적어 넣어도 같은 내용을 확인할 수 있다. 지역창이나 조사식 창은 메뉴의 "디버그" → "창"에서 추가할 수 있으며 드래그하여 위치를 변경할 수도 있다.

이름	값	형식
a	-858993460	int
b	-858993460	int
c	-858993460	int

Chapter 4


주석문과 디버깅

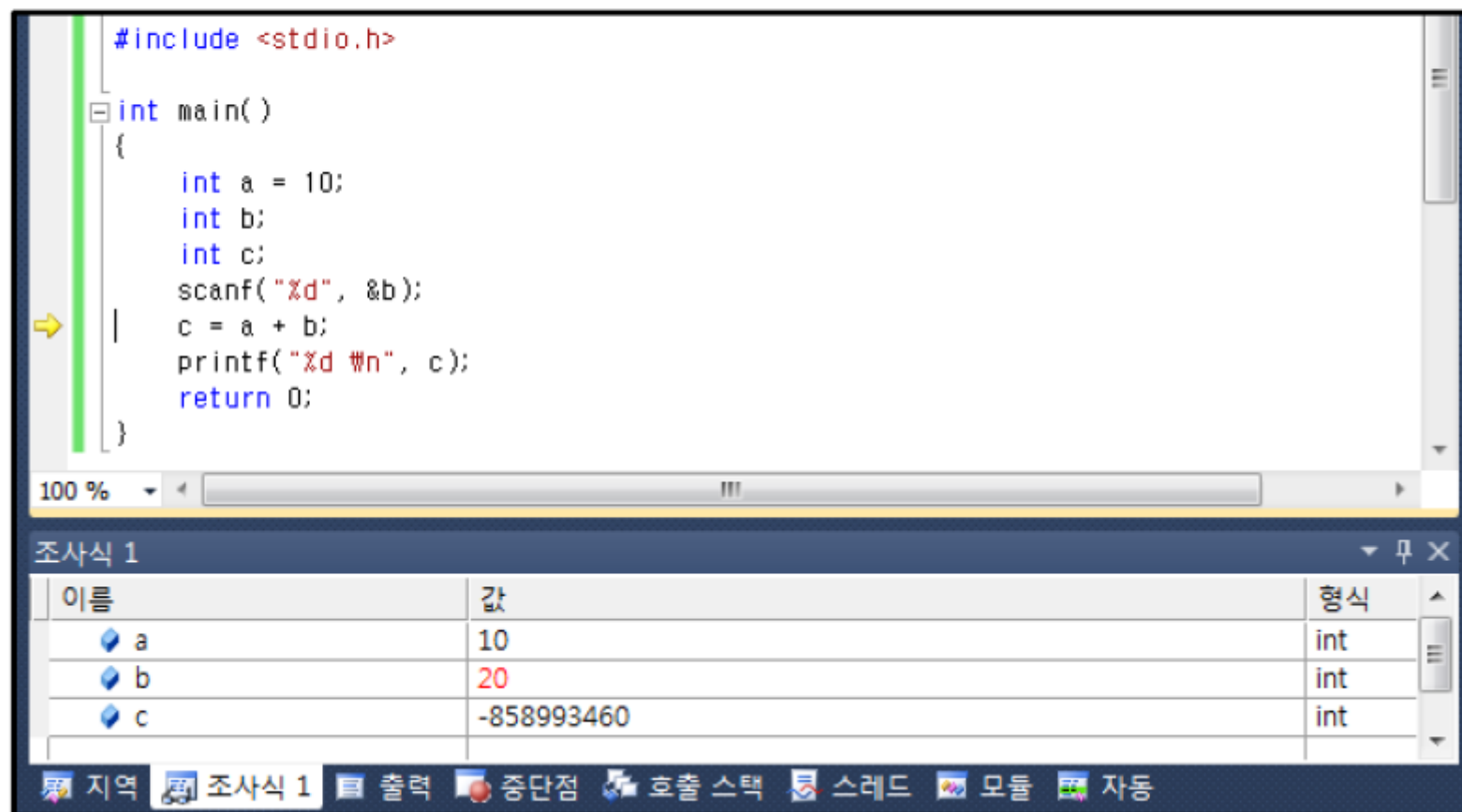
- 3 다시 F10을 누르면 $a = 10$; 이 실행되어 a 의 값이 10으로 바뀌고 화살표가 이동한 모습이 보인다.



Chapter 4

주석문과 디버깅

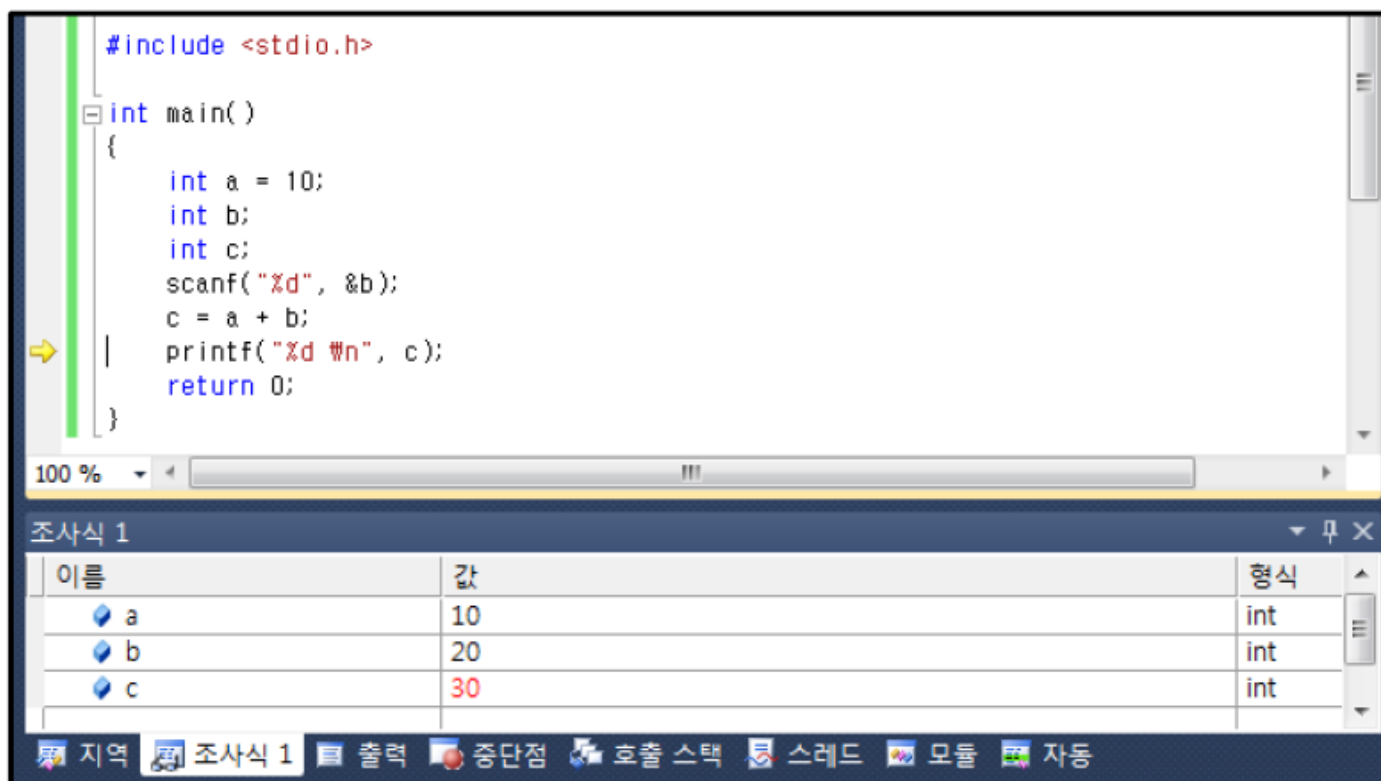
- 1 을 누르면 입력창에서 커서가 깜박이며 입력을 기다린다. 값을 입력하면 b의 값이 입력된 값으로 바뀌는 것을 확인할 수 있다.



Chapter 4

주석문과 디버깅

- 5 다시 **F10**을 누르면 “c = a + b;”를 실행하여 c의 값이 a와 b의 합으로 바뀌는 것을 확인할 수 있다.



- 6 **Ctrl+F10**을 누르면 커서 이전까지의 행이 실행되고 커서가 있는 위치로 화살표가 이동한다.
- 7 **Shift+F5**를 누르면 디버깅 모드가 끝난다.

Thank You!!!

