



쉽게 풀어쓴 C언어 Express

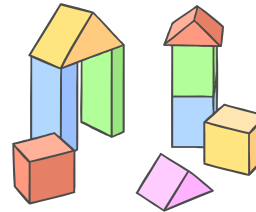
제3장 C프로그램 구성요소





이번 장에서 학습할 내용

- * 주석
- * 변수, 상수
- * 함수
- * 문장
- * 출력 함수 printf()
- * 입력 함수 scanf()
- * 산술 연산
- * 대입 연산



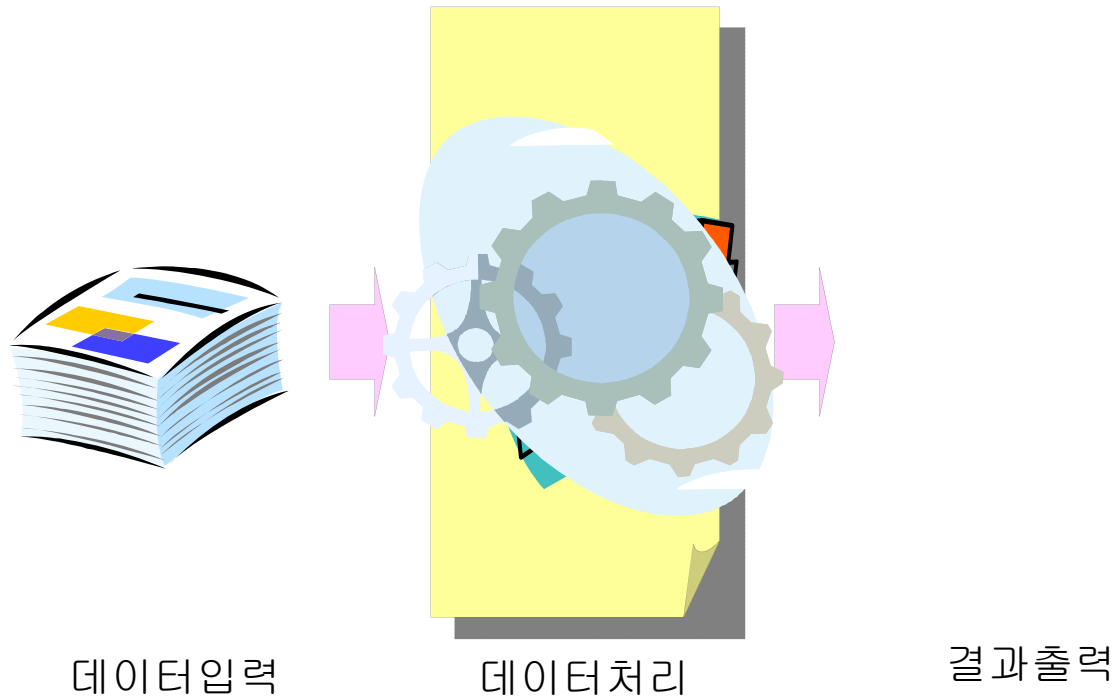
이번 장에서는
C 프로그램을
이루는
구성요소들을
살펴봅니다.





일반적인 프로그램의 형태

- 데이터를 받아서(입력단계), 데이터를 처리한 후에(처리단계), 결과를 화면에 출력(출력단계)한다.





덧셈 프로그램 #1

주석

```
/* 두개의 숫자의 합을 계산하는 프로그램 */
```

전처리기
지시어

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

함수

```
int main(void)
```

```
{
```

```
int x;           // 첫번째 정수를 저장할 변수  
int y;           // 두번째 정수를 저장할 변수  
int sum;         // 두 정수의 합을 저장하는 변수
```

변수선언

```
x = 100;  
y = 200;
```

문장

```
sum = x + y;  
printf("두수의합: %d", sum);
```

연산

```
return 0;
```

```
}
```



프로그램의 실행 화면





주석(comment)

```
/* 두개의 숫자의 합을 계산하는 프로그램 */
```

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

```
int main(void)
```

```
{
```

```
...
```

```
...
```

```
...
```

```
}
```

주석은 코드를
설명하는 글입니다.



주석



3가지 방법의 주석

- `/* 한줄로 된 주석 */`
- `/* 여러
줄로
된 주석 */`
- `// 여기서부터 줄의 끝까지 주석`



들여쓰기

- 들여쓰기(indentation): 같은 수준에 있는 문장들을 왼쪽 끝에서 몇 자 안으로 들여쓰는 것

```
#include <stdio.h>

int main(void)
{
    int x;
    int y;
    int sum;
    ...
    return 0;
}
```

빈줄을 넣어서 의미별로 구별을 한다.

같은 내용의 처리이면 들여쓰기를 한다.

// 첫번째 정수를 저장할 변수
// 두번째 정수를 저장할 변수
// 두 정수의 합을 저장하는 변수

프로그램의 의도를 주석으로 설명한다.



주석과 들여 쓰기가 없다면..

```
#include <stdio.h>
int main(void) { int x; int y; int sum;
x = 100; y = 200; sum = x + y;
printf("두수의 합: %d", sum); return 0; }
```

실행은 되지만 무슨
처리를 하고 있는
프로그램인지 알기
가 힘들고 또한 들
여쓰기가 안 되어
있어서 같은 수준에
있는 문장들을 구분
하기 힘듭니다.





중간 점검

- 주석은 /* /* */ */와 같이 중첩할 수 있을까?
- 주석은 한줄 이상이 될 수 있는가?
- 주석에는 어떤 내용을 쓰면 좋은가?
- 주석은 프로그램의 동작에 어떤 영향을 끼치는가?





전처리기

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

- **#**기호로 시작
- 헤더 파일 **stdio.h**를 소스 코드 안에 포함
- **stdio.h**는 표준 입출력에 대한 라이브러리 함수의 정의가 들어 있다.



전처리기

```
/* 첫번째 프로그램 */
```

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

```
int main(void)
{
    printf("Hello World!");
    return 0;
}
```

hello.c

```
// stdio.h
...
int printf(char *,...);
...
```

stdio.h



중간 점검

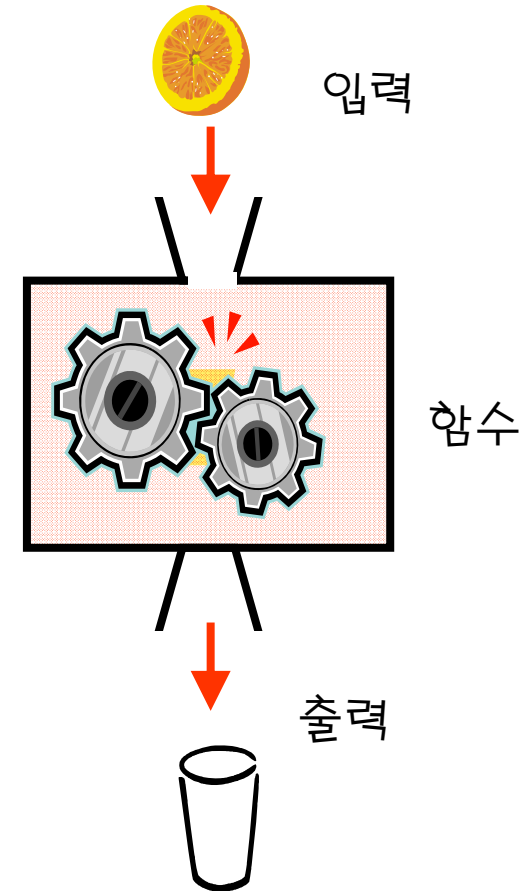
- `printf()`를 사용하기 위하여 포함시켜야 하는 헤더 파일은 무엇인가?
- 전처리기 `#include`의 의미는 무엇인가?





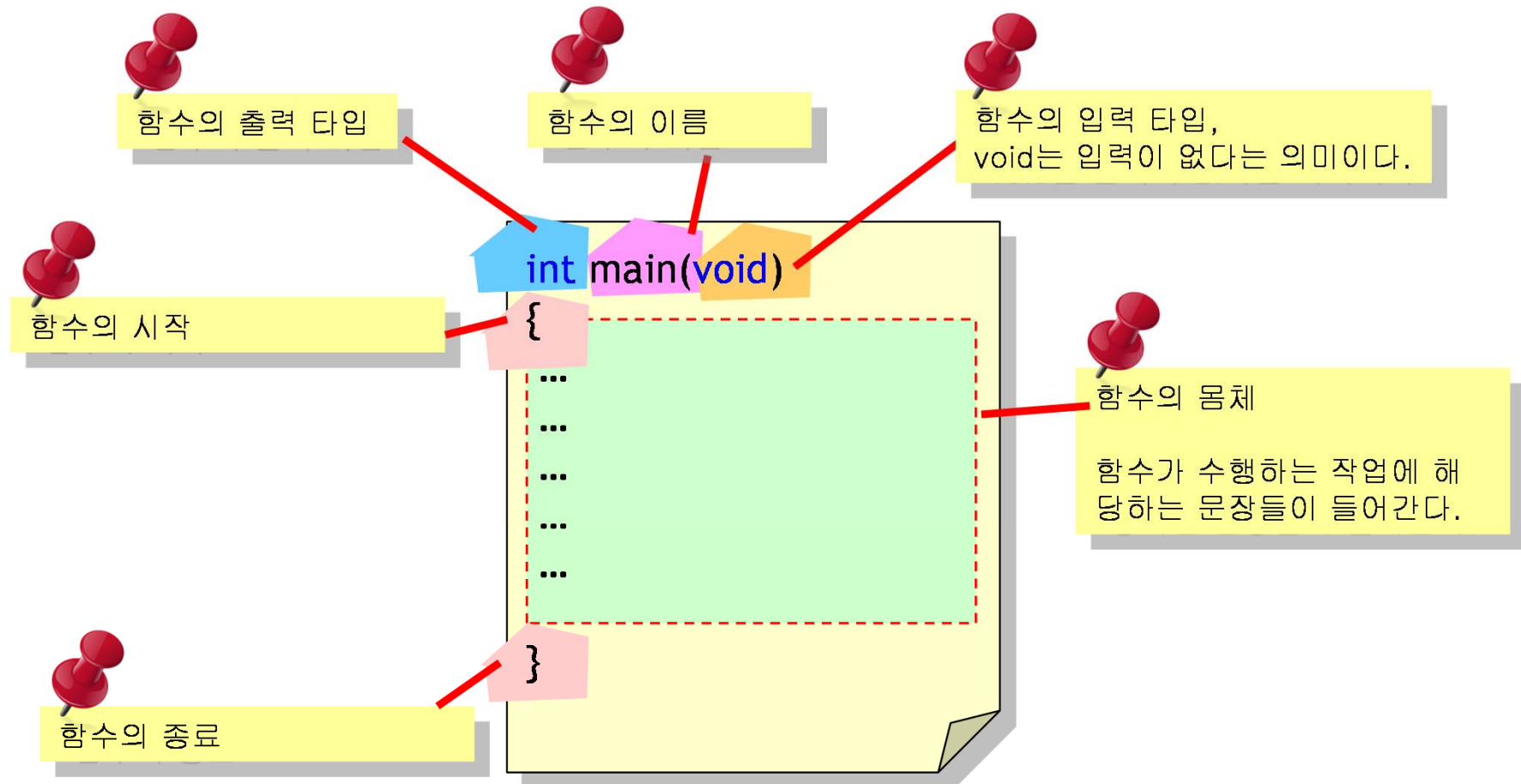
함수

- **함수(function):** 특정 기능을 수행하는 처리 단계들을 괄호로 묶어서 이름을 붙인 것
- 함수는 프로그램을 구성하는 기본적인 단위(부품)





함수의 구조

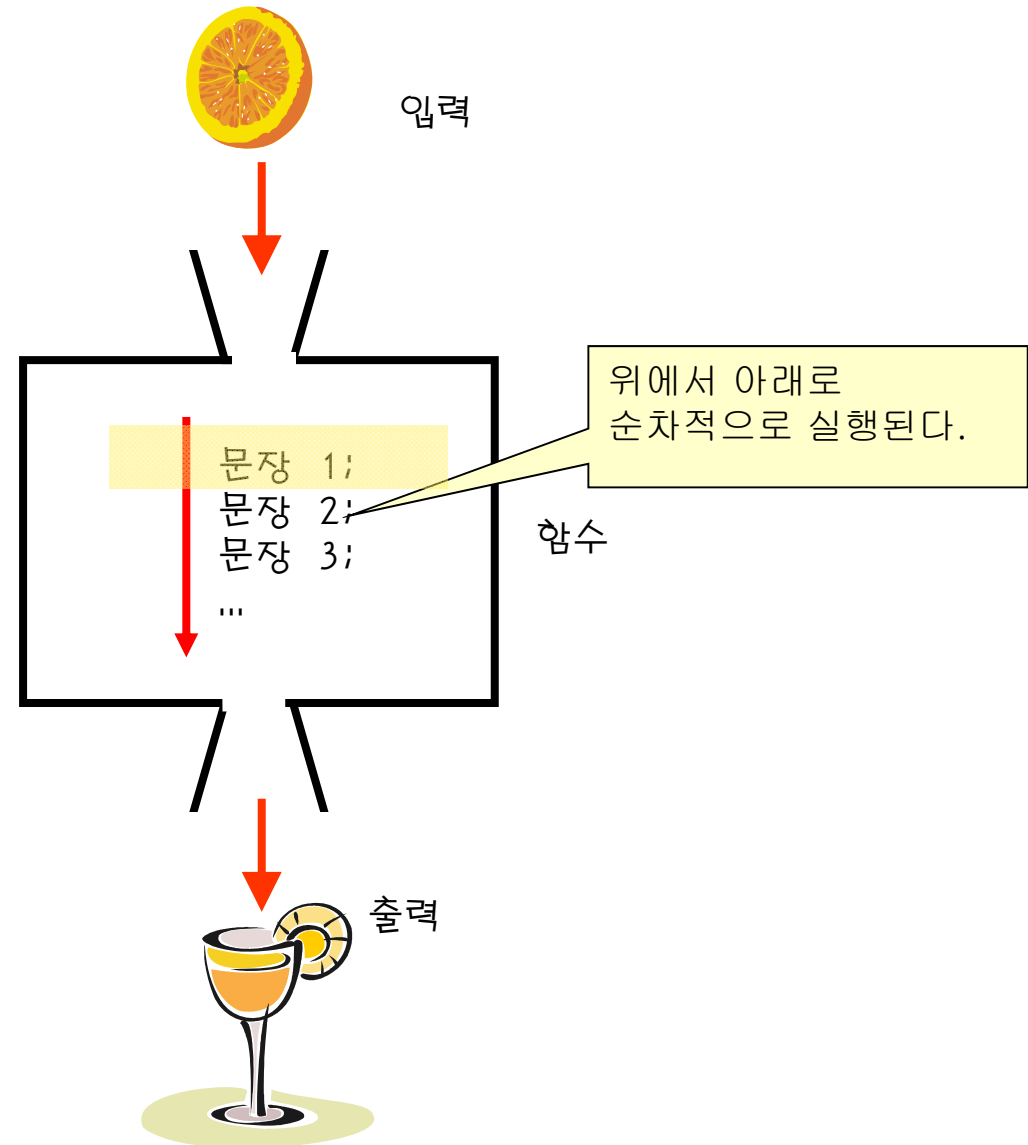




함수안에 들어 있는 것

Q) 그렇다면 함수 안에 들어 있는 것은 무엇인가?

A) 함수 안에는 함수가 처리하는 처리 단계(문장)들이 중괄호 안에 나열



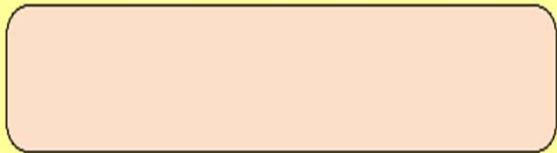


함수

- 작업을 수행하는 문장은 함수 안에 들어가야 함

```
int main(void)
```

```
{
```



작업을 지시하는 부분

```
}
```



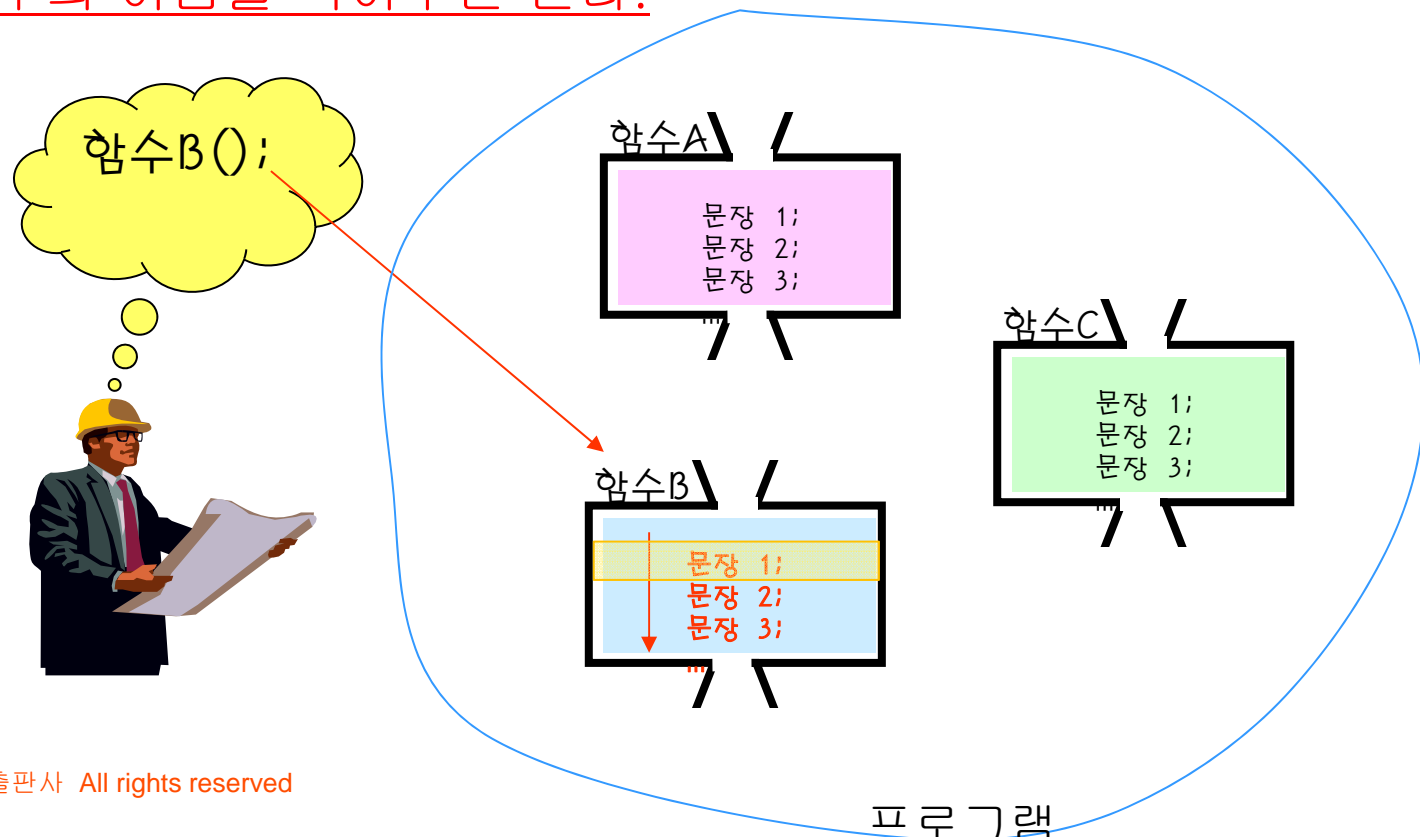
함수 호출

Q) 함수 안에 있는 문장들은 언제 실행되는가?

A) 함수가 호출되면 실행된다.

Q) 함수 호출은 어떻게 하는가?

A) 함수의 이름을 적어주면 된다.

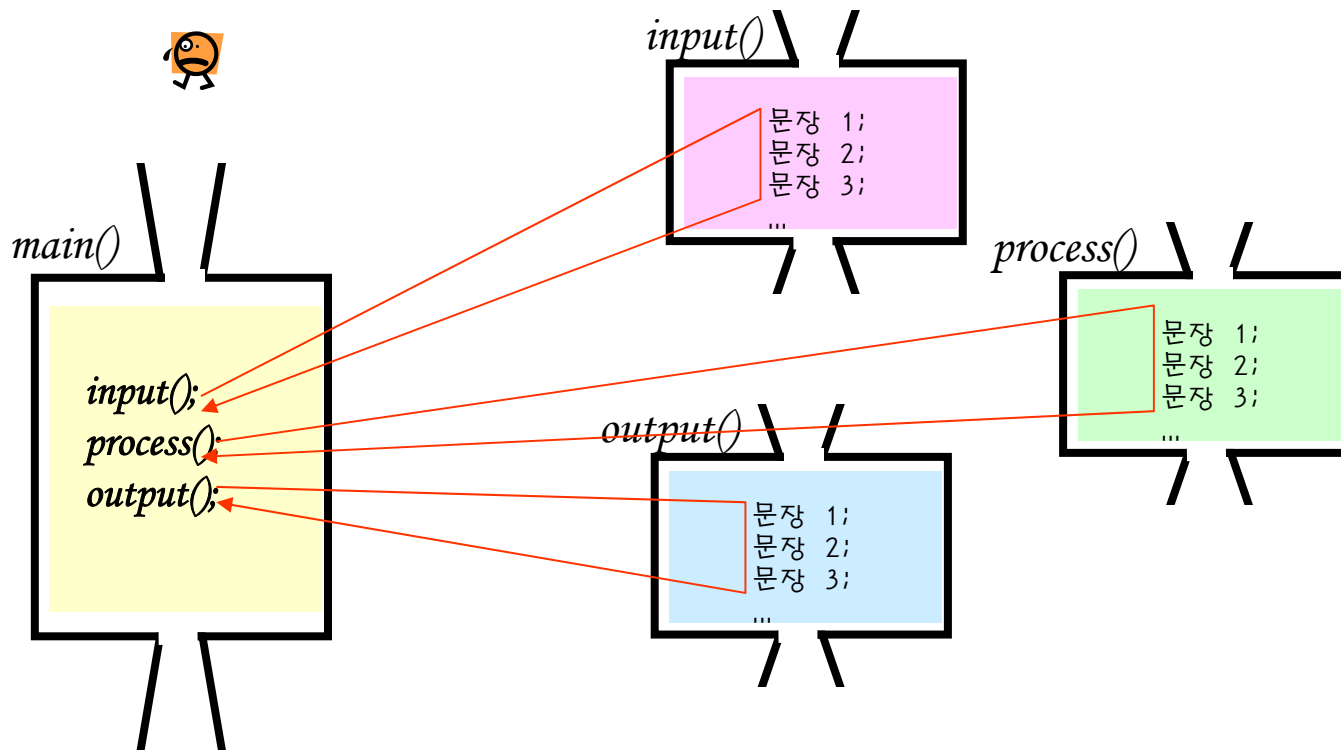




함수 호출

Q) 많은 함수 중에서 가장 먼저 실행되는 것은?

A) **main()** 함수이다. 다른 함수들은 **main()**으로부터 직간접적으로 호출된다.

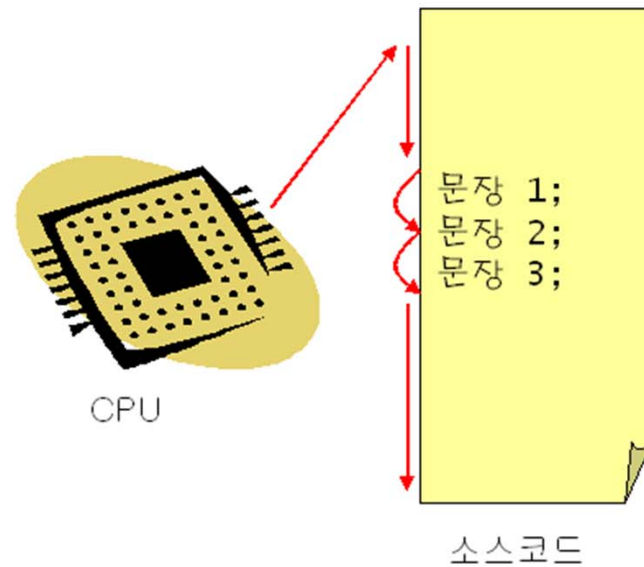




문장

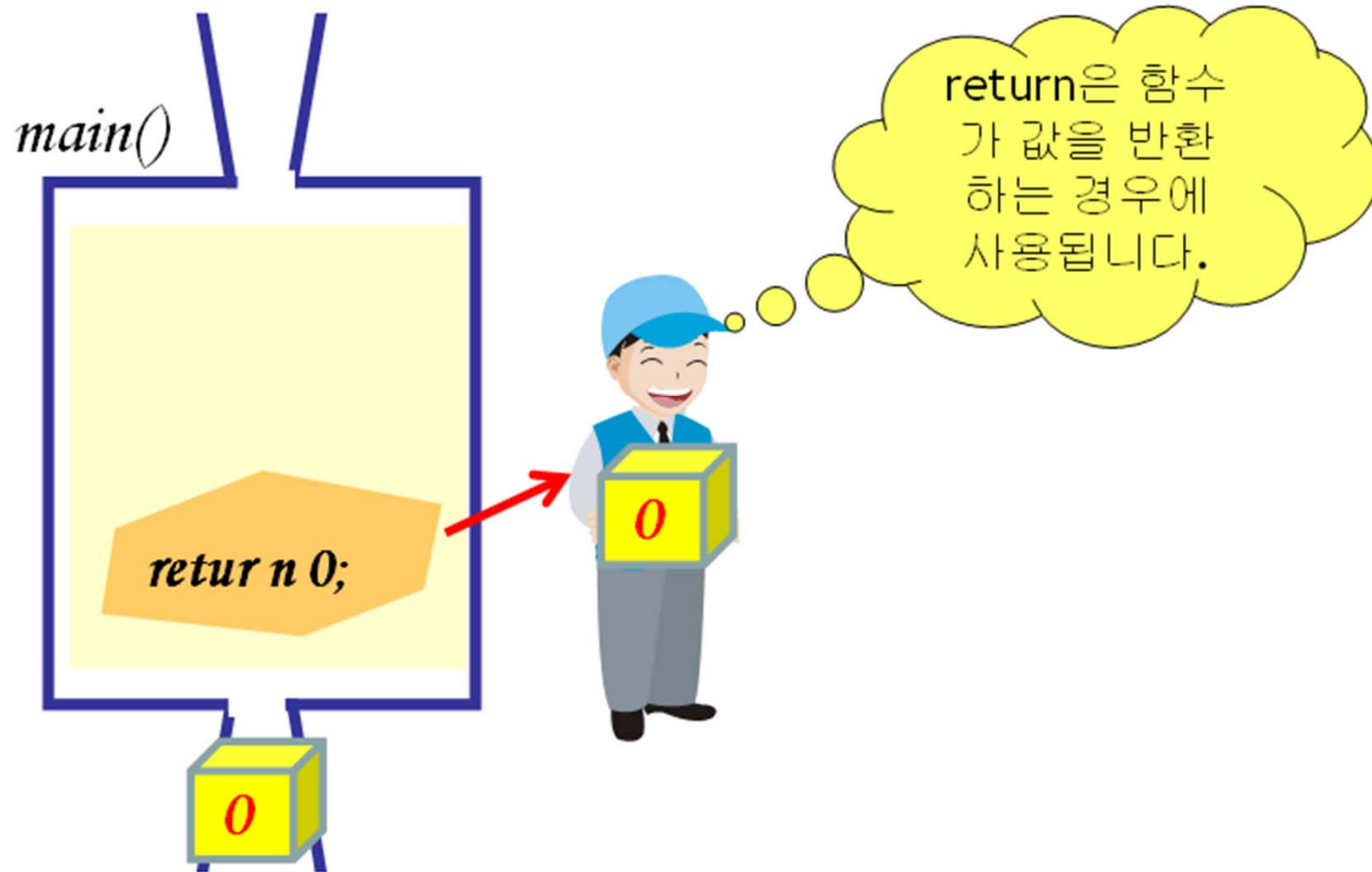
- 함수는 여러 개의 문장으로 이루어진다.
- 문장들은 순차적으로 실행된다.
- 문장은 ;(세미콜론)으로 끝나야 한다.

I



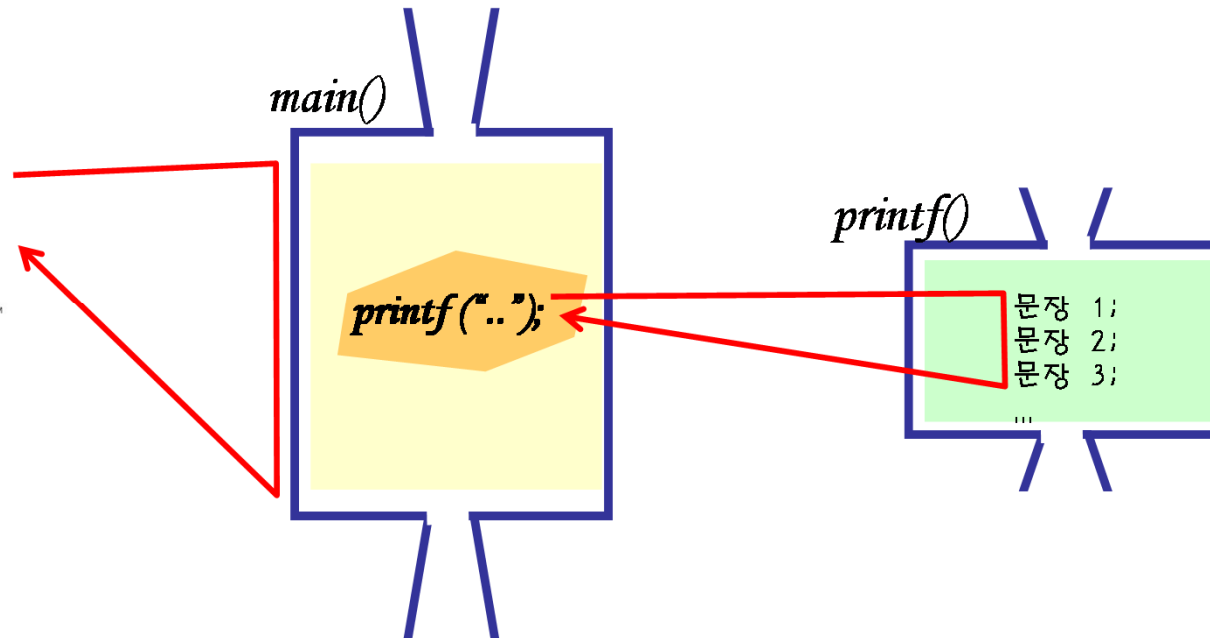


return 문장





main()은 누가 호출할까?



함수를 실행하
려면 함수를
호출하면 됩니
다.





중간 점검

- 모든 **C** 프로그램에 반드시 있어야 되는 함수는 무엇인가?
- 함수의 시작과 끝을 나타내는 기호는 무엇인가?
- 모든 문장은 어떤 기호로 끝나는가?



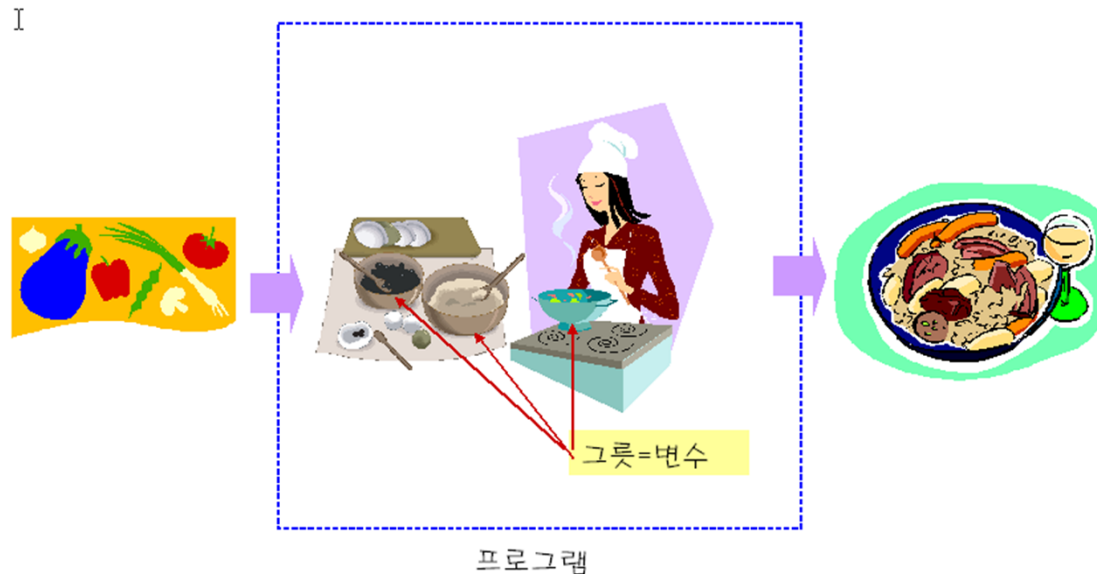


변수

```
int x;    // 첫번째 정수를 저장하는 변수  
int y;    // 두번째 정수를 저장하는 변수  
int sum;  // 두 정수의 합을 저장하는 변수
```

Q) 변수란 무엇인가?

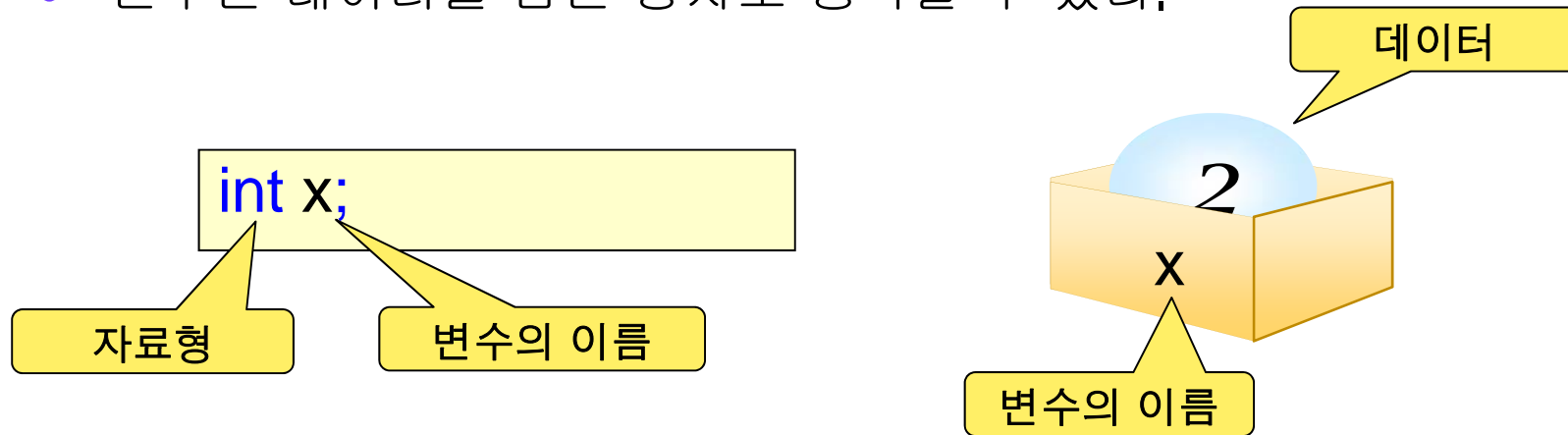
A) 프로그램이 사용하는 데이터를 일시적으로 저장할 목적으로 사용하는 메모리 공간



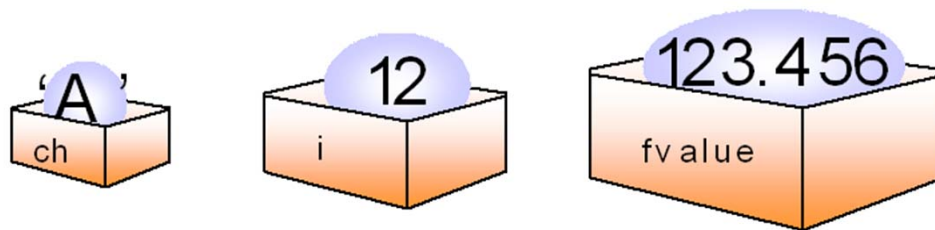


변수의 종류

- 변수는 데이터를 담는 상자로 생각할 수 있다.



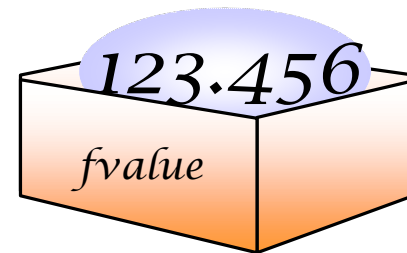
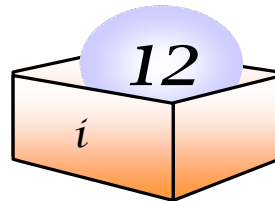
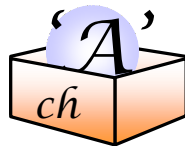
- 변수에는 데이터의 종류에 따라 여러 가지 타입이 존재한다.





변수의 이름

- 식별자(**identifier**): 변수나 함수의 이름
- 식별자를 만드는 규칙
 - 식별자는 영어의 대소문자, 숫자, 밑줄 문자 `_`로 이루어진다.
 - 식별자는 숫자로 시작할 수 없다.
 - 대문자와 소문자를 구별하며 **C** 언어의 키워드와 똑같은 이름은 허용되지 않는다.
- 식별자의 예:
 - `s, s1, student_number`: 올바른 식별자
 - `$s, 2nd_student, int`: 잘못된 식별자





변수 선언

- **변수 선언**: 컴파일러에게 어떤 타입의 변수가 사용되는지를 미리 알리는 것

```
int x;
```

자료형

변수의 이름

지금부터 이
프로그램에서
사용될 변수
들을 소개하겠
습니다.

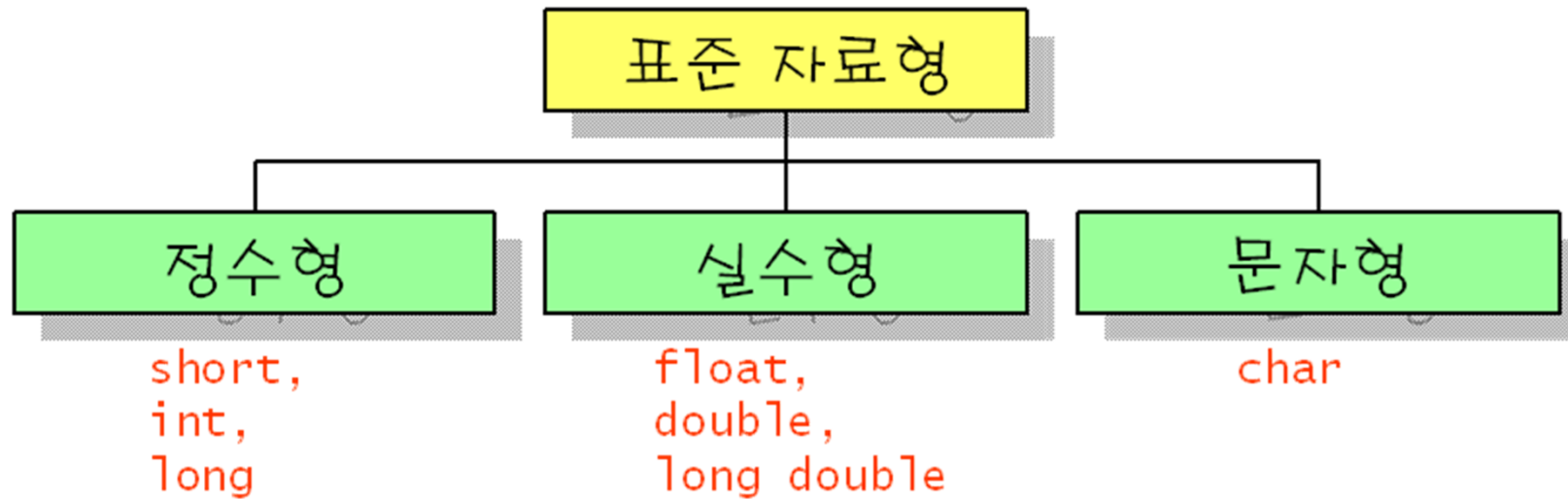
```
int x;  
int y;  
int sum;
```

컴파일러



자료형

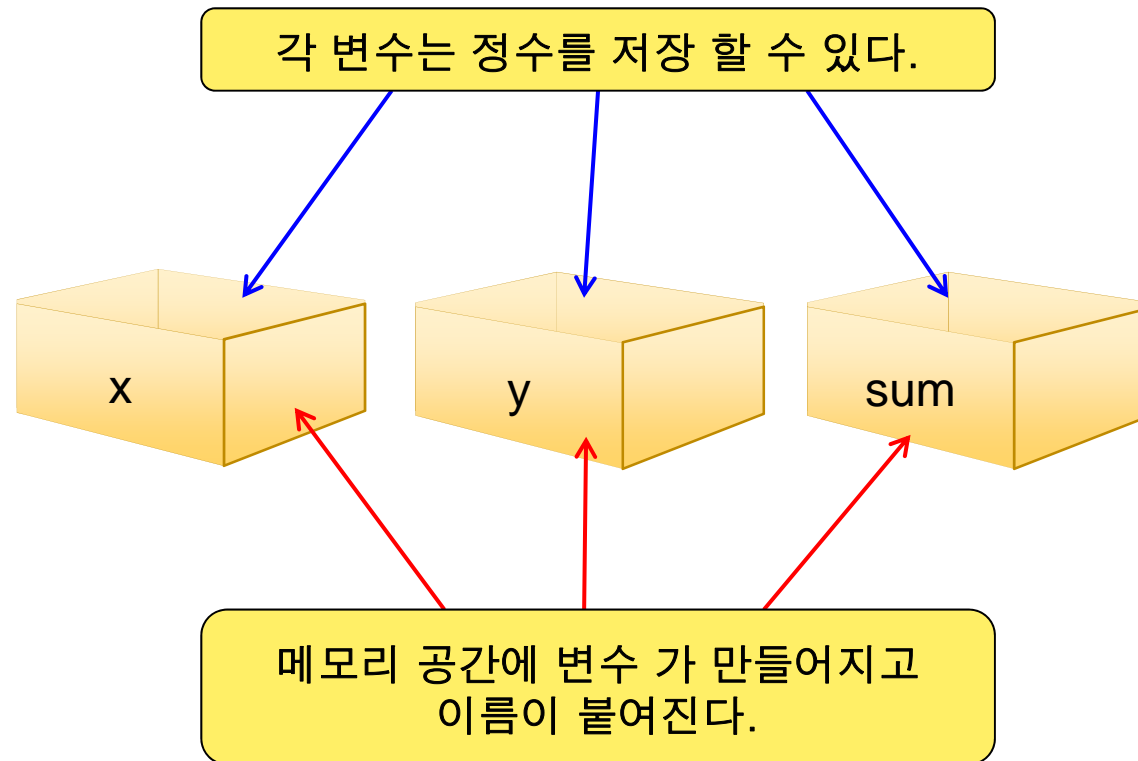
- **자료형(data type):** 변수가 저장할 데이터가 정수인지 실수인지, 아니면 또 다른 어떤 데이터인지를 지정하는 것





변수 선언

```
int x; // 첫번째 정수를 저장하는 변수  
int y; // 두번째 정수를 저장하는 변수  
int sum; // 두 정수의 합을 저장하는 변수
```



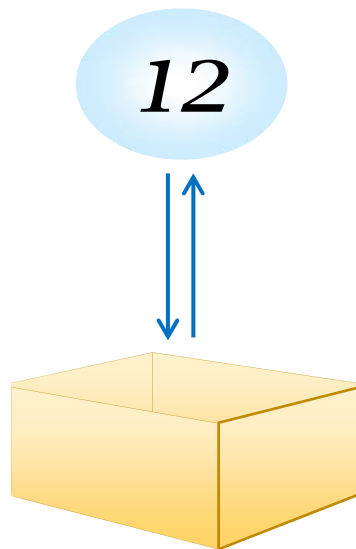


상수

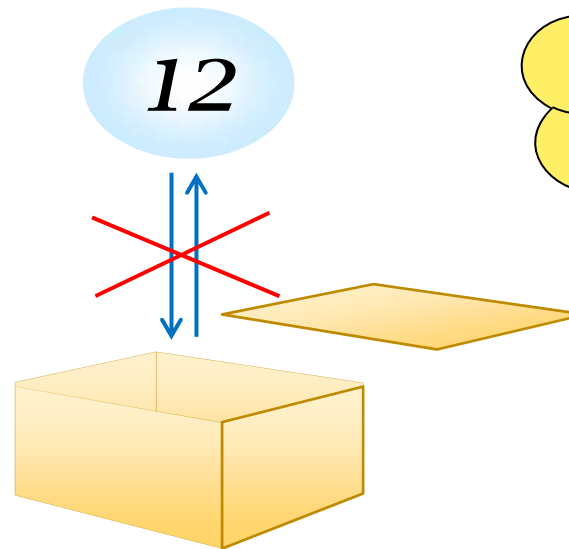
```
x = 100;  
y = 200;
```

→ 상수

- 상수(**constant**): 그 값이 프로그램이 실행하는 동안 변하지 않는 수



변수



상수

변수는 실행도중에
값을 변경할 수
있으나 상수는 한번
값이 정해지면
변경이 불가능합니다.





중간 점검

- `int`형 변수 `i`를 선언하는 문장을 작성하여 보자.
- `double`형 변수 `f`를 선언하는 문장을 작성하여 보자.
- 변수 선언은 함수의 어떤 위치에서 하여야 하는가?

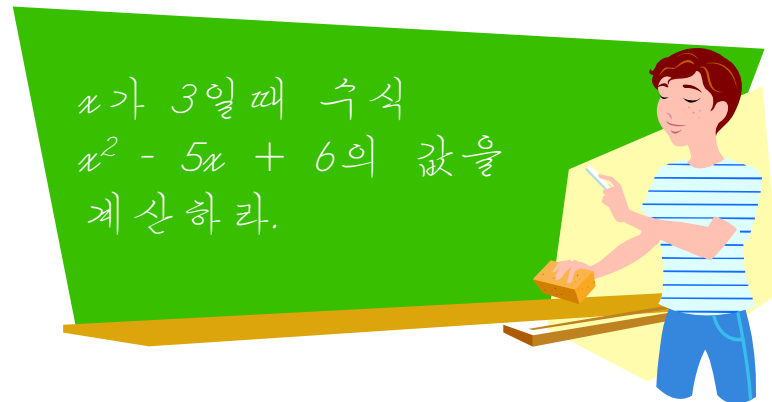




수식

```
sum = x + y;
```

- 수식(**expression**): 피연산자와 연산자로 구성된 식
- 수식은 결과값을 가진다.



```
int x, y;  
  
x = 3;  
y = x * x - 5 * x + 6;  
printf("%d\n", y);
```



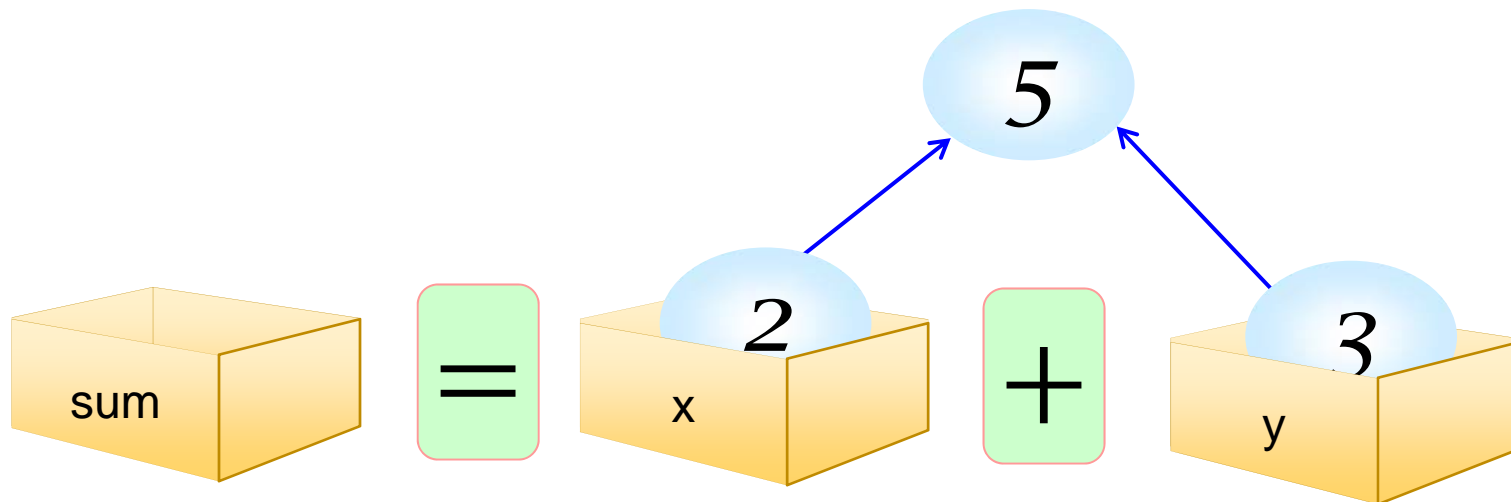

산술 연산

연산	연산자	C 수식	수학에서의 기호
덧셈	+	$x + y$	$x + y$
뺄셈	-	$x - y$	$x - y$
곱셈	*	$x * y$	xy
나눗셈	/	x / y	x / y
나머지	%	$x \% y$	$x \bmod y$



산술 연산

```
sum = x + y;
```

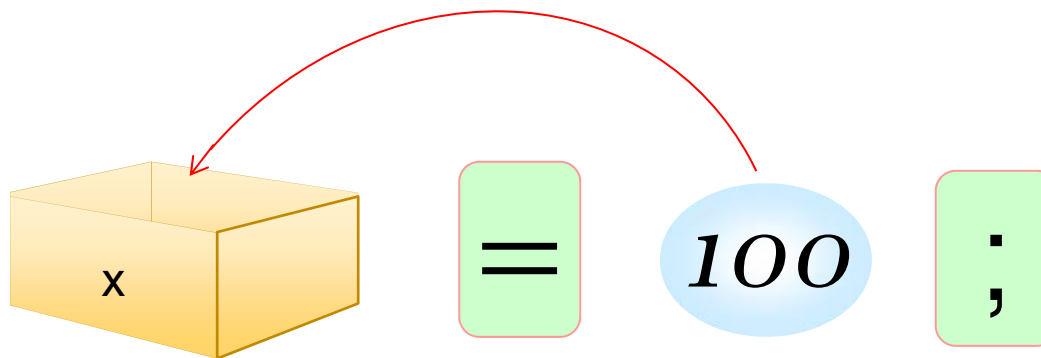




대입 연산

```
x = 100;
```

- 대입 연산(**assignment operation**): 변수에 값을 저장하는 연산
- 대입 연산 = 배정 연산 = 할당 연산



= 연산자는
변수에 값을
저장합니다.





정리

프로그램

컴퓨터 내부

```
#include <stdio.h>
int main(void)
{
```

```
    int x;
    int y;
    int sum;
```

```
    x = 100;
    y = 200;
```

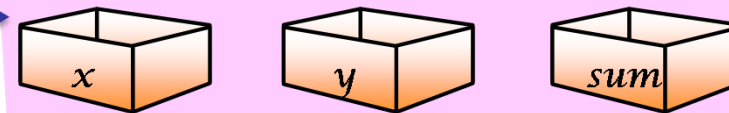
```
    sum = x + y;
```

```
    printf("두수의 합: %d", sum);
```

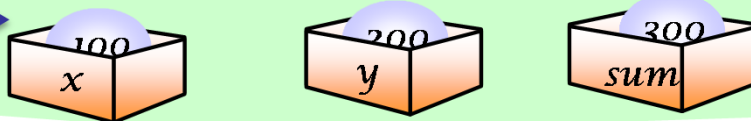
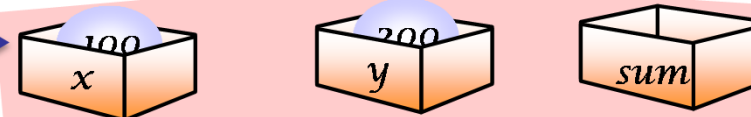
```
    return 0;
```

```
}
```

변수가 생성된다.



변수에 값이 대입된다.



덧셈 연산이 수행된다.



중간 점검

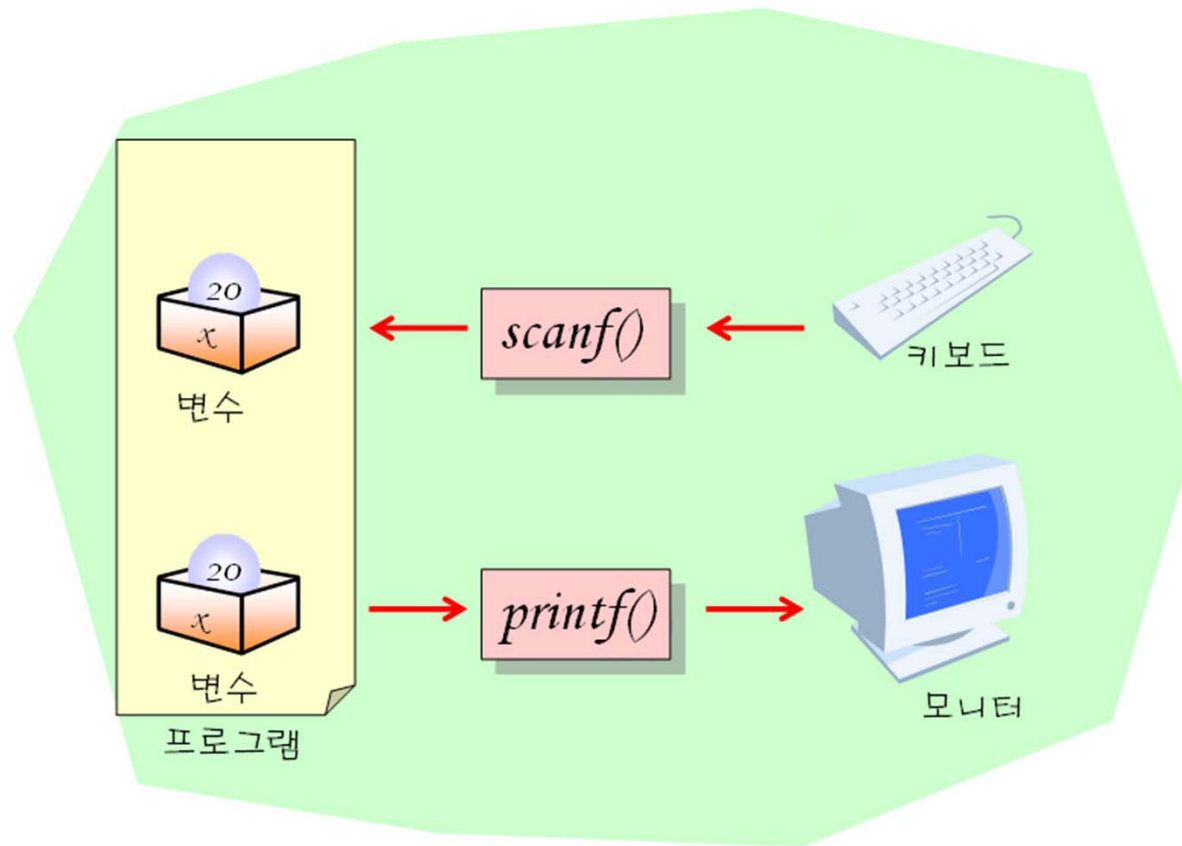
- 변수 **a**와 변수 **b**의 곱을 변수 **product**에 저장하는 문장을 작성하여 보자.
- 변수 **a**를 변수 **b**로 나눈 값을 변수 **quotient**에 저장하는 문장을 작성하여 보자.





printf()

- **printf():** 모니터에 출력을 하기 위한 표준 출력 라이브러리 함수

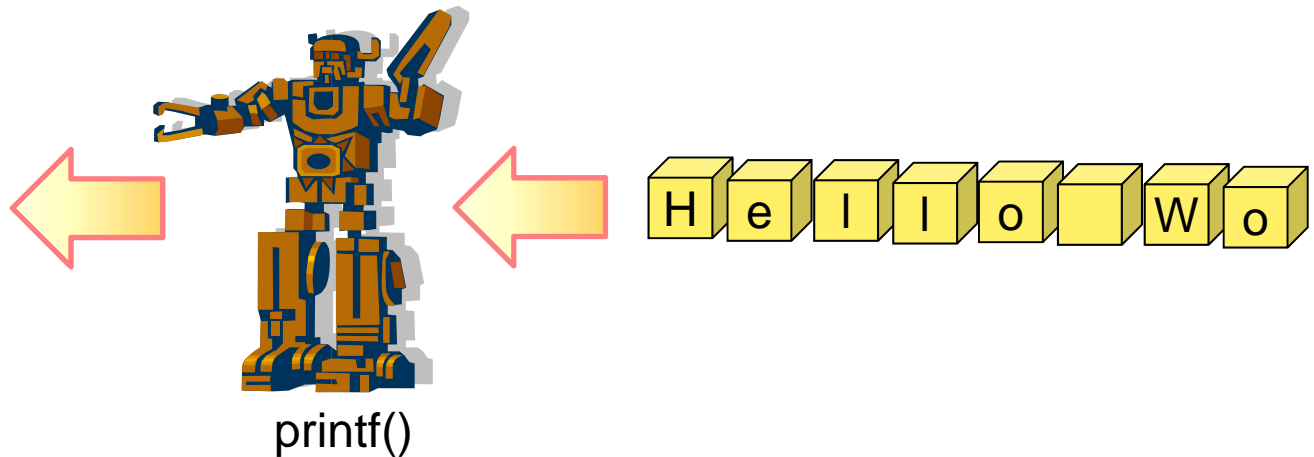




문자열 출력

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

- 문자열(**string**): "Hello World!\n"와 같이 문자들을 여러 개 나열한 것





변수값 출력

```
printf("두수의 합: %d", sum);
```

두수의 합: 30

형식 지정자 의 개수와
변수의 개수와 순서는
같아야 한다.





형식 지정자

- 형식 지정자: **printf()**에서 값을 출력하는 형식을 지정한다.

형식 지정자	의미	예	실행 결과
%d	10진 정수로 출력	<code>printf("%d \n", 10);</code>	10
%f	실수로 출력	<code>printf("%f \n", 3.14);</code>	3.14
%c	문자로 출력	<code>printf("%c \n", 'a');</code>	a
%s	문자열로 출력	<code>printf("%s \n", "Hello");</code>	Hello



여러 개의 변수값 출력

형식 제어 문자열

```
printf( "학번 %d 의 성적은 %f \n", number, height );
```

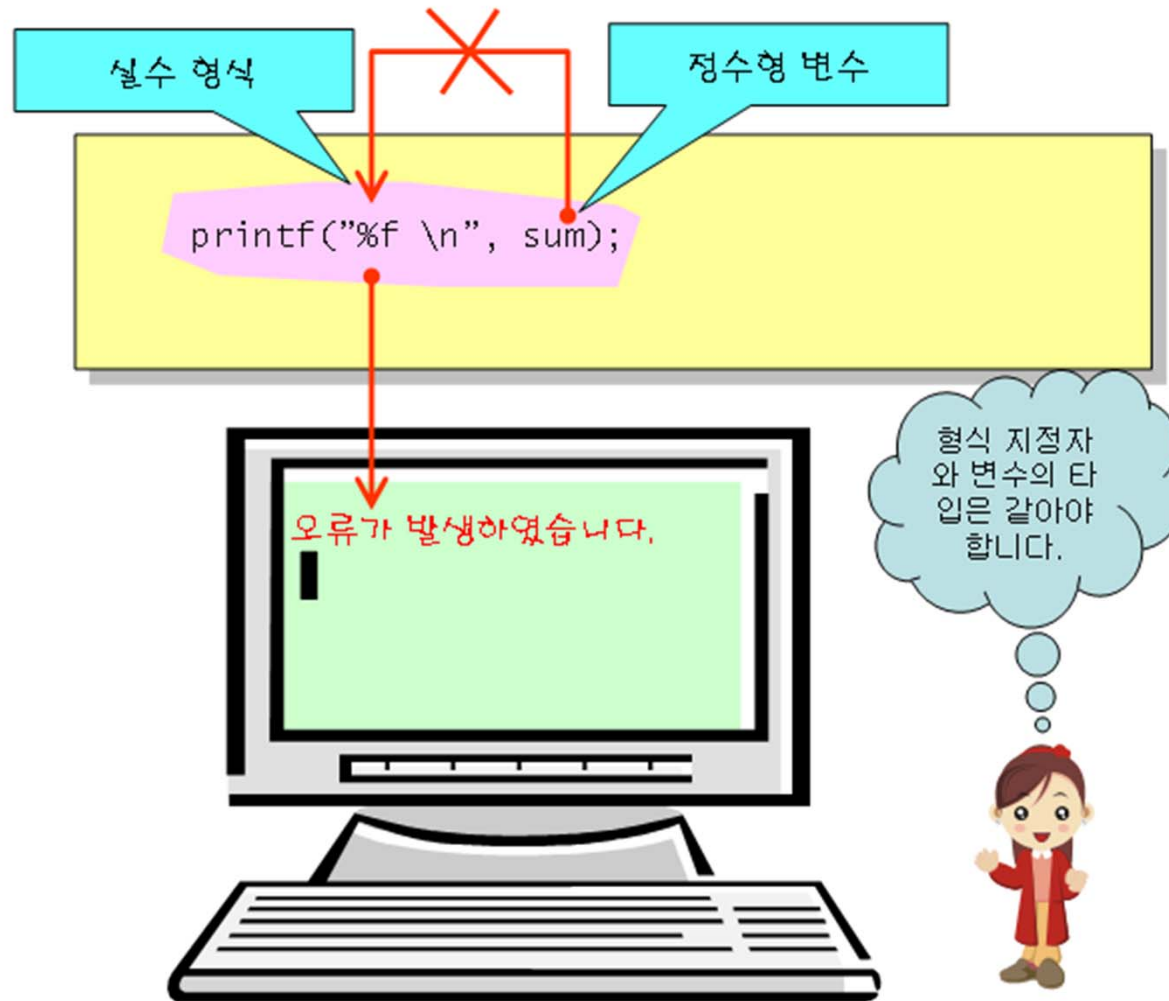
학번 23의 성적은 3.99

형식 지정자 의 개수와
변수의 개수와 순서는
같아야 한다.





주의!





중간 점검

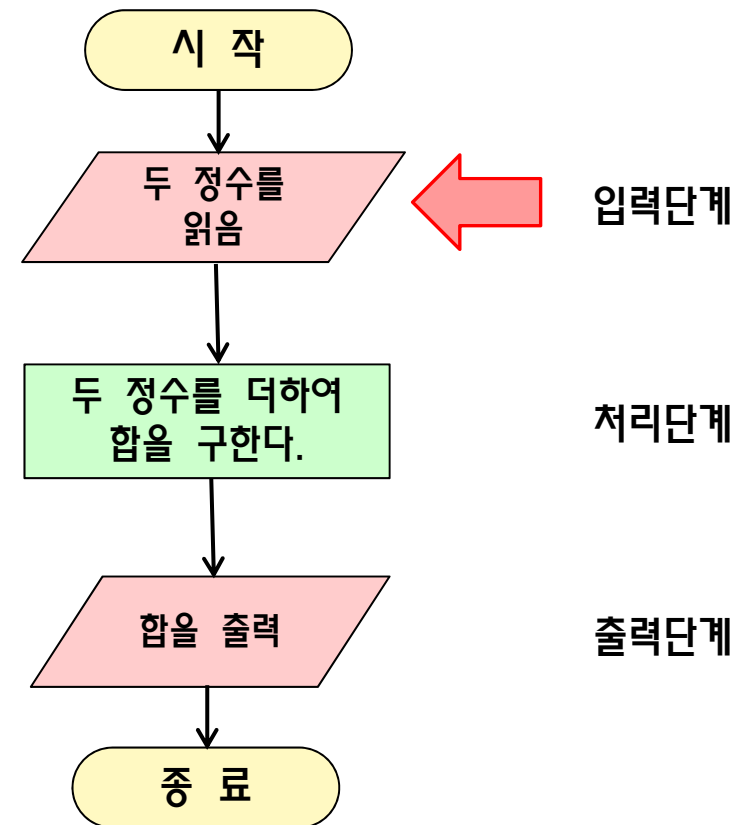
- `printf()`에서 변수의 값을 실수 형태로 출력할 때 사용하는 형식 지정자는 무엇인가?
- `printf()`를 사용하여 정수형 변수 `k`의 값을 출력하는 문장을 작성하여 보자.





덧셈 프로그램 #2

- 사용자로부터 입력을 받아보자.





두번째 덧셈 프로그램

```
// 사용자로부터 입력받은 2개의 정수의 합을 계산하여 출력

#include <stdio.h>

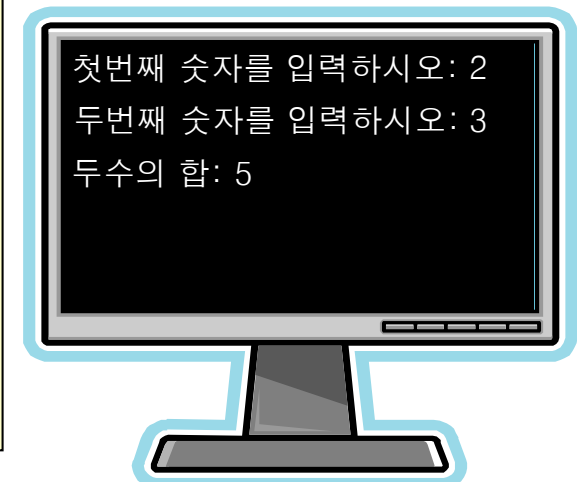
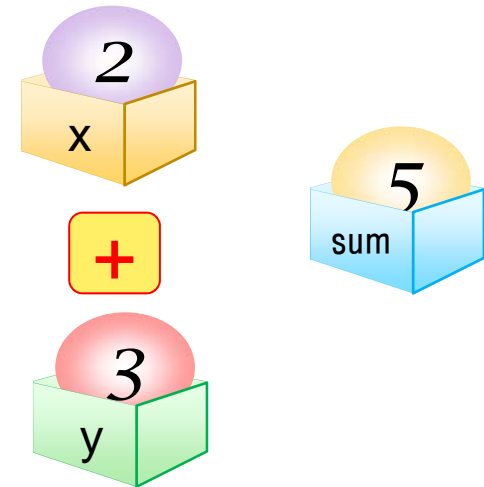
int main(void)
{
    → int x;           // 첫번째 정수를 저장할 변수
      int y;           // 두번째 정수를 저장할 변수
      int sum;         // 2개의 정수의 합을 저장할 변수

    printf("첫번째 숫자를 입력하시오:"); // 입력 안내 메시지 출력
    scanf("%d", &x);                   // 하나의 정수를 받아서 x에 저장

    printf("두번째 숫자를 입력하시오:"); // 입력 안내 메시지 출력
    scanf("%d", &y);                   // 하나의 정수를 받아서 x에 저장

    sum = x + y;                       // 변수 2개를 더한다.
    printf("두수의 합: %d", sum);       // sum의 값을 10진수 형태로 출력

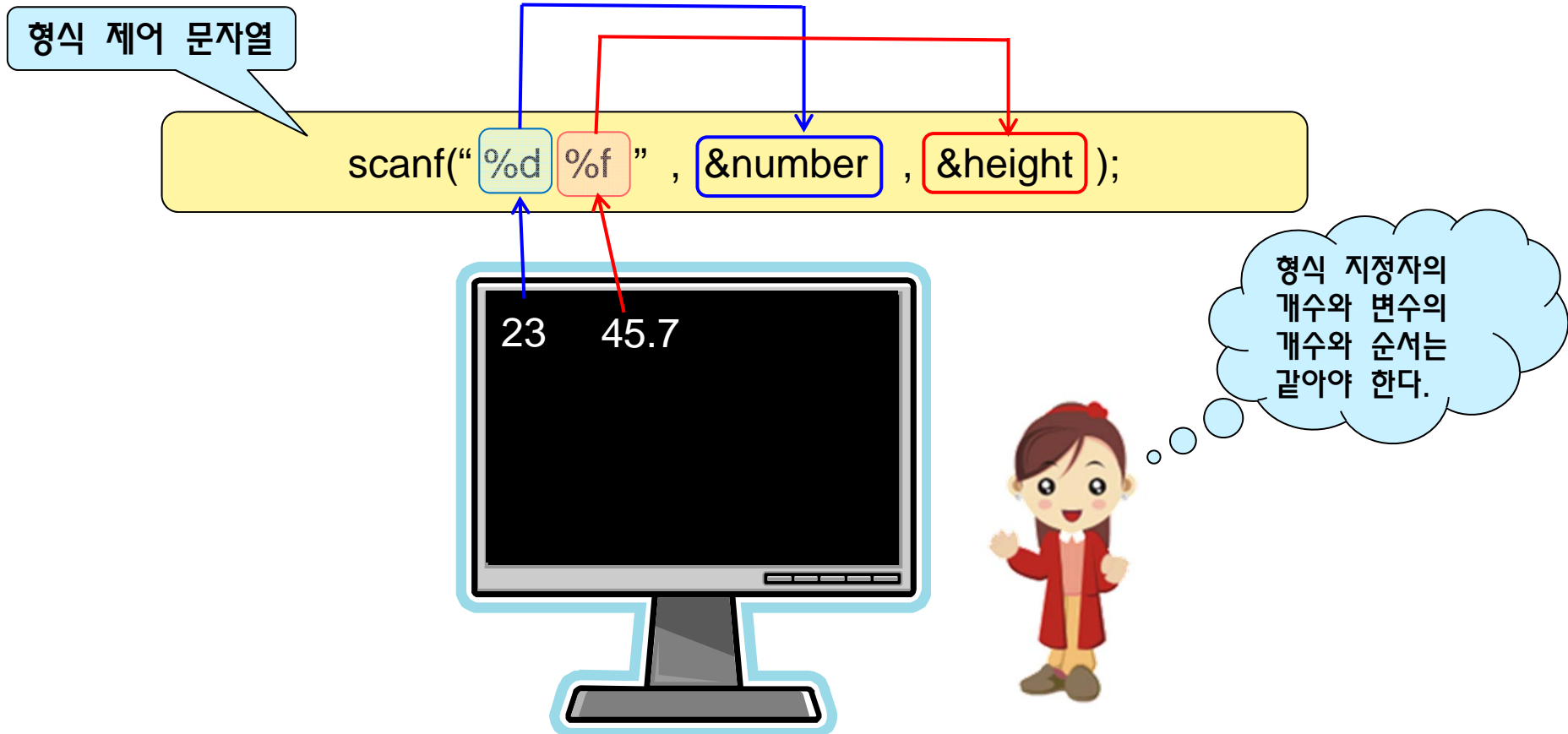
    return 0;                         // 0을 외부로 반환
}
```





scanf()

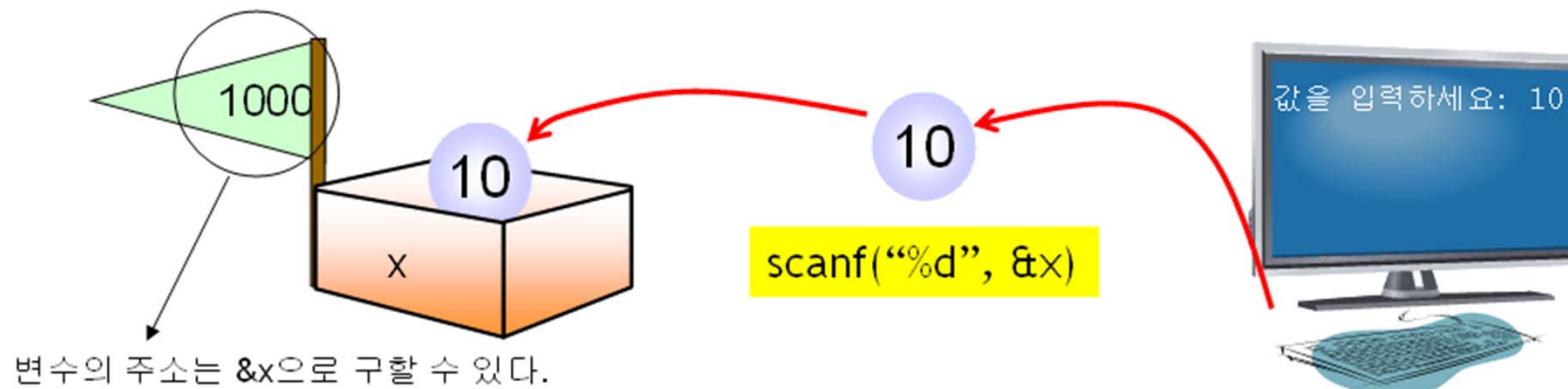
- `scanf()`: 키보드로부터 입력을 하기 위한 라이브러리 함수





scanf()의 동작

- 키보드로부터 값을 받아서 변수에 저장한다.
- 변수의 주소를 필요로 한다.





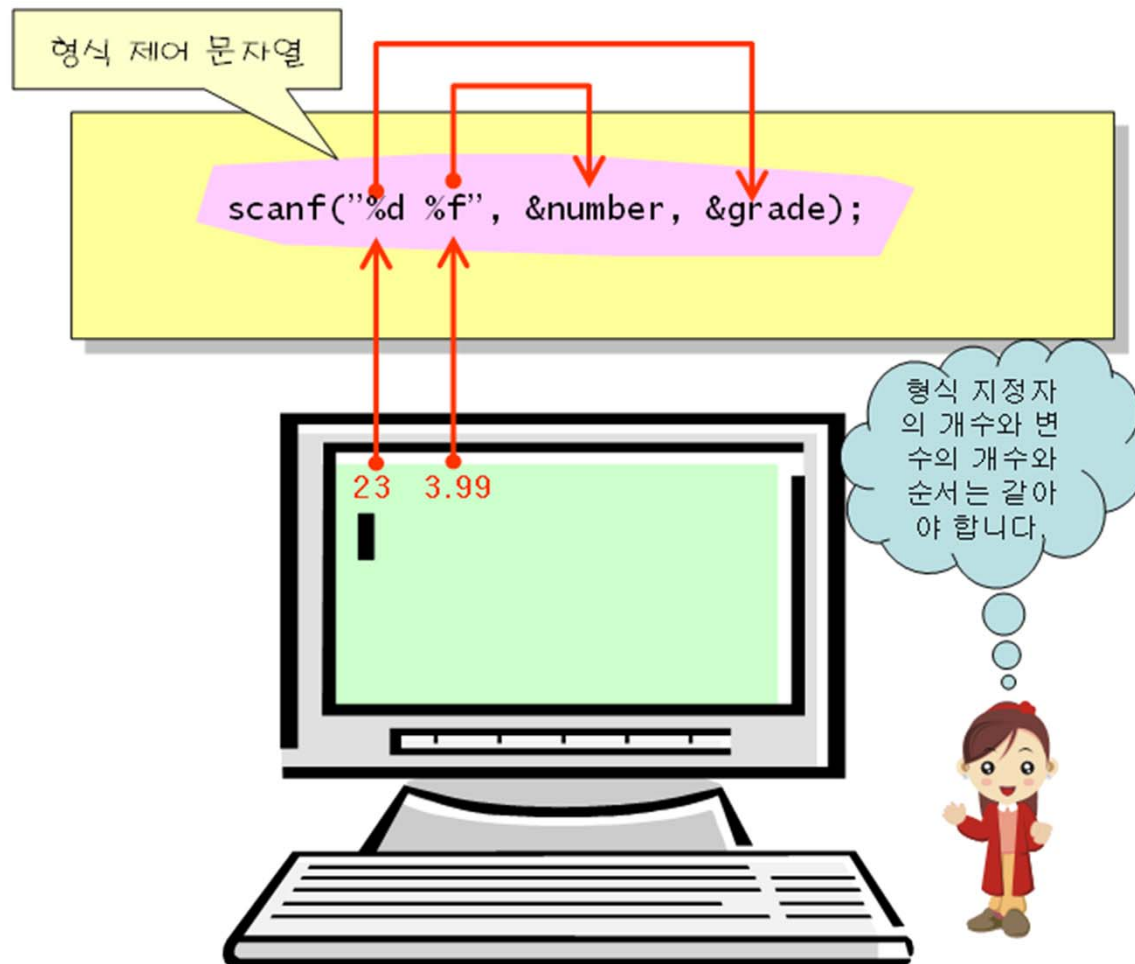
형식 지정자

형식 지정자	의미	예
%d	정수를 10진수로 입력한다	scanf("%d", &i);
%f	float 형의 실수로 입력한다.	scanf("%f", &f);
%lf	double 형의 실수로 입력한다.	scanf("%lf", &d);
%c	문자 형태로 입력한다.	scanf("%c", &ch);
%s	문자열 형태로 입력한다.	char s[10]; scanf("%s", &s);



scanf()

- 형식 지정자와 변수의 자료형은 일치하여야 함





실수 입력

- `float` ratio = 0.0;
scanf("%f", &ratio);
- `double` scale = 0.0;
scanf("%lf", &scale);

주의!!!



중간 점검

- `scanf()`를 사용하여 사용자로 부터 실수값을 받아서 `double`형의 변수 `value`에 저장하는 문장을 작성하여 보자.





연봉 계산 프로그램

```
/* 저축액을 계산하는 프로그램 */
```

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

```
int main(void)
```

```
{
```

```
    → int salary;           // 월급  
    int deposit;           // 저축액
```

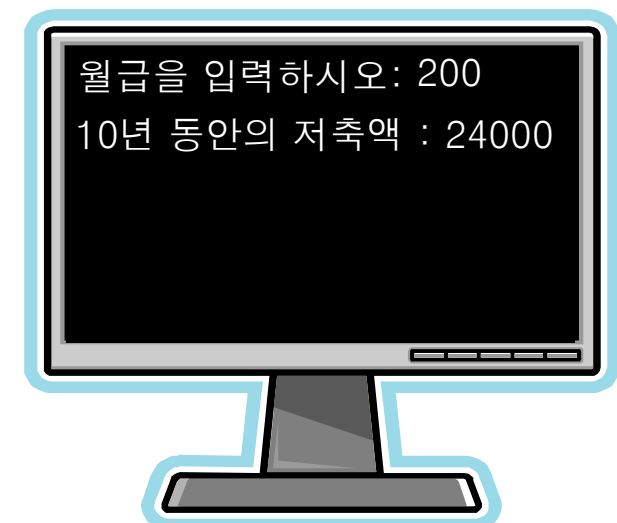
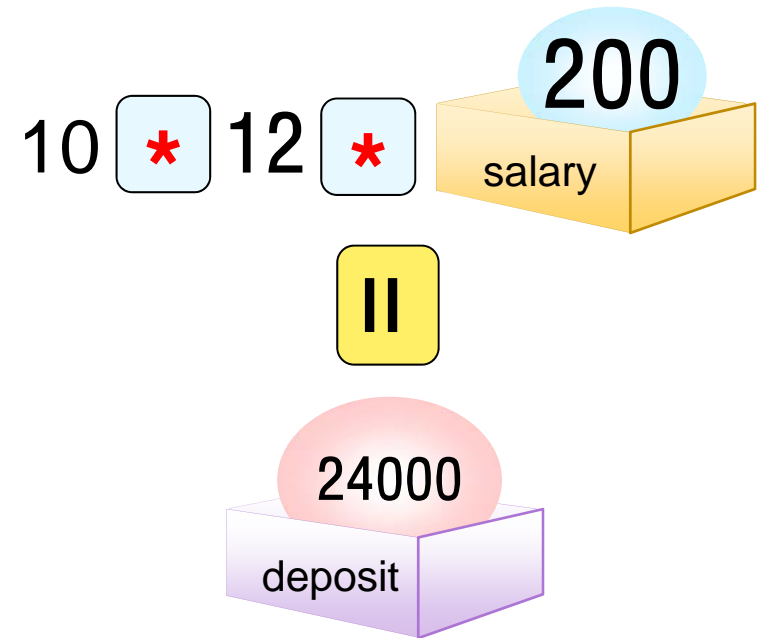
```
    printf("월급을 입력하시오: ");  
    scanf("%d", &salary);
```

```
    deposit = 10 * 12 * salary;
```

```
    printf("10년 동안의 저축액: %d\n", deposit);
```

```
    return 0;
```

```
}
```





원의 면적 프로그램

```
/* 원의 면적을 계산하는 프로그램*/
```

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

```
int main(void)
```

```
{
```

```
    float radius; // 원의 반지름
```

```
    float area;   // 면적
```

```
    printf("반지름을 입력하시오: ");
```

```
    scanf("%f", &radius);
```

```
    area = 3.14 * radius * radius;
```

```
    printf("원의 면적: %f\n", area);
```

```
    return 0;
```

```
}
```

5.0

*

5.0

*

3.14

=

78.500000

radius

area

반지름을 입력하시오: 5.0
원의 면적: 78.500000



환율 계산 프로그램

```
/* 환율을 계산하는 프로그램*/
```

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

```
int main(void)
```

```
{
```

```
    float rate; // 원/달러 환율  
    float usd;  // 달러화  
    int krw;    // 원화
```

```
    printf("달러에 대한 원화 환율을 입력하시오: ");  
    scanf("%f", &rate);
```

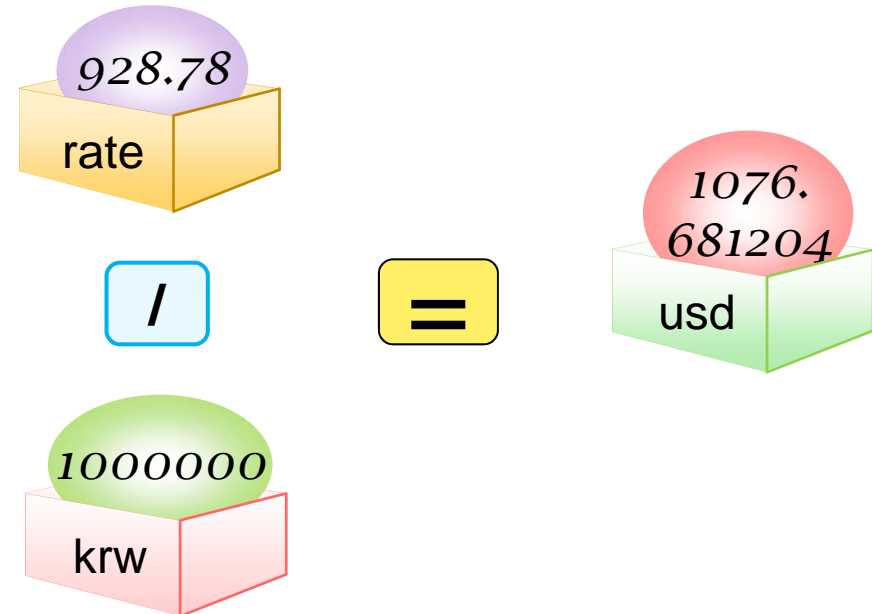
```
    printf("원화 금액을 입력하시오: ");  
    scanf("%d", &krw);
```

```
    usd = krw / rate;
```

```
    printf("원화 %d원은 %f달러입니다.\n", krw, usd);
```

```
    return 0;
```

```
}
```



```
달러에 대한 원화 환율을 입력하시오: 928.78  
원화 금액을 입력하시오: 1000000  
원화 1000000원은 1076.681204달러입니다.
```

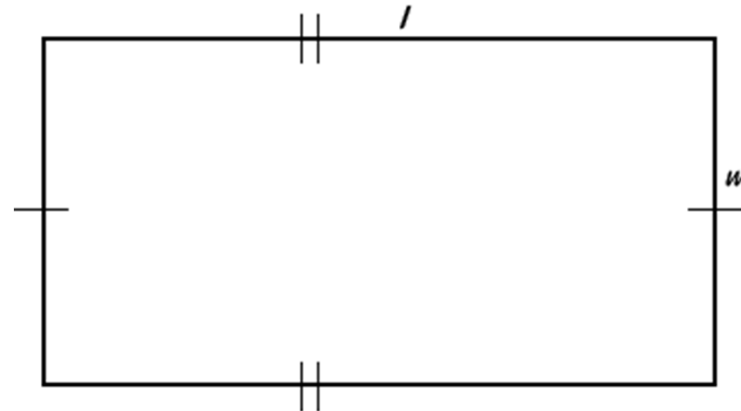


실습: 사각형의 둘레와 면적

- 필요한 변수는 w , h , $area$, $perimeter$ 라고 하자.
- 변수의 자료형은 실수를 저장할 수 있는 **double**형으로 하자.
- $area = w * h$;
- $perimeter = 2 * (w + h)$;

Rectangle

Perimeter $p = 2(l + w)$
or
 $p = 2l + 2w$



Area $A = lw$



프로그램의 실행 화면





코딩

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

```
int main(void)
```

```
{
```

```
    double w;
```

```
    double h;
```

```
    double area;
```

```
    double perimeter;
```

```
    w = 10.0;
```

```
    h = 5.0;
```

```
    area = w*h;
```

```
    perimeter = 2*(w+h);
```

```
    printf("사각형의 넓이: %lf", area);
```

```
    printf("사각형의 둘레: %lf", perimeter);
```

```
    return 0;
```

```
}
```





도전문제

1. 한번의 `printf()` 호출로 변수 `perimeter`와 `area`의 값이 동시에 출력 되도록 변경하라.
2. 변수들을 한 줄에 모두 선언하여 보자.
3. `w`와 `h`의 값을 사용자로부터 받도록 변경하여 보자. `%lf`를 사용한다.





Q & A

