

개정3판

Visual
Studio
2017

쉽게 풀어쓴

C언어
EXPRESS



천인국 지음

제5장 수식과 연산자



이번 장에서 학습할 내용

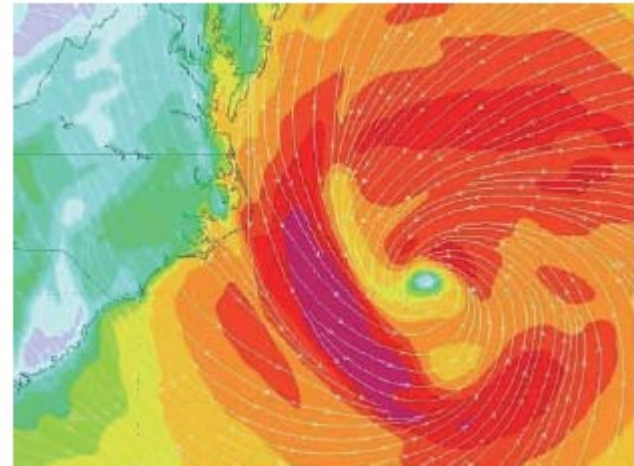


- * 수식과 연산자란?
- * 대입 연산
- * 산술 연산
- * 논리 연산
- * 관계 연산
- * 조건 연산
- * 비트 연산
- * 우선 순위와 결합 법칙





컴퓨터는 근본적으로 계산하는 기계





수식의 예

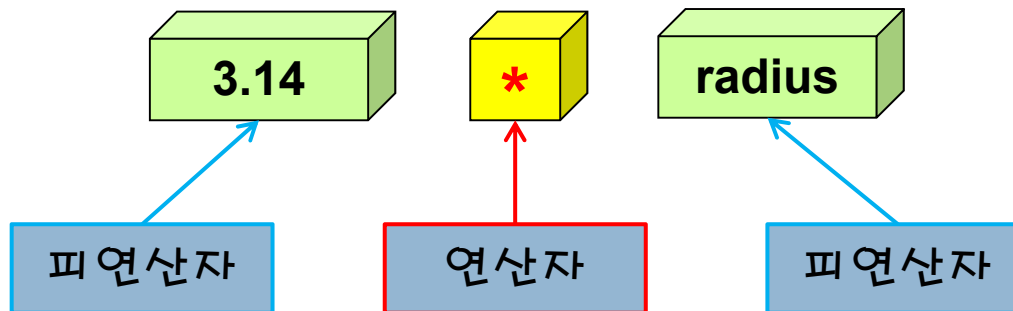


```
int x, y;  
  
x = 3;  
y = x*x - 5*x + 6;  
printf("%d\n", y);
```



수식

- 수식(expression)
 - 상수, 변수, 연산자의 조합
 - 연산자와 피연산자로 나누어진다.





기능에 따른 연산자의 분류

연산자의 분류	연산자	의미
대입	=	오른쪽을 왼쪽에 대입
산술	+ - * / %	사칙연산과 나머지 연산
부호	+ -	
증감	++ --	증가, 감소 연산
관계	> < == != >= <=	오른쪽과 왼쪽을 비교
논리	&& !	논리적인 AND, OR, NOT
조건	?	조건에 따라 선택
coma	,	피연산자들을 순차적으로 실행
비트 단위 연산자	& ^ ~ << >>	비트별 AND, OR, XOR, 반전, 이동
sizeof 연산자	sizeof	자료형이나 변수의 크기를 바이트 단위로 반환
형변환	(type)	변수나 상수의 자료형을 변환
포인터 연산자	* & []	주소계산, 포인터가 가리키는 곳의 내용 추출
구조체 연산자	. ->	구조체의 멤버 참조



피연산자수에 따른 연산자 분류

- 단항 연산자: 피연산자의 수가 1개

```
++x;  
--y;
```

- 이항 연산자: 피연산자의 수가 2개

```
x + y  
x - y
```

- 삼항 연산자: 연산자의 수가 3개

```
x ? y : z
```



중간 점검

1. 수식(expression)이란 어떻게 정의되는가?
2. 상수 10도 수식이라고 할 수 있는가?
3. 아래의 수식에서 피연산자와 연산자를 구분하여 보라.
 $y = 10 + 20;$
4. 연산자를 단항 연산자, 이항 연산자, 삼항 연산자로 나누는 기준은 무엇인가?





산술 연산자

- 산술 연산: 컴퓨터의 가장 기본적인 연산
- 덧셈, 뺄셈, 곱셈, 나눗셈, 나머지 연산을 수행하는 연산자

연산자	기호	사용예	결과값
덧셈	+	$7 + 4$	11
뺄셈	-	$7 - 4$	3
곱셈	*	$7 * 4$	28
나눗셈	/	$7 / 4$	1
나머지	%	$7 \% 4$	3



산술 연산자의 예

$$y = mx + b$$

$$y = m * x + b$$

$$y = ax^2 + bx + c$$

$$y = a * x * x + b * x + c$$

$$m = \frac{x + y + x}{3}$$

$$m = (x + y + z) / 3$$



(참고) 거듭 제곱 연산자는?

C에는 거듭 제곱을 나타내는 연산자는 없다.
 $x * x$ 와 같이 단순히 변수를 두 번 곱한다.



정수 사칙 연산

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

```
int main(void)
```

```
{
```

```
    int x, y, result;
```

```
    printf("두개의 정수를 입력하시오: ");
```

```
    scanf("%d %d", &x, &y);
```

```
    result = x + y;
```

```
    printf("%d + %d = %d", x, y, result);
```

```
    result = x - y;           // 뺄셈
```

```
    printf("%d - %d = %d", x, y, result);
```

```
    result = x * y;          // 곱셈
```

```
    printf("%d * %d = %d", x, y, result);
```

```
    result = x / y;          // 나눗셈
```

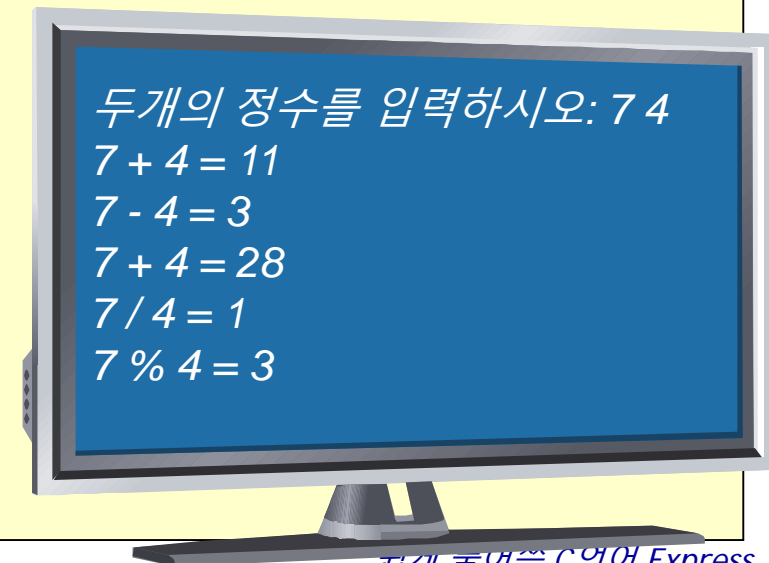
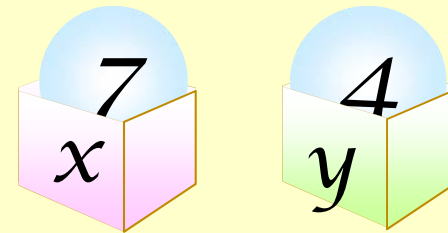
```
    printf("%d / %d = %d", x, y, result);
```

```
    result = x % y;          // 나머지
```

```
    printf("%d %% %d = %d", x, y, result);
```

```
    return 0;
```

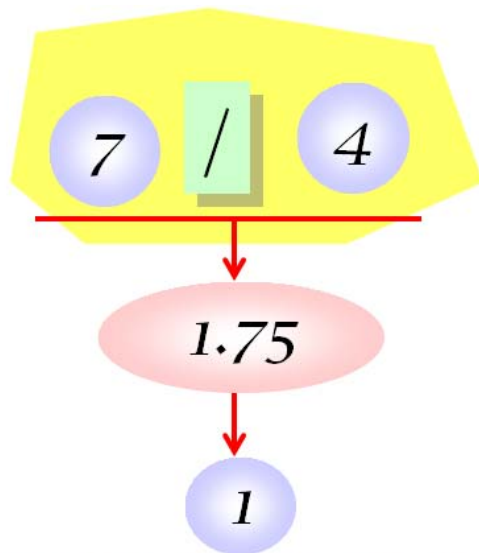
```
}
```



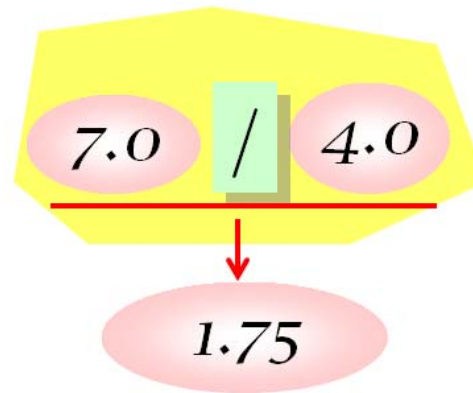


나눗셈 연산자

- 정수형끼리의 나눗셈에서는 결과가 정수형으로 생성하고 부동소수점형끼리는 부동소수점 값을 생성된다.
- 정수형끼리의 나눗셈에서는 소수점 이하는 버려진다.



정수와 정수 끼리의 나눗셈.



실수와 실수 끼리의 나눗셈.

형변환에서
자세히
학습합니다





실수 사칙 연산

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

```
int main()
```

```
{
```

```
    double x, y, result;
```

```
    printf("두개의 실수를 입력하시오: ");
```

```
    scanf("%lf %lf", &x, &y);
```

```
    result = x + y;           // 덧셈 연산을 하여서 결과를 result에 대입
```

```
    printf("%f / %f = %f", x, y, result);
```

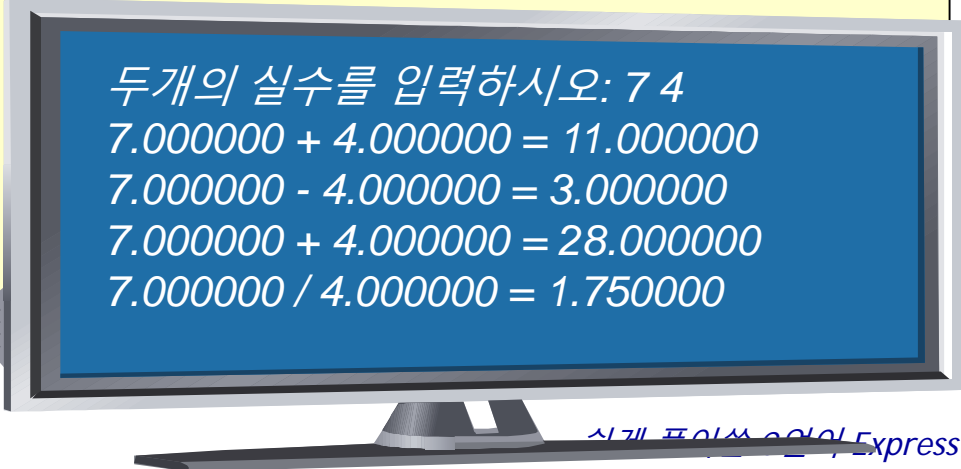
```
    ...
```

```
    result = x / y;
```

```
    printf("%f / %f = %f", x, y, result);
```

```
    return 0;
```

```
}
```



```
두개의 실수를 입력하시오: 7 4  
7.000000 + 4.000000 = 11.000000  
7.000000 - 4.000000 = 3.000000  
7.000000 * 4.000000 = 28.000000  
7.000000 / 4.000000 = 1.750000
```



나머지 연산자

- 나머지 연산자(modulus operator)는 첫 번째 피연산자를 두 번째 피연산자로 나누었을 경우의 나머지를 계산
 - ▣ $10 \% 2$ 는 0이다.
 - ▣ $5 \% 7$ 는 5이다.
 - ▣ $30 \% 9$ 는 3이다.
- 나머지 연산자를 이용한 짝수와 홀수를 구분
 - ▣ $x \% 2$ 가 0이면 짝수
- 나머지 연산자를 이용한 5의 배수 판단
 - ▣ $x \% 5$ 가 0이면 5의 배수

아주
유용한
연산자
입니다.





나머지 연산자

// 나머지 연산자 프로그램

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

```
#define SEC_PER_MINUTE 60 // 1분은 60초
```

```
int main(void)
```

```
{
```

```
    int input, minute, second;
```

```
    printf( " 초를 입력하시요: ");
```

```
    scanf("%d", &input);        // 초단위의 시간을 읽는다.
```

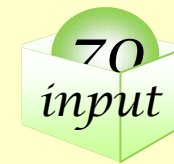
```
    minute = input / SEC_PER_MINUTE; // 몇 분
```

```
    second = input % SEC_PER_MINUTE; // 몇 초
```

```
    printf("%d초는 %d분 %d초입니다. \n",  
           input, minute, second);
```

```
    return 0;
```

```
}
```



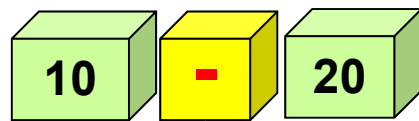
초를 입력하시요: 1000
1000초는 16분 40초 입니다.



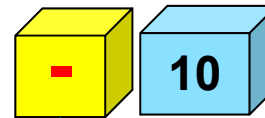
부호 연산자

- 변수나 상수의 부호를 변경

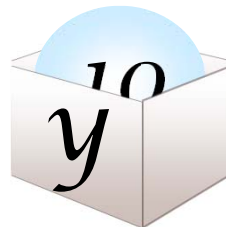
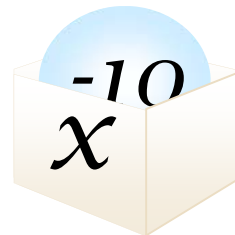
```
x = -10;  
y = -x; // 변수 y의 값은 10이 된다.
```



이항연산자



단항연산자



-는 이항
연산자이기도
하고 단항
연산자이기도
하죠





증감 연산자

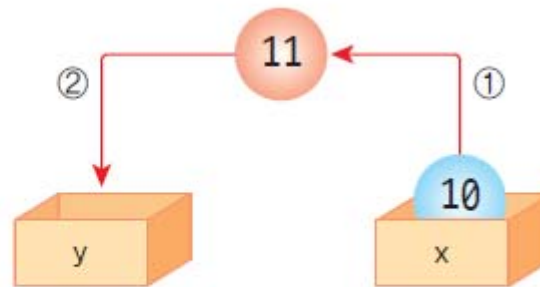
- 증감 연산자: ++, --
- 변수의 값을 하나 증가시키거나 감소시키는 연산자
- ++X;
 - ▣ $x = x + 1;$
- --X;
 - ▣ $x = x - 1;$





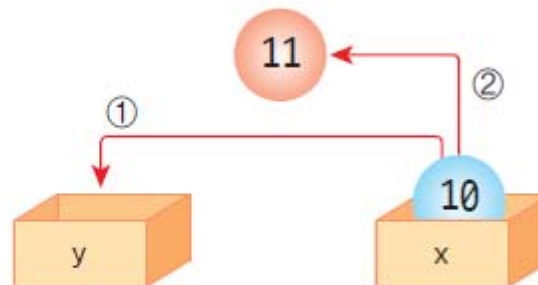
++x와 x++의 차이

`y=++x;`



증가된 x의 값이 y에 대입된다.

`y=x++;`



먼저 대입하고 나중에 증가한다.



증감 연산자 정리

증감 연산자	의미
<code>++x</code>	수식의 값은 증가된 x값이다.
<code>x++</code>	수식의 값은 증가되지 않은 원래의 x값이다.
<code>--x</code>	수식의 값은 감소된 x값이다.
<code>x--</code>	수식의 값은 감소되지 않은 원래의 x값이다.



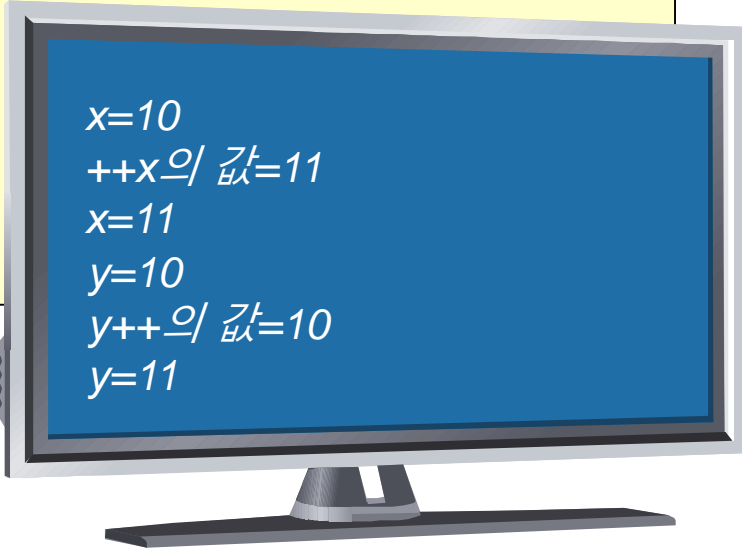
예제: 증감 연산자

```
#include <stdio.h>
int main(void)
{
    int x=10, y=10;

    printf("x=%d\n", x);
    printf("++x의 값=%d\n", ++x);
    printf("x=%d\n\n", x);

    printf("y=%d\n", y);
    printf("y++의 값=%d\n", y++);
    printf("y=%d\n", y);

    return 0;
}
```



```
x=10
++x의 값=11
x=11
y=10
y++의 값=10
y=11
```



중간 점검

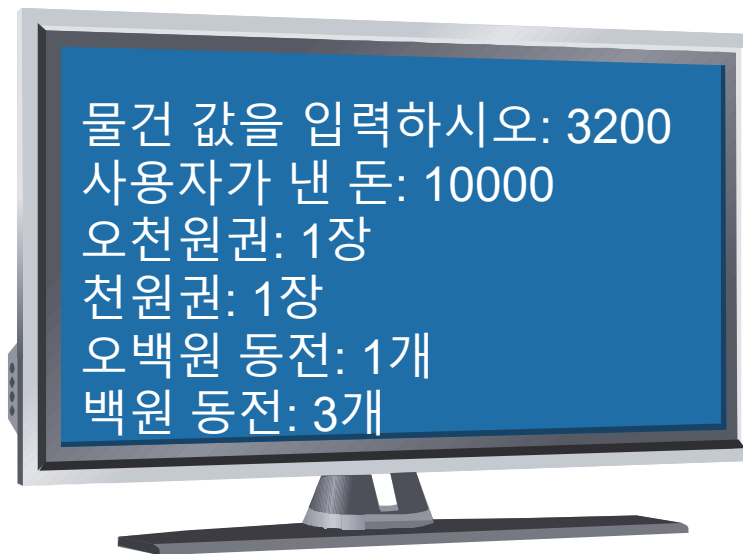
1. 증감 연산자 `x++`를 풀어쓰면 어떻게 되는가?
2. `x++`와 `++x`의 차이점은 무엇인가?
3. `Int x=10; printf("%d \n", (1 + x++) + 2);`의 출력은?





Lab: 거스름돈 계산하기

- 편의점에서 물건을 구입하고 만 원을 냈을 때, 거스름돈의 액수와 점원이 지급해야 할 거스름돈을 화폐와 동전수를 계산하는 프로그램을 작성해보자.





```
#include <stdio.h>
int main(void)
{
    int user, change = 0;
    int price, c5000, c1000, c500, c100;

    printf("물건 값을 입력하시오: ");
    scanf("%d", &price); // 물건 값을 입력받는다.
    printf("사용자가 낸 돈: ");
    scanf("%d", &user);
    change = user - price;      // 거스름돈을 change에 저장
```



`c5000 = change / 5000; // 몫 연산자를 사용하여 5000원권의 개수를 계산한다.`
`change = change % 5000; // 나머지 연산자를 사용하여 남은 잔돈을 계산한다.`

`c1000 = change / 1000; // 남은 잔돈에서 1000원권의 개수를 계산한다.`
`change = change % 1000; //나머지 연산자를 사용하여 남은 잔돈을 계산한다.`

`c500 = change / 500; // 남은 잔돈에서 500원 동전의 개수를 계산한다.`
`change = change % 500; //나머지 연산자를 사용하여 남은 잔돈을 계산한다.`

`c100 = change / 100; // 남은 잔돈에서 100원 동전의 개수를 계산한다.`
`change = change % 100; //나머지 연산자를 사용하여 남은 잔돈을 계산한다.`

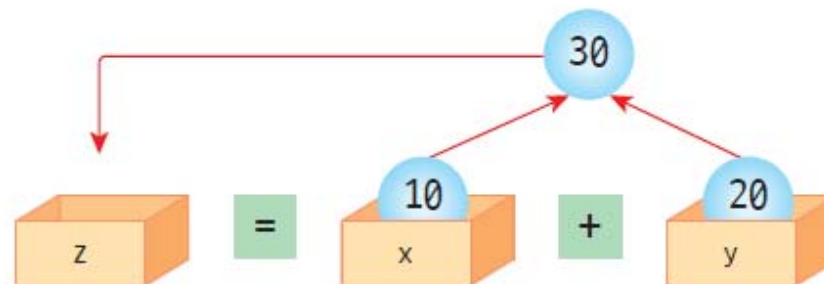
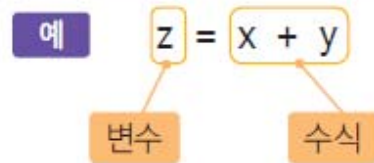
```
printf("오천원권: %d장\n", c5000);  
printf("천원권: %d장\n", c1000);  
printf("오백원 동전: %d개\n", c500);  
printf("백원 동전: %d개\n", c100);  
return 0;
```

}



대입(배정, 할당) 연산자

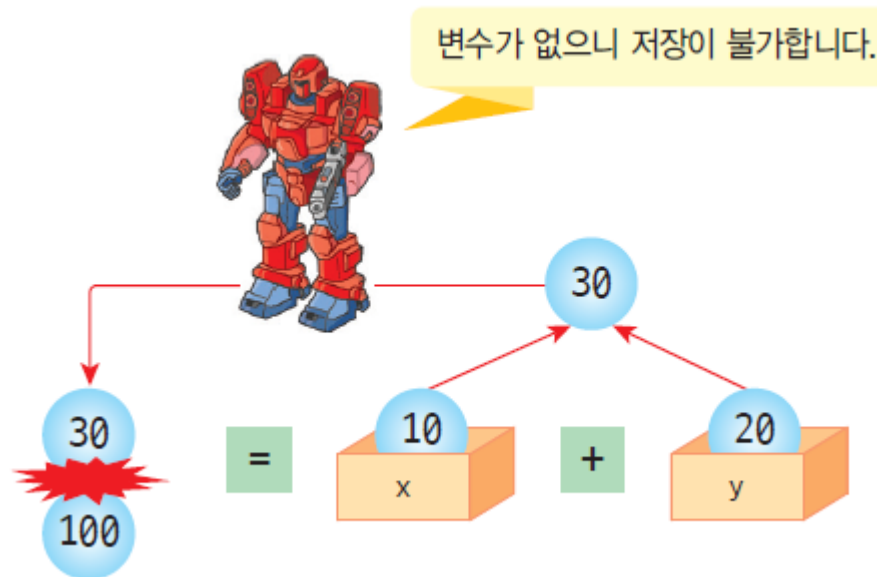
Syntax: 대입 연산자





대입 연산자 주의점

□ $100 = x + y;$ // 컴파일 오류!

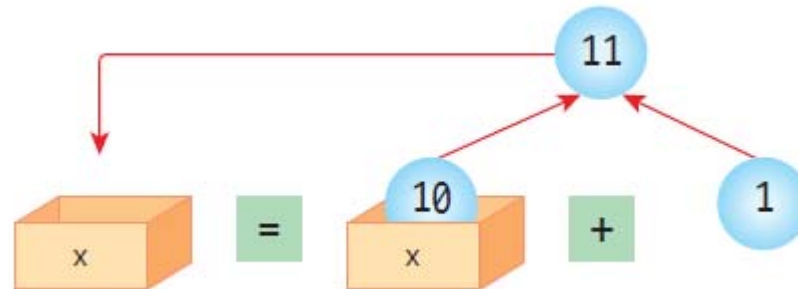




대입 연산자 주의점

수학적으로는 올바르지 않지만 c에서는 올바른 문장임

$x = x + 1;$





대입 연산의 결과값

$$y = 10 + (x = 2 + 7);$$

덧셈연산의 결과값은 9

대입연산의 결과값은 9

덧셈연산의 결과값은 19

대입연산의 결과값은 19

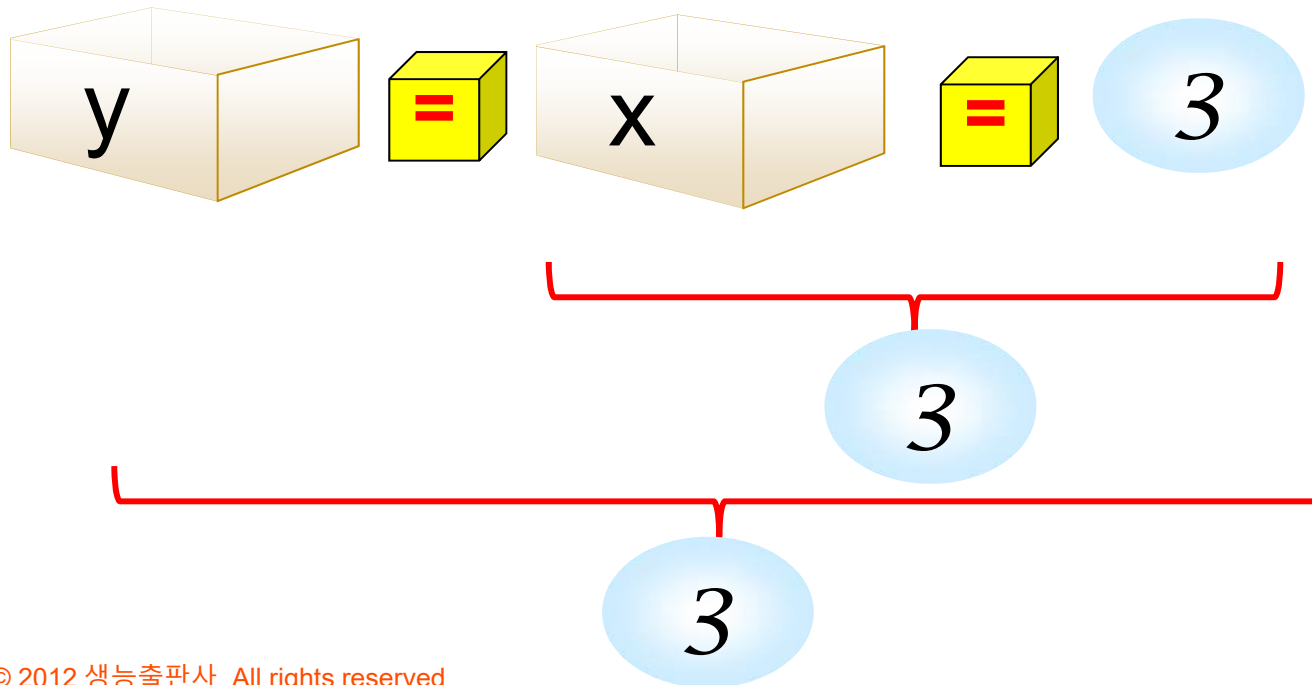
모든 연산에는
결과값이 있고
대입 연산도
결과값이 있습니다.





예제

```
y = x = 3;
```





예제

```
/* 대입 연산자 프로그램 */
```

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

```
int main(void)
```

```
{
```

```
    int x, y;
```

```
    x = 1;
```

```
    printf("수식 x+1의 값은 %d\n", x+1);
```

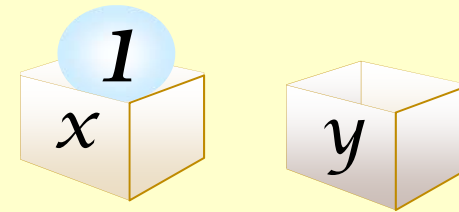
```
    printf("수식 y=x+1의 값은 %d\n", y=x+1);
```

```
    printf("수식 y=10+(x=2+7)의 값은 %d\n", y=10+(x=2+7));
```

```
    printf("수식 y=x=3의 값은 %d\n", y=x=3);
```

```
    return 0;
```

```
}
```



수식 x+1의 값은 2

수식 y=x+1의 값은 2

수식 y=10+(x=2+7)의 값은 19

수식 y=x=3의 값은 3



복합 대입 연산자

- 복합 대입 연산자란 +=처럼 대입연산자 =와 산술연산자를 합쳐 놓은 연산자
- 소스를 간결하게 만들 수 있음

$x += y$

$x = x + y$ 와 의미가 같음!



복합 대입 연산자

복합 대입 연산자	의미
$x += y$	$x = x + y$
$x -= y$	$x = x - y$
$x *= y$	$x = x * y$
$x /= y$	$x = x / y$
$x \% = y$	$x = x \% y$
$x \& = y$	$x = x \& y$
$x = y$	$x = x y$
$x \wedge = y$	$x = x \wedge y$
$x >> = y$	$x = x >> y$
$x << = y$	$x = x << y$

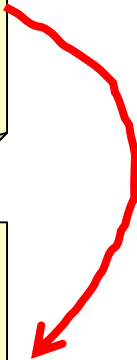


Quiz

- 다음 수식을 풀어서 다시 작성하면?

$x^* = y + 1$
 $x \% = x + y$

$x = x^* (y + 1)$
 $x = x \% (x + y)$





복합 대입 연산자

// 복합 대입 연산자 프로그램

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

```
int main(void)
```

```
{
```

```
    int x = 10, y = 10, z = 33;
```

```
    x += 1;
```

```
    y *= 2;
```

```
    z %= 10 + 20;
```

```
    printf("x = %d   y = %d   z = %d \n", x, y, z);
```

```
    return 0;
```

```
}
```

