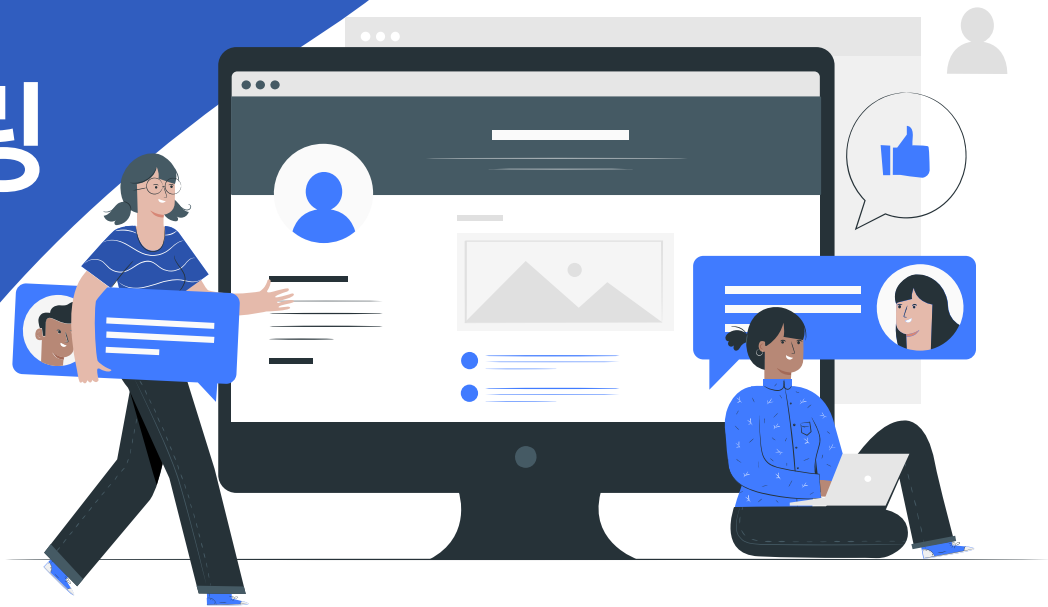


2주차 튜터링



연산자



목차

- 연산자의 종류
- 산술 연산자
- 대입 연산자
 - 상수와 변수 복습
- 복합 대입 연산자
 - 복합 대입 연산자를 사용하는 이유
- 증감 연산자
 - 전위형과 후위형
- 관계연산자
- 논리연산자
- 연산자 우선순위와 연산 방향이 있는 이유

연산자의 종류

분류	연산자
대입 연산자	=
산술 연산자	+, -, *, /, %
복합 대입 연산자	+=, -=, *=, /=, %=
증감 연산자	++, --
관계 연산자	>, <, ==, !=, >=, <=
논리 연산자	&&, , !



산술 연산자

```
0  
4 int main() {  
5     int x = 2;  
6     int y = 3;  
7  
8     printf("%d\n", x + y);  
9     printf("%d\n", x - y);  
10    printf("%d\n", x * y);  
11    printf("%d\n", x / y);  
12    printf("%d\n", x % y);  
13  
14    return 0;  
15 }  
~
```

- 다들 아는 내용이죠~^^



대입 연산자

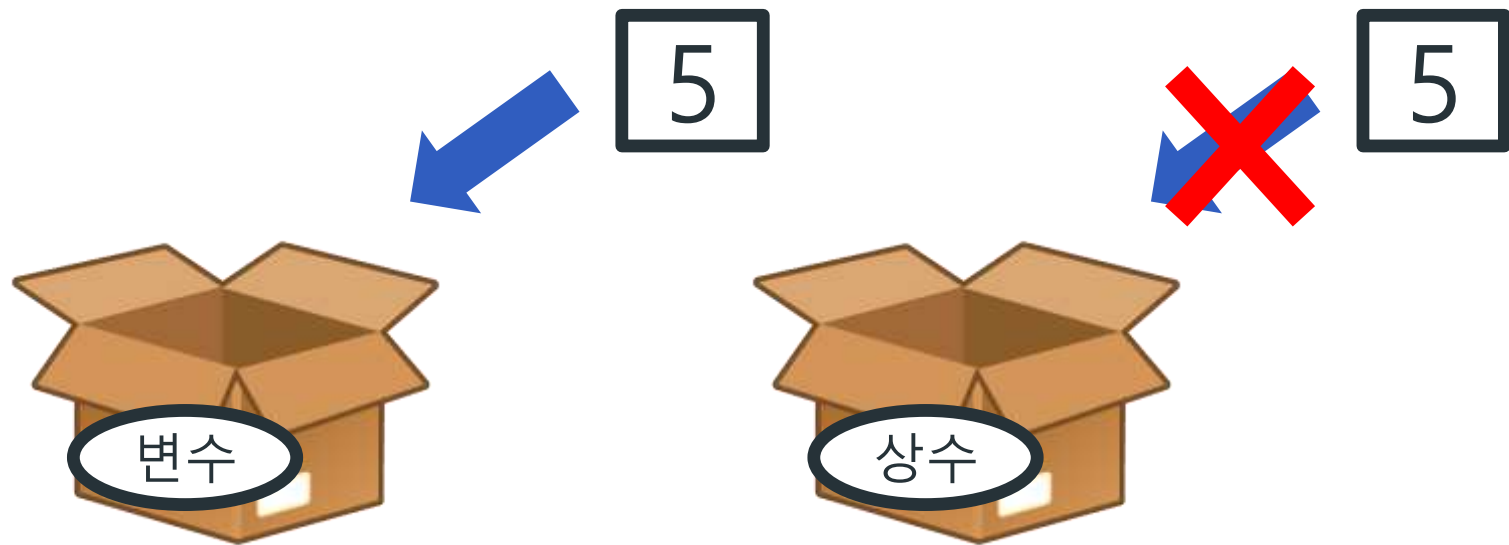
- C문법에서 =는 '같다'는 뜻이 아니라 **'대입' 한다는 뜻입니다.**
 - 대입을 할 때는 **오른쪽에서 왼쪽으로 대입**을 하며,
변수를 대입합니다.
- 
- 

상수와 변수 복습

```
1  #include <stdio.h>
2
3  #define dirthday 512
4
5  int main()
6  {
7
8      int const day = 512;
9      printf("%d\n", dirthday);
10     printf("%d\n", day);
11
12     return 0;
13 }
14
15
```

- 변수 = 변할 수 있는 수
- 상수 = 변하지 않는 수
- 이 코드에서 상수는 define과 const를 선언한 birthday와 day이다.

대입 연산자



복합 대입 연산자

$$A = A + B \quad \rightarrow \quad A += B$$

$$A = A - B \quad \rightarrow \quad A -= B$$

$$A = A * B \quad \rightarrow \quad A *= B$$

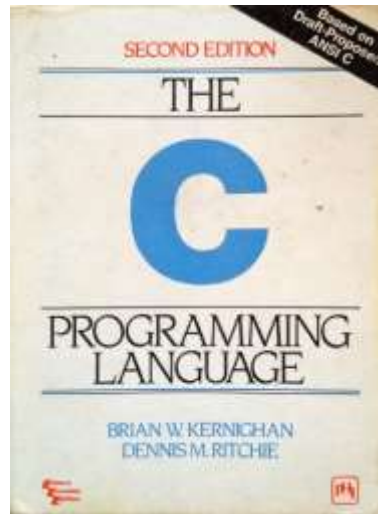
$$A = A / B \quad \rightarrow \quad A /= B$$

$$A = A \% B \quad \rightarrow \quad A \% = B$$

=> 그냥 공식처럼 알아두세요.

복합 대입 연산자를 사용하는 이유

- 브라이언 커닝햄(Brian Kernighan) 과 데니스 리치(Dennis Ritchie) 가 작성한 'The C Programming Language'라는 책에 보면...



복합 대입 연산자를 사용하는 이유

간결함과는 별도로 할당 연산자는 사람들이 생각하는 방식에 더 잘 부합한다는 이점이 있습니다. 우리는 "i에 2를 더하라" 또는 "i를 2씩 증가시키라"고 말하지, "i를 취하고, 2를 더한 다음 그 결과를 다시 i에 넣자"라고 말하지 않습니다. 따라서 `i += 2`. 또한 다음과 같은 복잡한 표현에 대해서는

```
yyval[yyvsp[p3+p4] + yyvp[p1+p2]] += 2
```

대입 연산자는 코드를 이해하기 쉽게 만듭니다. 독자는 두 개의 긴 표현식이 실제로 동일한지 힘들게 확인하거나 왜 그렇지 않은지 궁금해할 필요가 없기 때문입니다. 그리고 할당 연산자는 컴파일러가 보다 효율적인 코드를 생성하는 데 도움이 될 수도 있습니다.

→ 간결함 때문에 사용한다 한다.

증감 연산자

- ++ 증가 연산자 : 변수가 가지고 있는 값을 1증가
- -- 감소 연산자 : 변수가 가지고 있는 값을 1 감소

덧셈 연산자(이항 연산자)

```
int i = 5;  
i = i + 1;
```

// i값에 1을 더하고 다시 i에 대입

증가 연산자 (단항 연산자)

```
int i = 5;  
i++;
```

// i값에 1 증가

- 덧셈 연산자는 두 개의 메모리가 연산에 사용. (ADD 명령어)
- 증가 연산자는 한 개의 메모리가 연산에 사용. (INC 명령어)

증감 연산자

- 후위형 : 연산하고 난 뒤 값을 증감
- 전위형 : 값을 먼저 증감한 뒤 연산

전위형

```
int sum, i = 5;  
sum = ++i;
```

// i값이 6으로 증가한 뒤 sum에 대입

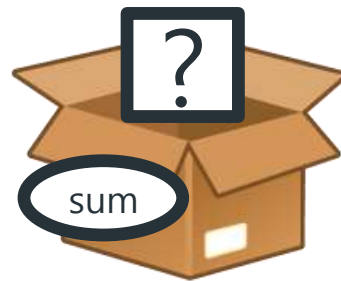
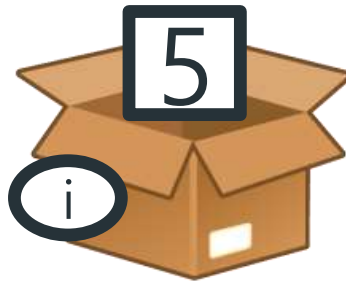
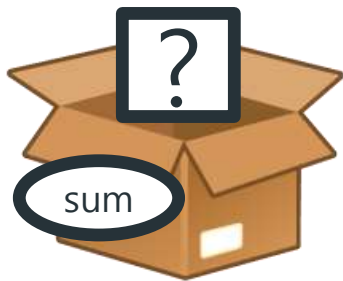
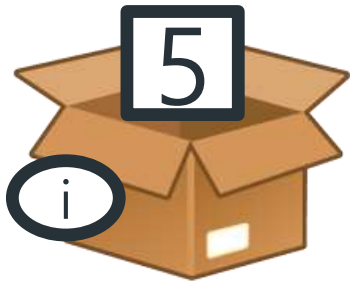
증가 연산자 (단항 연산자)

```
int sum, i = 5;  
sum = i++;
```

// i값을 먼저 sum에 대입한 뒤 i 증가

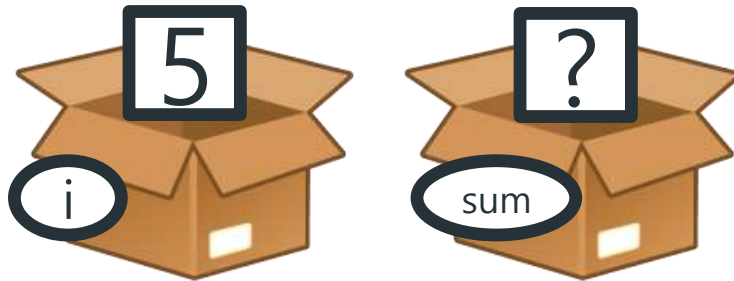
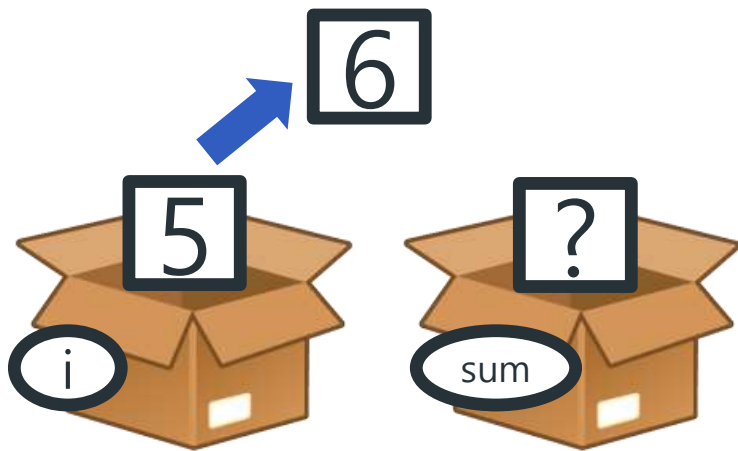
증감 연산자

- 후위형 : 연산하고 난 뒤 값을 증감 (i값이 6으로 증가한 뒤 sum에 대입)
- 전위형 : 값을 먼저 증감한 뒤 연산 (i값을 먼저 sum에 대입한 뒤 i 증가)



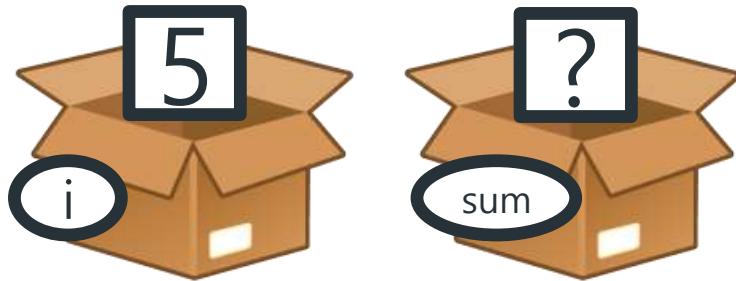
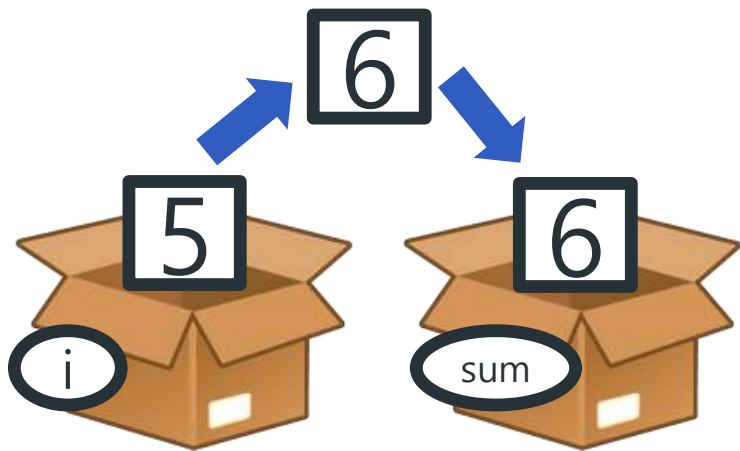
증감 연산자

- 후위형 : 연산하고 난 뒤 값을 증감 (i 값이 6으로 증가한 뒤 sum 에 대입)
- 전위형 : 값을 먼저 증감한 뒤 연산 (i 값을 먼저 sum 에 대입한 뒤 i 증가)



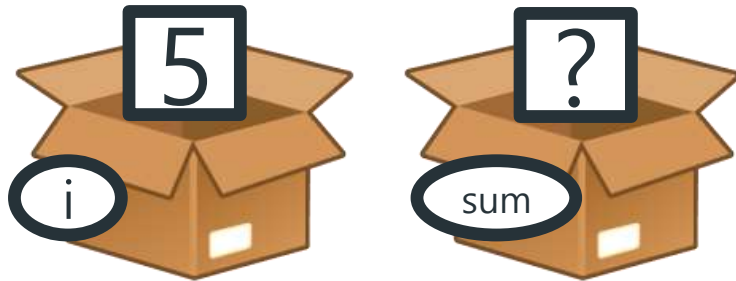
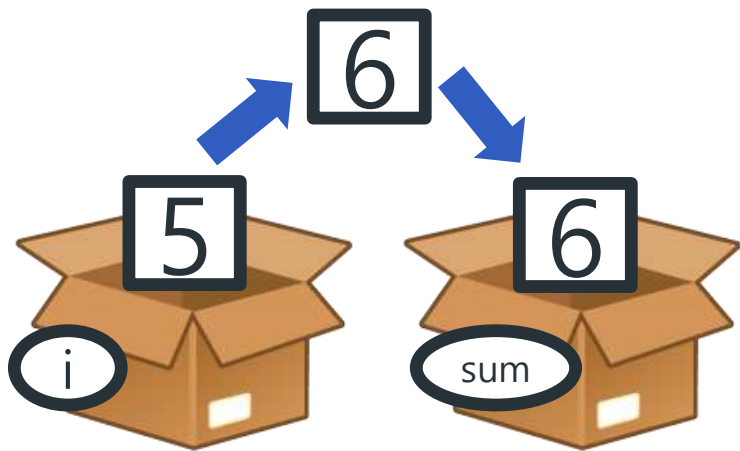
증감 연산자

- 후위형 : 연산하고 난 뒤 값을 증감 (i값이 6으로 증가한 뒤 sum에 대입)
- 전위형 : 값을 먼저 증감한 뒤 연산 (i값을 먼저 sum에 대입한 뒤 i 증가)



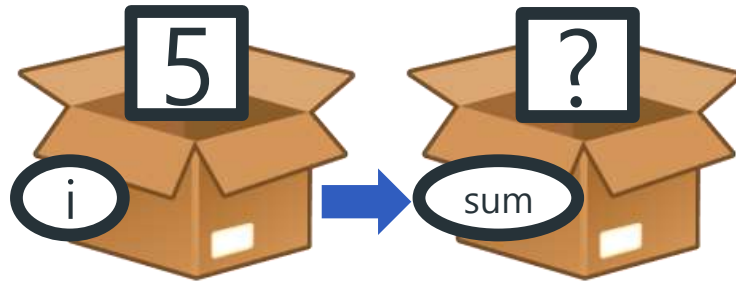
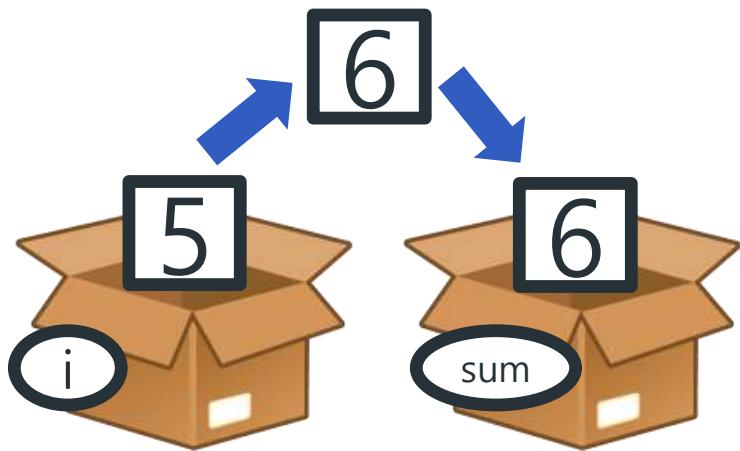
증감 연산자

- 후위형 : 연산하고 난 뒤 값을 증감 (i 값이 6으로 증가한 뒤 sum 에 대입)
- 전위형 : 값을 먼저 증감한 뒤 연산 (i 값을 먼저 sum 에 대입한 뒤 i 증가)



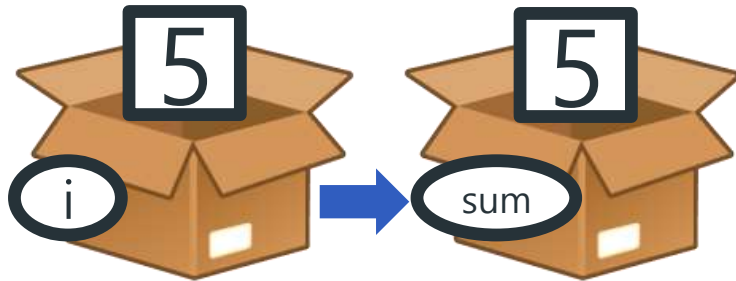
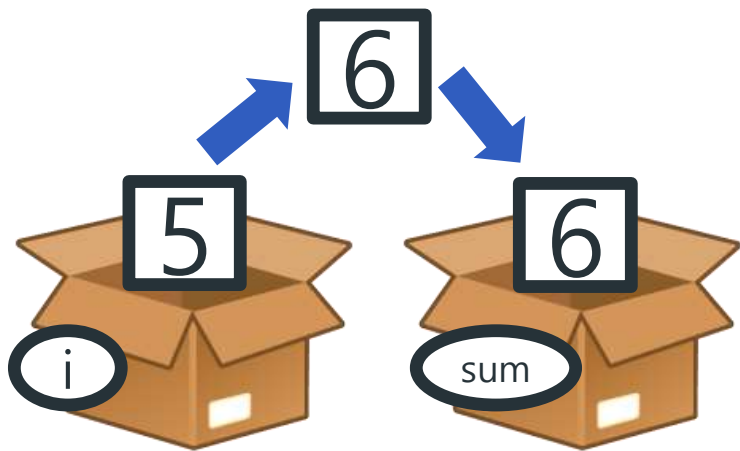
증감 연산자

- 후위형 : 연산하고 난 뒤 값을 증감 (i값이 6으로 증가한 뒤 sum에 대입)
- 전위형 : 값을 먼저 증감한 뒤 연산 (i값을 먼저 sum에 대입한 뒤 i 증가)



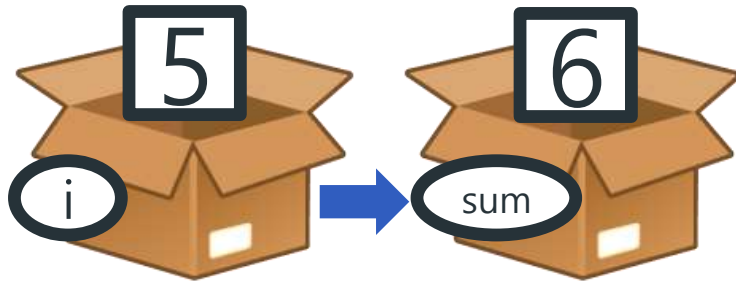
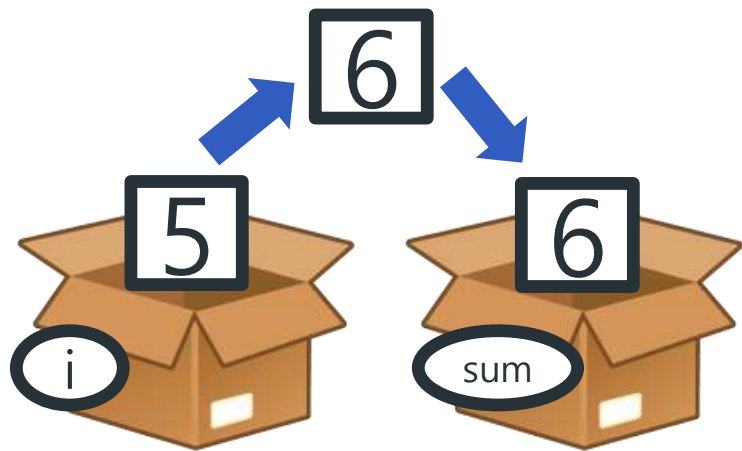
증감 연산자

- 후위형 : 연산하고 난 뒤 값을 증감 (i값이 6으로 증가한 뒤 sum에 대입)
- 전위형 : 값을 먼저 증감한 뒤 연산 (i값을 먼저 sum에 대입한 뒤 i 증가)



증감 연산자

- 후위형 : 연산하고 난 뒤 값을 증감 (i값이 6으로 증가한 뒤 sum에 대입)
- 전위형 : 값을 먼저 증감한 뒤 연산 (i값을 먼저 sum에 대입한 뒤 i 증가)



관계 연산자

- 두 값을 비교하여 그 결과 값을 진리값(참, 거짓)으로 얻는 연산자
- 연산의 결과가 참이면 1, 거짓이면 0

관계 연산자	연산 특성	관계 연산자	연산 특성
$A < B$	A가 B보다 작으면 참	$A > B$	A가 B보다 크면 참
$A \leq B$	A가 B보다 작거나 같으면 참	$A \geq B$	A가 B보다 크거나 같으면 참
$A == B$	A와 B가 같으면 참	$A != B$	A와 B가 같지 않으면 참

논리 연산자

- A와 B의 상황을 일정한 규칙(AND, OR, NOT)으로 연결해주는 연산자
- A와 B는 진리값(참, 거짓)을 사용
- 참 : 0이 아닌 모든 값
- 거짓 : 0

논리 연산자

A	B	A && B	A B	!A
거짓(0)	거짓(0)	거짓(0)	거짓(0)	참(1)
거짓(0)	참(1)	거짓(0)	참(1)	참(1)
참(1)	거짓(0)	거짓(0)	참(1)	거짓(0)
참(1)	참(1)	참(1)	참(1)	거짓(0)

논리 연산자

종 류		연 산 자		연산방향	우선순위	
?		[], .		➡	1(높음)	
단항 연산자		변수++, 변수--		⬅	2	
		++변수, --변수, +, -, ~, !, (자료형)		⬅	3	
이 항 연 산 자	산술 연산자	*, /, %		➡	4	
		+, -		➡	5	
		비트 산술 연산자	<<, >>, >>>	➡	6	
	비교 연산자	대소 비교 연산자	<, >, <=, >=	instanceof	➡	7
		등가 비교 연산자	==, !=		➡	8
	논리 연산자	비트 논리 연산자	&		➡	9
			^		➡	10
					➡	11
		&&		➡	12	
				➡	13	
삼항 연산자		조건 ? 참 : 거짓		⬅	14	
대입 연산자		=, +=, -=, *=, /=, %=, <<=, >>=, >>>=, &=, ^=, =		⬅	15(낮음)	



END