

개정3판

Visual
Studio
2017

쉽게 풀어쓴

C언어
EXPRESS



천인국 지음

제5장 수식과 연산자

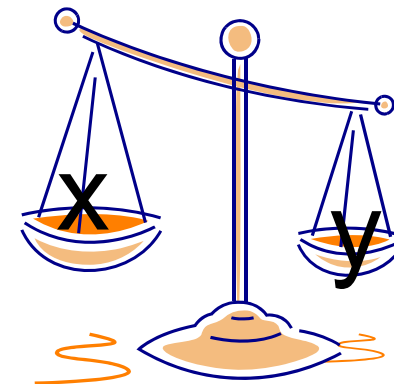


관계 연산자

- 두개의 피연산자를 비교하는 연산자
- 결과값은 참(1) 아니면 거짓(0)

$x == y$

x 와 y의
값이
같은지
비교한
다.





관계 연산자

연산자	의미
$x == y$	x와 y가 같은가?
$x != y$	x와 y가 다른가?
$x > y$	x가 y보다 큰가?
$x < y$	x가 y보다 작은가?
$x >= y$	x가 y보다 크거나 같은가?
$x <= y$	x가 y보다 작거나 같은가?





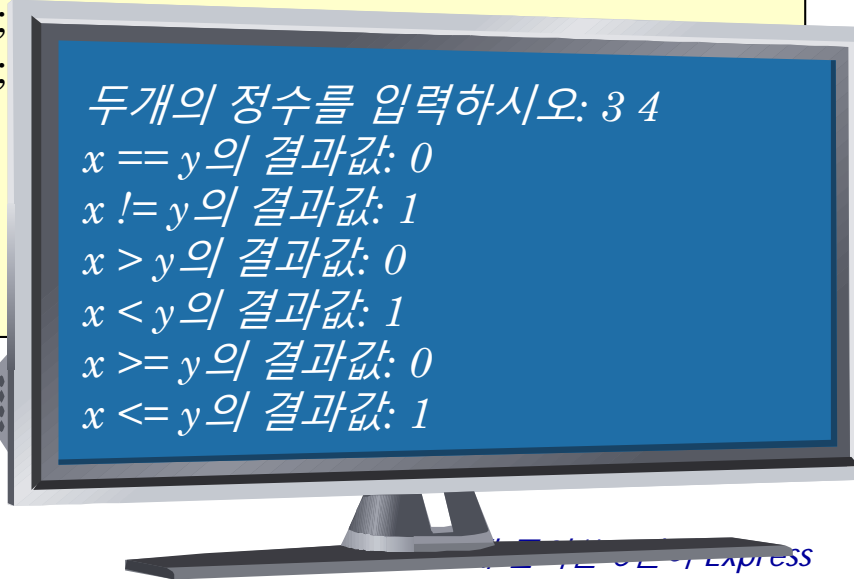
예제

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

    printf("두개의 정수를 입력하시오: ");
    scanf("%d%d", &x, &y);

    printf("x == y의 결과값: %d", x == y);
    printf("x != y의 결과값: %d", x != y);
    printf("x > y의 결과값: %d", x > y);
    printf("x < y의 결과값: %d", x < y);
    printf("x >= y의 결과값: %d", x >= y);
    printf("x <= y의 결과값: %d", x <= y);

    return 0;
}
```



두개의 정수를 입력하시오: 3 4
x == y의 결과값: 0
x != y의 결과값: 1
x > y의 결과값: 0
x < y의 결과값: 1
x >= y의 결과값: 0
x <= y의 결과값: 1



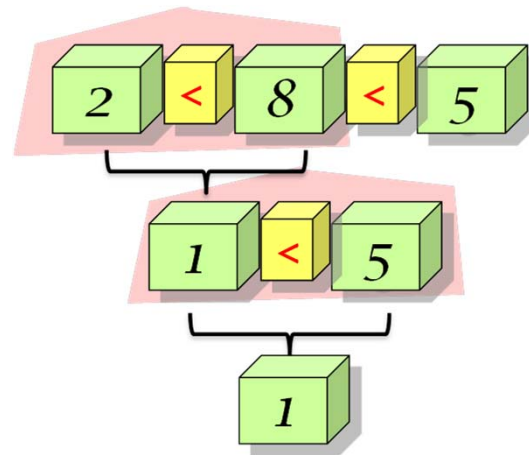
관계 연산자 사용시 주의점

- 대입 연산자와 관계 연산자의 혼동
- $(x = y)$
 - ▣ y 의 값을 x 에 대입한다. 이 수식의 값은 x 의 값이다.
- $(x == y)$
 - ▣ x 와 y 가 같으면 1, 다르면 0이 수식의 값이 된다.
 - ▣ $(x == y)$ 를 $(x = y)$ 로 잘못 쓰지 않도록 주의!



관계 연산자 사용시 주의점

- 수학에서처럼 $2 < x < 5$ 와 같이 작성하면 잘못된 결과가 나온다.



- 올바른 방법: $(2 < x) \&\& (x < 5)$



관계 연산자 사용시 주의점

- 실수를 비교하는 경우
- 예: $(1e32 + 0.01) > 1e32$
 - ▣ -> 양쪽의 값이 같은 것으로 간주되어서 거짓





중간 점검

1. 관계 수식의 결과로 생성될 수 있는 값은 무엇인가?
2. $(3 \geq 2) + 5$ 의 값은?



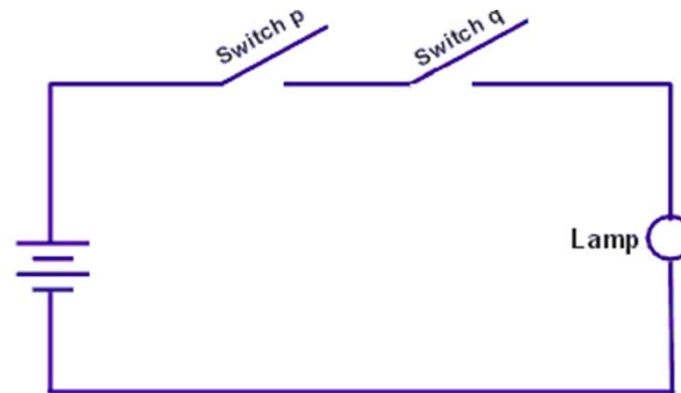


논리 연산자

- 여러 개의 조건을 조합하여 참과 거짓을 따지는 연산자
- 결과값은 참(1) 아니면 거짓(0)

`x && y`

x와 y가 모두 참인
경우에만 참이 된다.





논리 연산자

연산자	의미
$x \ \&\& \ y$	AND 연산, x 와 y 가 모두 참이면 참, 그렇지 않으면 거짓
$x \ \ y$	OR 연산, x 나 y 중에서 하나만 참이면 참, 모두 거짓이면 거짓
$!x$	NOT 연산, x 가 참이면 거짓, x 가 거짓이면 참





AND 연산자

27 800

(*age* <= 30) && (*toeic* >= 700)

참 (1) 참 (1)

참 (1)



OR 연산자

27 699

$(age \leq 30) \parallel (toeic \geq 700)$

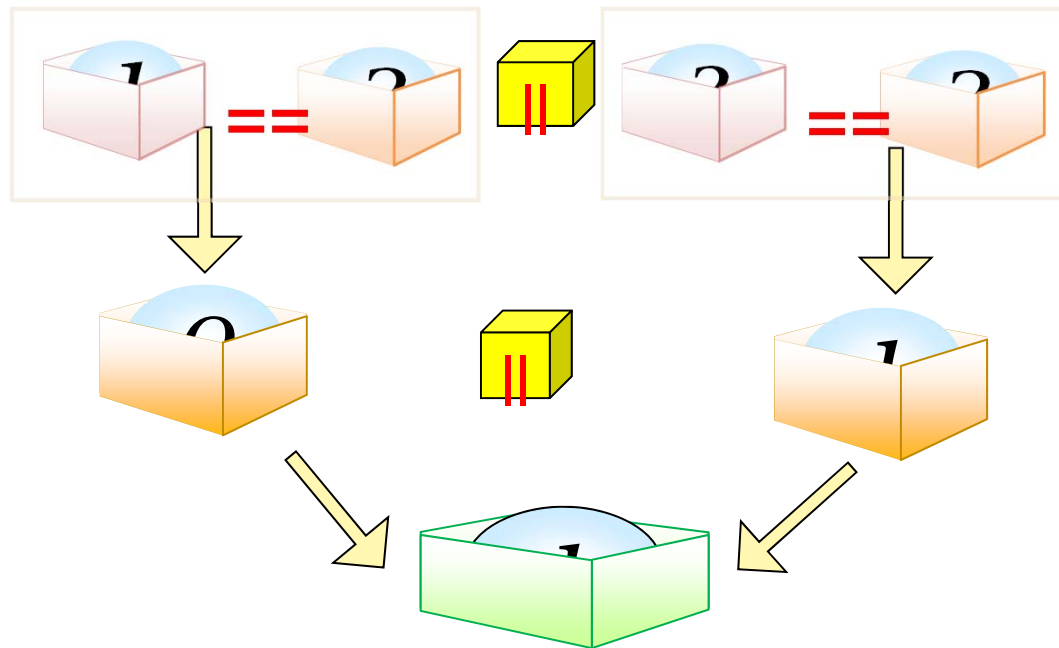
참 (1) 거짓 (0)

참 (1)



논리 연산자의 계산 과정

- 논리 연산의 결과값은 항상 1 또는 0이다.
- (예) $(1 == 2) \parallel (2 == 2)$



0이 아닌 값을
참으로
취급하지만 논리
연산의 결과값은
항상 1 또는
0입니다.





NOT 연산자

- 피연산자의 값이 참이면 연산의 결과값을 거짓으로 만들고, 피연산자의 값이 거짓이면 연산의 결과값을 참으로 만든다.



- `result = !1;` `// result에는 0가 대입된다.`
- `result = !(2==3);` `// result에는 1이 대입된다.`



참과 거짓의 표현 방법

- 관계 수식이나 논리 수식이 만약 참이면 1이 생성되고 거짓이면 0이 생성된다.
- 피연산자의 참, 거짓을 가릴 때에는 0이 아니면 참이고 0이면 거짓으로 판단한다.
 - ▣ 음수도 참

!0	// 식의 값은 1
!3	// 식의 값은 0
!-3	// 식의 값은 0



논리 연산자의 예

- “x는 1, 2, 3중의 하나인가”
 - ▣ `(x == 1) || (x == 2) || (x == 3)`

- “x가 60이상 100미만이다.”
 - ▣ `(x >= 60) && (x < 100)`

- “x가 0도 아니고 1도 아니다.”
 - ▣ `(x != 0) && (x != 1)` // x≠0 이고 x≠1이다.



예제

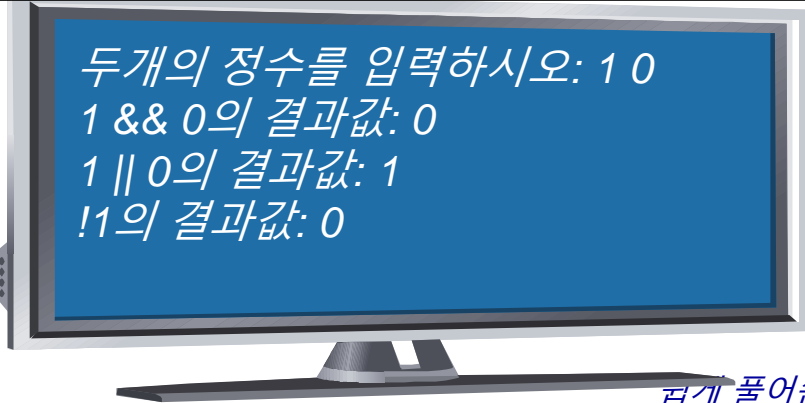
```
#include <stdio.h>

int main(void)
{
    int x, y;

    printf("두개의 정수를 입력하시오: ");
    scanf("%d%d", &x, &y);

    printf("%d && %d의 결과값: %d", x, y, x && y);
    printf("%d || %d의 결과값: %d", x, y, x || y);
    printf("!x의 결과값: %d", x, !x);

    return 0;
}
```



```
두개의 정수를 입력하시오: 1 0
1 && 0의 결과값: 0
1 || 0의 결과값: 1
!1의 결과값: 0
```



단축 계산

- && 연산자의 경우, 첫번째 피연산자가 거짓이면 다른 피연산자들을 계산하지 않는다.

`(2 > 3) && (++x < 5)`

- || 연산자의 경우, 첫번째 피연산자가 참이면 다른 피연산자들을 계산하지 않는다.

`(3 > 2) || (--x < 5)`



첫번째 연산자가
거짓이면 다른
연산자는 계산할
필요가 없겠군!!

++나 --는
실행이
안될 수도
있으니
주의하세
요.





lab: 윤년

- 윤년의 조건
 - ▣ 연도가 4로 나누어 떨어진다.
 - ▣ 100으로 나누어 떨어지는 연도는 제외한다.
 - ▣ 400으로 나누어 떨어지는 연도는 윤년이다.

February 29

+1
Day





lab: 윤년

□ 윤년의 조건을 수식으로 표현

▣ `((year % 4 == 0) && (year % 100 != 0)) || (year % 400 == 0)`






lab: 윤년

```
#include <stdio.h>
int main(void)
{
    int year, result;

    printf("연도를 입력하시오: ");
    scanf("%d", &year);

    result = ((year % 4 == 0) && (year % 100 != 0)) || (year % 400 == 0);
    printf("result=%d \n", result);

    return 0;
}
```



연도를 입력하시오: 2012
result=1



중간 점검

1. 다음의 조건에 해당하는 논리 연산식을 만들어 보시오. 변수는 적절하게 선언되어 있다고 가정한다.

“무주택 기간 3년 이상, 가구주의 연령이 40세 이상, 가족의 수가 3명 이상”

2. 상수 10은 참인가 거짓인가?

3. 수식 !3의 값은?

4. 단축 계산의 예를 들어보라.





조건 연산자

$x > y$ 가 참이면 x 가 수식의 값이 된다.

$max_value = (x > y) ? x : y;$

$x > y$ 가 거짓이면 y 가 수식의 값이 된다.

```
absolute_value = (x > 0) ? x: -x;           // 절대값 계산  
max_value = (x > y) ? x: y;                 // 최대값 계산  
min_value = (x < y) ? x: y;                 // 최소값 계산  
(age > 20) ? printf("성인\n"): printf("청소년\n");
```



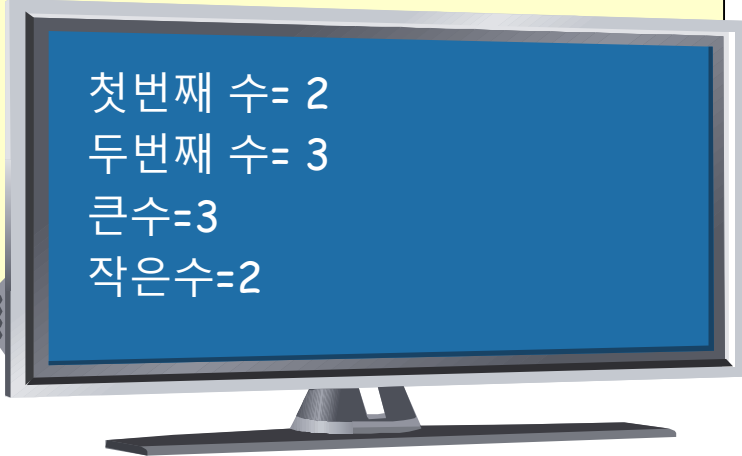
예제

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

    printf("첫번째 수=");
    scanf("%d", &x);
    printf("두번째 수=");
    scanf("%d", &y);

    printf("큰수=%d \n", (x > y) ? x : y);
    printf("작은수=%d \n", (x < y) ? x : y);

    return 0;
}
```



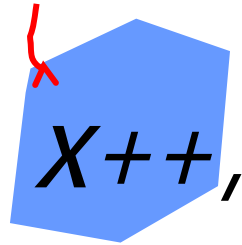
첫번째 수= 2
두번째 수= 3
큰수=3
작은수=2



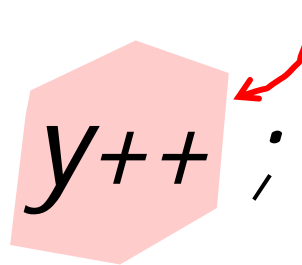
coma 연산자

- 콤마로 연결된 수식은 순차적으로 계산된다.

먼저 계산된다.



나중에 계산된다.



`X++, y++;`

어떤
문장이든지
순차적으로
실행됩니다.

