
C 프로그래밍 및 실습

4. 수식과 연산자

세종대학교

목차

- 1) 수식과 연산자 개요
- 2) 산술 연산자
- 3) 대입 연산자
- 4) 관계 연산자
- 5) 논리 연산자
- 6) 그 외 연산자
- 7) 연산자 우선순위와 결합 수칙

1) 수식과 연산자 개요

■ 연산자

- 데이터를 가공하고 처리하기 위한 가장 기본 도구
- 연산 종류에 따른 분류: 산술 연산자, 관계 연산자, 논리 연산자, 증감 연산자, 비트 연산자, 대입 연산자, 조건 연산자 등
- 피연산자 개수에 따른 분류: 단항 연산자, 이항 연산자, 삼항 연산자

■ 수식

- 피연산자들과 연산자의 조합으로 어떠한 값을 갖는 요소

`num = 10;`

⇒ 상수인 10이 수식

`num1 = num;`

⇒ 변수인 num이 수식

`num2 = num + 1;`

⇒ 연산식인 num+1이 수식

2) 산술 연산자

- 산술 연산자 종류

- 사칙 연산자(+, -, *, /)와 나머지 연산자(%)가 있음
- **+** (더하기), **-** (빼기), ***** (곱하기)

```
int math = 99, korean = 90, science = 94;  
  
// 과목의 총합을 변수 total에 저장  
int total = math + korean + science;  
  
printf("총점 : %d\n", total);  
  
return 0;
```

2) 산술 연산자

- 나누기와 나머지 연산자
 - 피연산자의 자료형에 따라 계산 결과 다름

연산자	정수 연산	부동소수 연산
/	몫	실수 나눗셈
%	나머지	(정의 되지 않음)

11 / 4 ⇒ 연산 결과: 2

11 % 4 ⇒ 연산 결과: 3

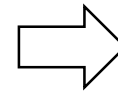
11.0 / 4.0 ⇒ 연산 결과: 2.75

11.0 % 4.0 ⇒ 컴파일 오류

2) 산술 연산자

- 나머지 연산자 활용예: 정수의 자릿수 구하기
 - 일의 자릿수 계산

```
int d = 2715 % 10;  
printf("일의 자릿수: %d\n", d);
```

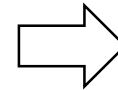


실행 결과

일의 자릿수: 5

- 백의 자릿수 계산

```
int d = 2715 / 100 % 10;  
printf("백의 자릿수: %d\n", d);
```



실행 결과

백의 자릿수: 7

✓ 계산 과정

✓ $2715 / 100 \rightarrow 27$ (백미만 자릿수 제거)

✓ $27 \% 10 \rightarrow 7$ (일의 자릿수 추출)

2) 산술 연산자

- [예제 4.1]
 - 다음 연산의 결과를 예측해보고, 프로그램을 작성하여 확인해보자.

$$\checkmark 5 / 2 * 3$$

$$\checkmark 3 * 5 / 2$$

2) 산술 연산자

- 연산 순서

- 연산자 우선순위: $*$, $/$, $%$ 연산자는 $+$, $-$ 보다 먼저 적용
✓ 적용 순서를 변경하고자 하는 경우 소괄호 사용

- 결합 수칙: 왼쪽에서 오른쪽 방향으로 적용

- 예)

✓ $2 * 5 + 28 / 6 \rightarrow$ 결과 값: 14, 연산 순서: $*$ \Rightarrow $/$ \Rightarrow $+$

✓ $2 * (5 + 28) / 6 \rightarrow$ 결과 값: 11, 연산 순서: $+$ \Rightarrow $*$ \Rightarrow $/$

✓ $10 / 2 * 5 \rightarrow$ 결과 값: 25, 연산 순서: $/$ \Rightarrow $*$

2) 산술 연산자

- 산술 연산과 자료형

- 연산 결과도 자료형이 정해져 있어야 함
- 산술 연산의 경우 피연산자의 자료형에 따라 연산 결과값의 자료형이 결정됨
 - ✓ 정수형과 정수형 \rightarrow 정수형
 - ✓ $5 / 2 \rightarrow$ 정수 2
 - ✓ 부동소수형과 부동소수형 \rightarrow 부동소수형
 - ✓ $5.0 / 2.0 \rightarrow$ 부동소수 2.5
 - ✓ 정수형과 부동소수형 \rightarrow 부동소수형 (정보 손실 방지)
 - ✓ $5.0 / 2 \rightarrow$ 부동소수 2.5
 - ✓ $5 / 2.0 \rightarrow$ 부동소수 2.5

2) 산술 연산자

- 산술 연산과 자료형

- 앞의 규칙은 연산자 별로 적용

✓ $3 / 2 * 4.0 \rightarrow 1 * 4.0 \rightarrow 4.0$

✓ $3 / 2.0 * 4 \rightarrow 1.5 * 4 \rightarrow 6.0$

- 앞의 규칙은 변수에도 동일하게 적용

✓ 각 x의 결과는?

```
int a = 5, b = 2;  
double x, c = 2.0;
```

```
x = a / b;
```

```
x = a / 2;
```

```
x = a / c;
```

```
x = a / 2.0;
```

2) 산술 연산자

- 피연산자가 모두 정수형인데, 부동소수 연산을 하고 싶으면?
 - 명시적 형변환 이용
 - ✓ 아래 예에서 각 x의 결과는?
 - ✓ 형 변환이 적용되는 범위에 주의

```
int a = 5, b = 2;
```

```
double x;
```

```
x = (double) a / b;    → a의 자료형을 변환
```

```
x = (double) (a / b); → a/b의 결과 값의 자료형 변환
```


목차

- 1) 수식과 연산자 개요
- 2) 산술 연산자
- 3) **대입 연산자**
- 4) 관계 연산자
- 5) 논리 연산자
- 6) 그 외 연산자
- 7) 연산자 우선순위와 결합 수칙

3) 대입 연산자

- 대입 연산자 =

- 연산자 오른쪽 수식의 값을 왼쪽 변수에 대입
(수학의 등호와 완전히 다른 의미)



변수 = 수식

✓ $a = c$; ➔ '변수 a'에 '변수 c의 값' 대입

✓ $a = a + 1$; ➔ '변수 a'에 '변수 a의 값에 1 더한 값' 대입

- 다음은 컴파일 에러 발생 (의미적으로 불가능)

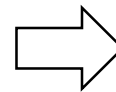
✓ $200 = x$;

✓ $x + 2 = 0$;

3) 대입 연산자

- `a = a + 1;`
 - "a와 a+1이 같다"는 뜻이 아니라
 - "변수 a에 a+1의 값을 저장하라($a \leftarrow a+1$)"는 뜻
- 동작 과정
 - ✓ 대입문 수행 전에 변수 a에 20이 저장되어 있었다면
 - ✓ `a = a+1` \rightarrow `a = (a에 저장되어 있던)20 + 1`

```
a = 20;  
a = a+1;  
printf("%d",a);
```



실행 결과

21

3) 대입 연산자

- 연속 대입

- 대입 연산의 결과는 왼쪽 변수에 저장되는 값
- 이를 이용하면 대입을 연속적으로 수행할 수 있음

- `a = b = c = 2;` 이 문장의 의미는

`c = 2;` //오른쪽부터 수행함에 주의

`b = c;`

`a = b;`

✓ `a = b+2 = c = d = 5` // 컴파일 에러

`b+2 = c (X)`

3) 대입 연산자

- 복합 대입 연산자: 대입 연산자와 산술 연산자의 결합
 - $a \text{ += } x \rightarrow a$ 의 값을 x 만큼 증가시킴

복합 대입 연산	동일 대입문
$a \text{ += } x$	$a = a + (x)$
$a \text{ -= } x$	$a = a - (x)$
$a \text{ *= } x$	$a = a * (x)$
$a \text{ /= } x$	$a = a / (x)$
$a \text{ %= } x$	$a = a \% (x)$

```
int a = 3;
int b = 2;
a += 5;    // a = a+5; 결과: 8
a /= b;    // a = a/b; 결과: 4
a %= 3;    // a = a%3; 결과: 1
```

- 다음 중 $a *= b+3$ 이 나타내는 수식은?
 - ✓ $a = a * (b+3);$
 - ✓ $a = a * b + 3;$

3) 대입 연산자

증감 연산자

- 변수의 값을 1씩 증가(++) 혹은 감소(--) 시키는 단항 연산자
- 변수의 앞에 오느냐 뒤에 오느냐에 따라 수식의 해석이 달라진다.

증감 연산	의미
++a	a의 값 1 증가 → a의 값 사용
--a	a의 값 1 감소 → a의 값 사용
a++	a의 값 사용 → a의 값 1 증가
a--	a의 값 사용 → a의 값 1 감소

```
a = 1;  
b = ++a;  
printf("a: %d\n", a);  
printf("b: %d\n", b);
```

```
a = 1;  
b = a++;  
printf("a: %d\n", a);  
printf("b: %d\n", b);
```

3) 대입 연산자

- 다음 각각에 해당하는 C언어 수식을 작성하시오

(각 문제는 독립적이고, 문제마다 하나의 수식으로 작성할 것)

1. 변수 sum에 1부터 5까지 더한 값을 대입
2. 변수 x의 값에 2를 곱한 후 변수 y를 더한 값을 변수 x에 대입
3. 변수 x에는 $x+2$ 의 값을, 변수 y에는 변수 x의 값을, 변수 z에는 변수 y의 값을 차례로 대입 (연속 대입 사용할 것)
4. 변수 x의 값을 3만큼 감소 (복합대입연산자 사용할 것)
5. 변수 x의 값을 3으로 나눈 나머지를 x에 대입 (복합대입연산자 사용할 것)
6. 변수 x의 값을 1만큼 감소 (증감연산자 사용할 것)
7. 변수 x의 값을 1만큼 감소시키고, 그 결과에 변수 y와 z의 합을 곱한 결과를 변수 z에 대입
8. 변수 x의 값을 y배 시키고, y는 1만큼 증가

목차

- 1) 수식과 연산자 개요
- 2) 산술 연산자
- 3) 대입 연산자
- 4) **관계 연산자**
- 5) 논리 연산자
- 6) 그 외 연산자
- 7) 연산자 우선순위와 결합 수칙

4) 관계 연산자

■ 관계 연산자

- 왼쪽과 오른쪽의 **대소 관계를 비교**하는 연산자
- 연산의 결과는 참 아니면 거짓으로, 참이면 **1**이고 거짓이면 **0**
✓ (참고) C언어에서는 0이 아닌 값은 모두 참으로 간주

관계연산	의미
$x == y$	x의 값과 y의 값이 같다
$x != y$	x와 y가 같지 않다
$x < y$	x가 y 보다 작다
$x <= y$	x가 y 보다 작거나 같다
$x > y$	x가 y 보다 크다
$x >= y$	x가 y 보다 크거나 같다

```
a = (4 < 5);  
printf("a: %d\n", a);
```

```
a = (4 == 5);  
printf("a: %d\n", a);
```

4) 관계 연산자

- 실습 예제

- 다음 소스 코드의 실행 결과를 예상해보고, 코드를 작성하여 확인해 보시오.

```
int a = 3;
printf("%d\n", a > 4);
printf("%d\n", a < 4);
printf("%d\n", a == 5);
printf("%d\n", a != 3);
printf("%d\n", 2 >= a);
printf("%d\n", a <= a+1);
```

4) 관계 연산자

- 실습 예제
 - 실행 결과

0
1
0
0
0
1

4) 관계 연산자

- 관계 연산자와 산술 연산자의 우선순위

- 관계 연산자의 우선순위는 산술 연산보다 낮음
- 산술 계산 먼저 → 계산된 값 비교

$$✓ a + b < c - d \Rightarrow (a + b) < (c - d)$$

$$✓ a <= 2 * n \Rightarrow a <= (2 * n)$$

$$✓ a \% 2 == 0 \Rightarrow (a \% 2) == 0$$

- ✓ 우선순위가 익숙하지 않으면 괄호를 사용하여 수식을 써도 되지만, 나중에는 괄호를 사용하지 않는 왼쪽 형태에도 익숙해져야 한다

4) 관계 연산자

- 다음 각각에 해당하는 C언어 수식을 작성하시오

(각 문제는 독립적이고, 문제마다 하나의 수식으로 작성할 것)

1. 변수 x 는 7과 같다.
2. 변수 x 와 y 의 합은 3보다 크다.
3. 변수 y 는 $x+1$ 보다 작거나 같다.
4. 변수 x 는 3과 같지 않다.
5. 변수 x 는 y 보다 크거나 같다
6. 12를 5로 나눈 나머지는 x 를 5로 나눈 나머지보다 작거나 같다.
7. 변수 x 에서 y 를 뺀 값은 음수이다.
8. x 를 1만큼 증가시키고, 그 값은 y 와 같지 않고, y 를 1만큼 감소시킨다.

4) 관계 연산자

- 수식 $4 < 5 < 2$ 의 결과는?
 - 수학적 의미
 - ✓ 5는 4보다 크고 2보다 작다 (거짓)
 - C 언어에서의 의미
 - ✓ $4 < 5 < 2 \rightarrow (4 < 5) < 2 \rightarrow 1 < 2 \rightarrow$ 수식 결과: 참(1)
 - 위 수식을 수학적 의미로 사용하지 않도록 주의
 - ✓ 문법적으로는 아무 문제 없어 정상적으로 컴파일
 - 수학적 $4 < 5 < 2$ 에 해당하는 C 언어 수식은?
 - ➔ $4 < 5 \ \&\& \ 5 < 2$ (논리 연산자 활용 : 다음 절에서 학습)

목차

- 1) 수식과 연산자 개요
- 2) 산술 연산자
- 3) 대입 연산자
- 4) 관계 연산자
- 5) 논리 연산자
- 6) 그 외 연산자
- 7) 연산자 우선순위와 결합 수칙

5) 논리 연산자

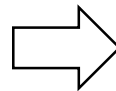
■ 논리 연산자

- 논리 연산 값으로 참이면 1이고 거짓이면 0
(0을 제외한 모든 값은 참으로 간주)

논리 연산	의미	연산 결과
! x	논리부정 (NOT)	x가 참이면 거짓, 거짓이면 참
x && y	논리곱 (AND)	x, y가 둘 다 참이면 참이고, 그렇지 않으면 거짓
x y	논리합 (OR)	x, y 중 하나라도 참이면 참이고, 그렇지 않으면 거짓

```
int x = 1, y = 0;
```

```
printf("%d\n", !x);  
printf("%d\n", x&& x);  
printf("%d\n", x&& y);  
printf("%d\n", x || y);  
printf("%d\n", y&& y);
```



실행 결과

```
0  
1  
0  
1  
0
```

5) 논리 연산자

- 실습

- 다음 소스 코드의 실행 결과를 예상해보고, 코드를 작성하여 확인해 보시오.

```
int a = 3;
int b = 5;

printf("%d\n", (a>=3)&&(b<6));
printf("%d\n", (a!=3)&&(a>2));
printf("%d\n", (b!=5)|| (a==1));
printf("%d\n", (a!=!b)|| (b==2));
```

5) 논리 연산자

- 실습 예제
 - 실행 결과

1
0
0
1

5) 논리 연산자

■ 논리 연산자의 연산 순서

- 우선순위: **!** > **&&** > **||** (논리부정이 가장 높음)
- 우선순위가 같은 경우, 왼쪽부터 계산

✓ $!x \ \&\& \ y$	⇒ $(!x) \ \&\& \ y$
✓ $x \ \&\& \ y \ \&\& \ z$	⇒ $(x \ \&\& \ y) \ \&\& \ z$
✓ $x \ \ y \ \&\& \ z$	⇒ $x \ \ (y \ \&\& \ z)$
✓ $x \ \ y \ \&\& \ z \ \ w$	⇒ $(x \ \ (y \ \&\& \ z)) \ \ w$
✓ $x \ \&\& \ y \ \ z \ \&\& \ w$	⇒ $(x \ \&\& \ y) \ \ (z \ \&\& \ w)$

• 우선 순위를 무시하려면 괄호 사용

- ✓ $!(x \ \&\& \ y)$ ⇨ 수식 $!x \ \&\& \ y$ 와 의미가 다름에 주의
- ✓ $x \ \&\& \ !(y \ \&\& \ z)$

5) 논리 연산자

- 논리 연산자와 관계/산술연산자의 우선순위
 - **&&** 와 **||** 의 우선순위는 관계/산술 연산자보다 낮음
 - But, **!** 의 우선순위는 관계/산술 연산자보다 높음

✓ $a \geq 3 \text{ \&\& } b < 6$ $\Rightarrow (a \geq 3) \text{ \&\& } (b < 6)$

✓ $a \neq b \text{ || } b == 2$ $\Rightarrow (a \neq b) \text{ || } (b == 2)$

✓ $!(a \neq b) \text{ || } b == 2$ $\Rightarrow (!(a \neq b)) \text{ || } (b == 2)$

5) 논리 연산자

- [예제 4.2]
 - 다음 논리 연산을 계산해보고 프로그램을 작성하여 확인해보자.

```
int a = 3, b = 5;

printf("%d\n", a>=3 && b<6 );
printf("%d\n", a!=3 && a>2  );
printf("%d\n", b!=5 || a==1 );
printf("%d\n", a!=b || b==2 );
printf("%d\n", !(a!=b) || b==2 );
```


5) 논리 연산자

- 다음 각각에 해당하는 C언어 수식을 작성하시오

(각 문제는 독립적이고, 문제마다 하나의 수식으로 작성할 것)

1. x 가 참이거나 y 가 참이다.
2. x 는 참이 아니다. (즉, x 는 거짓이다.)
3. x 가 3보다 크거나 y 가 4보다 작다.
4. y 는 3이 아니면서 x 보다는 크거나 같다.
5. $2 < x < 9$
6. x 는 0보다 크고, 2로 나누어 떨어지고, 5로는 나누어 떨어지지 않는다. (2로 나누어 떨어진다는 \equiv 2로 나눈 나머지는 0이다)

목차

- 1) 수식과 연산자 개요
- 2) 산술 연산자
- 3) 대입 연산자
- 4) 관계 연산자
- 5) 논리 연산자
- 6) 그 외 연산자**
- 7) 연산자 우선순위와 결합 수칙

6) 그 외 연산자

■ 조건 연산자

- if ~ else 문(5장에서 학습)을 대신하여 사용할 수 있는 연산자
- 피연산자 수가 3개인 삼항 연산자

조건 ? A : B	조건이 true인 경우, 결과 값은 A
	조건이 false인 경우 결과 값은 B

- ✓ $a < 0 ? -a : a$ ⇨ a가 0보다 작으면 연산 결과는 -a, 그렇지 않으면 a (절댓값 계산)
- ✓ $a < b ? a : b$ ⇨ a가 b보다 작으면 연산 결과는 a, 그렇지 않으면 b (둘 중 작은 값 계산)

5) 논리 연산자

▪ [예제 4.3]

- 조건 연산자를 이용하여 부호를 판별하고, 절댓값을 구하는 프로그램을 작성해보자.

```
int a = -10, b;  
  
b = a < 0 ? -1 : 1;      // 부호 판별  
printf("%d\n", b);  
  
b = a < 0 ? -a : a;      // 절댓값 구하기  
printf("%d\n", b);
```

- (참고) 다음과 같은 형태도 가능하나 추천하지 않음
✓ `a < 0 ? (b = -1) : (b = 1);` // 괄호 필요

6) 조건 연산자

- 다음 각각에 해당하는 C언어 수식을 작성하시오
(각 문제는 독립적이고, 문제마다 하나의 수식으로 작성할 것)
 1. x 가 y 보다 크면 z 에 3대입하고, 작거나 같으면 z 에 2대입.
 2. x 와 y 중 큰 값을 z 에 대입
 3. x 와 y 가 같지 않고 y 가 7보다 작으면, x 값 1 증가, 그렇지 않으면 y 값 1 감소

6) 그 외 연산자

- **coma 연산자:** 여러 수식을 하나의 문장으로 표현할 때 사용

a = b+3;

b = 2; ⇨ a = b+3, b = 2, b += a; (왼쪽 수식부터 계산)

b += a;

- **sizeof 연산자:** 저장 공간의 크기를 바이트 단위로 계산

sizeof(char) ⇨ 결과 값 1 (괄호 필수)

sizeof(3.14) 또는 sizeof 3.14 ⇨ 결과 값 8

sizeof(num) 또는 sizeof num ⇨ 결과 값 4 (num이 int 형 변수 일 때)

※ 시스템에 따라 결과 값은 다를 수 있음

- **형변환 연산자:** 명시적 형변환을 위해 사용

✓ x = (double) a/b; ⇨ a의 자료형을 부동 소수로 변환

✓ x = (double) (a/b); ⇨ a/b의 결과 값을 부동 소수로 변환

목차

- 1) 수식과 연산자 개요
- 2) 산술 연산자
- 3) 대입 연산자
- 4) 관계 연산자
- 5) 논리 연산자
- 6) 그 외 연산자
- 7) 연산자 우선순위와 결합 수칙

7) 연산자의 우선순위와 결합수칙

- 연산자 우선순위

- 여러 연산자가 함께 사용된 경우 우선순위에 의존.
- 다른 순서로 연산을 하고 싶은 경우 괄호를 사용.

- 결합 수칙

- 연산의 순서를 나타냄.
- 연산자 우선순위가 같은 경우 결합수칙에 의존.
- 예) $5 / 2 * 4$ 의 결과는?
 - ✓ / 와 * 의 우선순위는 동일
 - ✓ / 와 * 의 결합수칙은 '왼쪽우선'

7) 연산자의 우선순위와 결합수칙

순위	연산자	종류	결합수칙
1	() 함수호출 [] . ->	멤버	왼쪽 우선
2	+ 부호 - 부호 ++ -- ! ~ * 간접참조 & 주소 sizeof (자료형)	단항	오른쪽 우선
3	* 곱하기 % /	산술	왼쪽 우선
4	+ 더하기 - 빼기		
5	<< >>	비트 (이동)	
6	< > <= >=	관계	
7	== !=		
8	& 비트곱	비트 (논리)	
9	^		
10			
11	&&	논리	
12			
13	?:	조건	오른쪽 우선
14	= += -= *= %= /= ^= <<= >>=	대입	
15	,	coma	왼쪽 우선

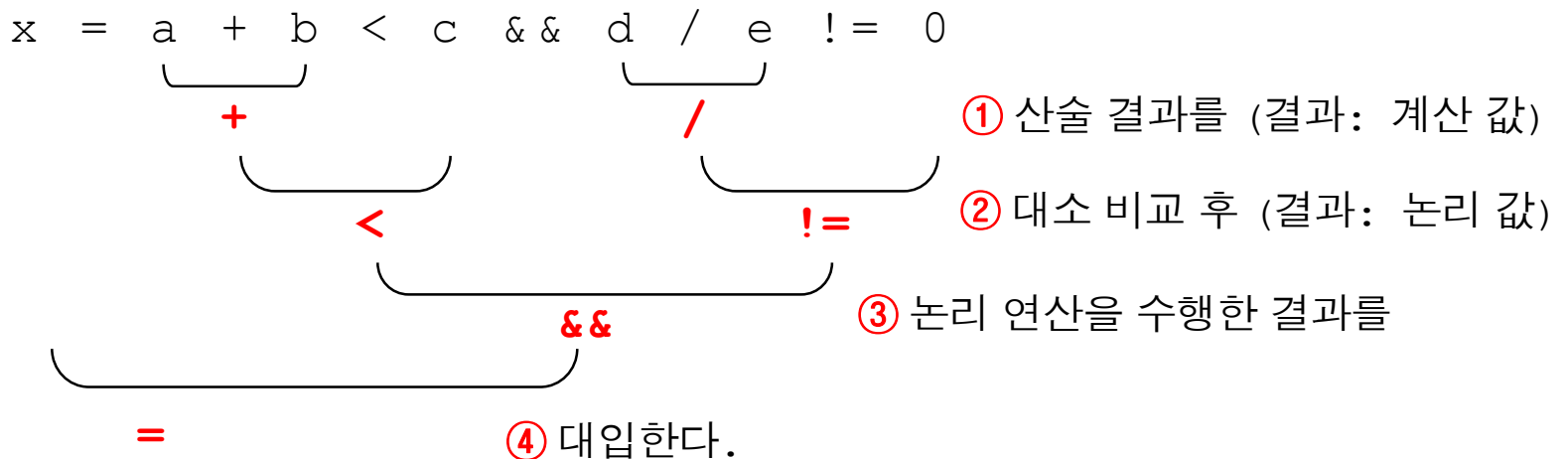
7) 연산자의 우선순위와 결합수칙

■ 주요 연산자 우선순위

• 기억하기 쉬운 연산자

- ✓ 멤버와 관련된 연산자(추후 학습)는 최우선 순위이다.
- ✓ **단항 연산자**는 이항 연산자보다 우선순위가 **높다**. (멤버 관련 연산자 제외)
- ✓ **콤마 연산자**는 우선순위가 가장 **낮다**.

• 산술 → 관계 → 논리 → 대입 연산 순 (각 연산의 의미 고려)



7) 연산자의 우선순위와 결합수칙

- 다음 각각에 해당하는 C언어 수식을 작성하시오

(각 문제는 독립적이고, 문제마다 하나의 수식으로 작성하고, 불필요한 괄호는 사용하지 마시오)

1. x 의 값에 2를 더한 결과 값에 y 를 곱한 값을 z 에 대입
2. x 와 y 를 각각 2로 나눈 나머지의 합을 z 에 대입
3. x 를 3으로 나눈 나머지를 y 에 대입하고 x 를 1증가 시킨다.
4. x 는 0보다 작거나, $5 < x < 9$ 이다.
5. x 는 양수이면서, 2로 나누어 떨어지거나 5로 나누어 떨어진다.
(즉, x 에 해당하는 수는 2, 4, 5, 6, 8, 10, ...)
6. 5번의 명제는 거짓이다.