연산자

1. 강의

▼ 연산자 : 연산에 사용 되는 기호

- 수식 : 값을 갖는 요소, 값이 있다면 수식으로 보자
- 연산식 : 연산자가 있는 수식

```
int num = 10; 수식
int num1 = num; 수식
int num2 = num + 123; 연산식
```

- 피연산자 : 연산의 대상이 되는 값
- 단항 연산자 : 피연산자 개수가 1, ++, —, !, &, sizeof
- 삼항 연산자 : 피연산자 개수가 3, <조건> ? 참일 때 대입 값 : 거짓일 때 대입 값
- 이항 연산자 : 대부분 연산자로 피연산자 개수가 2

▼ 산술연산자

- + - * / % → 코딩해보기
- / , % 연산자 쓰임을 연습한다. → 지폐 매수 구하기

▼ 증감연산자

- 자동으로 1씩 증가 혹은 감소 시키는 연산자
- 전위형 : 변수 이름 전에 ++ 혹은 - - 를 붙여 연산
- 후위형 : 변수 이름 뒤에 ++ 혹은 - - 를 붙여 연산
- 실수에도 사용 가능

```
count = 4;
result = ++count; // result = 5, count = 5
result = count--; // result = 5, count = 4
```

▼ 관계연산자

예시 : a > b, a > = b, a < b, a < = b, a ==b, a ! = b

num1 = 10, num2 = 20 인 경우에 대해 ==, ! = 계산 결과를 출력해 보자 (%d) 사용

▼ 논리연산자

논리연산자

Aa a	b	a && b	a || b	!a
0	0	0	0	1
0	1	0	1	1
1	0	0	1	0
1	1	1	1	0

- 논리 연산자와 관계 연산자의 연산 순서

```
score > 90 && score <= 100

연산 순서
(1) score > 90
(2) score <= 100
(3) &&
```

- 단축 계산 : 특정 수식에 대한 평가를 생략

▼ 비트연산자

- 비트 단위로 연산을 수행
- 비트 AND(&), 비트 OR(|), 비트 XOR(^), 비트 NOT (~)
- 비트 왼쪽 이동 (<<), 비트 오른쪽 이동(>>)
- char x = 5, y = 6 일때 비트 연산

x = 5	0	0	0	0	0	1	0	1
y = 6	0	0	0	0	0	1	1	0
x & y	0	0	0	0	0	1	0	0
x | y	0	0	0	0	0	1	1	1
x ^ y	0	0	0	0	0	0	1	1
~ x	1	1	1	1	1	0	1	0
x << 1	0	0	0	0	1	0	1	0
x >> 1	0	0	0	0	0	0	1	0

- 비트 왼쪽 이동 연산에서 빈자리는 0으로 채워 진다.
- 비트 왼쪽 이동 연산의 결과 =원래 십진수에 2^N을 곱한것과 같다
- 비트 오른쪽 이동 연산에서 빈자리는 양수일 때 0, 음수 일때 1로 채워진다.
- 비트 오른쪽 이동 연산의 결과 = 원래 십진수를 2^N으로 나눈 것과 같다

▼ 대입 연산자 (=)

- 좌변의 값을 우변의 변수에 대입한다.
- 산술 비트 연산자와 함께 쓰여 복합 연산자를 만든다.

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 ^= y	x = x ^ y
x >>= y	x = x >> y
x <<= y	x = x << y

- 복합 대입 연산자는 우선 순위가 낮기 때문에 수식표현에 주의한다.
- a = a * 2 + 3; ↔ a *= (2+3)

▼ 조건 연산자

- 수식1 ? 수식2 : 수식3
- 수식 1 이 참이면 수식2가, 거짓이면 수식 3이 연산 결과가 된다.

▼ 형변환 연산자

- 묵시적 형변환 : 자동으로 처리되는 형 변환
- 정수의 연산은 4 byte로 변환 처리 (int * int 를 short 에 저장하면 오버플로우가 될 수 있다.)
- 실수 연산은 8 byte로 변환 처리
- 명시적 형변환 : 형변환 연산자 사용

```
형식 : (data 형) 수식

(int) 3.14
(double) num
(float) sum
```

▼ 연산자의 우선순위와 결합방향

- 일반적 우선 순위 : 단항 > 산술 > 관계 > 논리 > 대입 > 콤마
- 우선순위와 결합 방향

우선순위	연산자	결합 방향
1	( ) [ ] -> .	→
2	++ -- +(부호) -(부호) sizeof ~ ! * & (type)	←
3	* / %	→
4	+ -	→
5	<< >>	→
6	< <= > >=	→
7	== !=	→
8	&	→
9	^	→
10	|	→
11	&&	→
12	||	→
13	?:	→
14	= += -= *= /= %= &= |= ^= <<= >>=	←
15	,(콤마)	→

2. 실습
    - 지폐 매수 계산 프로그램 작성해 보기