제 6 강 : 연산자

※ 학습목표

✓ 진수 변환을 수행할 수 있다.

✓ 연산자를 활용할 수 있다.

✓ 연산자 우선순위를 설명할 수 있다.

1. 연산자란 무엇인가?

✓ 연산자란 자료의 가공을 위해 정해진 방식에 따라 계산하고 결과를 얻기 위한 행위를 의미하는 기호들의 총칭이다.

✓ 그리고 각 연산자들은 연산을 하기 위해 인식하는 자료형 들이 정해져 있다

✓ 연산자 우선순위

종 류	연산자	우선순위
최우선 연산자	. , [], ()	1
단항 연산자	!, ~, +/-, ++/--, (cast)	2
산술 연산자	*, /, %, +, -	3
시프트 연산자	<<, >>, >>>	4
관계 연산자	>, >=, <, <=, ==, !=	5
비트 연산자	&, ¦, ^	6
논리 연산자	&&, ¦¦	7
조건(삼항) 연산자	조건 ? 항1 : 항2	8
배정 대입 연산자	*=, /=, %=, +=, -=, <<=, >>=, >>>=, &=, ¦=, ^=, =	9
후위형 증감 연산자	++/--	10
순차 연산자	,	11

2. 최우선 연산자

① . period 연산자 : 접근연산자

✓ 특정 범위 내에 속해 있는 멤버를 지칭할 때 사용함

ex) System.out.println(“test“);

② [] 대괄호 연산자

✓ 배열 참조 연산자

✓ 자료형이나 클래스와 함께 사용되어 해당 변수나 객체가 배열로 선언됨을 알리는 역할

ex) String[] arr = { “AA” , “BB” , “CC” };

③ () 괄호 연산자

✓ 특정 연산자들을 묶어서 먼저 처리할 수 있도록 만들어주는 연산자

✓ ex) int x = 5 * (3 + 2);

3. 단항 연산자

① ! 논리부정 연산자

✓ 논리 자료형의 데이터 값을 부정하는 연산자

✓ ex) boolean bool = false; 또는 boolean bool2 = !bool;

② ~ 비트 부정 연산자

✓ 비트 값으로 존재하는 모든 자료들에 대해 부정의 값을 취할 수 있는 연산자

✓ 단, boolean, float, double형은 ~ 연산자를 사용할 수 없음

✓ byte, short, char, int형은 ‘~’ 연산 결과후 int, long형에만 담을 수 있음

ex) byte b = 120; int i = ~b;

✓ long형은 ‘~’ 연산 후 long형에만 담을 수 있음.

ex) long l = 120L; long l = ~l;

③ +/- 양수, 음수 판별 연산자

✓ 양수, 음수 판별해주는 연산자(+ 생략 가능).

ex) int i = -120;

④ ++/-- 전위형 증감 연산자

✓ 특정 변수의 값을 하나 증가시키거나 하나 감소시키는 연산자

ex) int a = 4; int b = ++a;

✓ 후위 연산자와 우선순위의 차이가 존재함

[실습] day03 프로젝트를 새로 만들고 작업할 것.

```
1  package tommy.java.exam01;
2
3  public class OperEx1 {
4          public static void main(String[] ar) {
5                  int x = 10;
6                  int y = ++x;
7                  System.out.println("x = " + x);
8                  System.out.println("y = " + y);
9          }
10 }
```

4. 산술연산자

✓ +, -, *, / , % 연산자

✓ byte, short, char, int 자료형 사이의 연산에서는 결과가 int 임

✓ long, float, double 자료형이 연산되면 큰 자료형으로 결과가 결정 됨

✓ '/' 몫, '%' 나머지 값

[실습]

```
1  package tommy.java.exam02;
2
3  public class OperEx2 {
4          public static void main(String[] ar) {
5                  short a, b;
6                  a = b = 10;
7                  short c = a + b;
8                  System.out.println("c의 값 :" + c);
9          }
10 }
```

5. 시프트 연산자

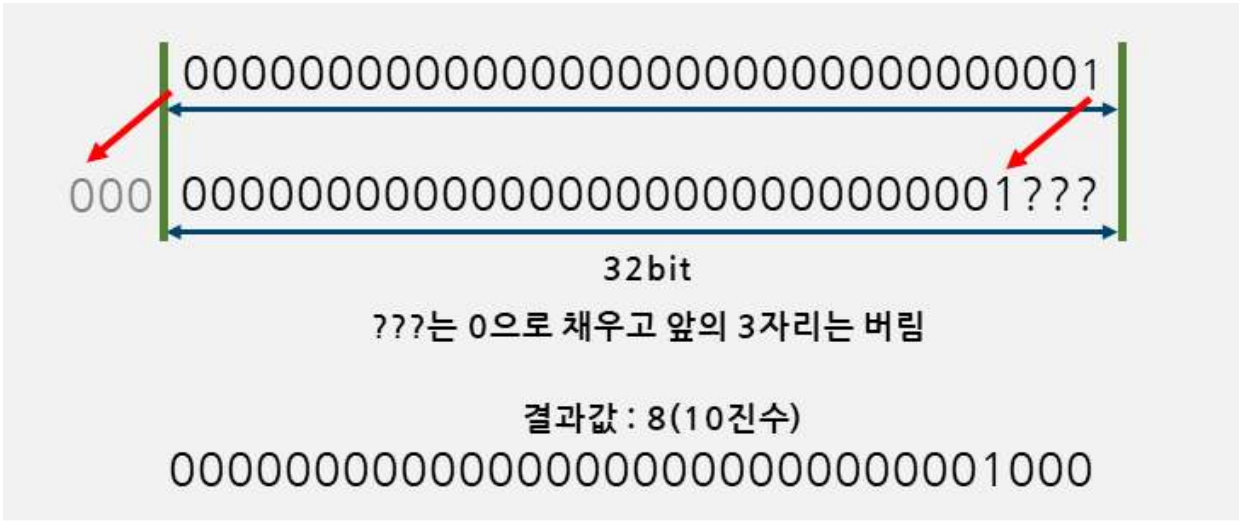
✓ 대상 변수의 값을 2진 비트로 바꾼 후 특정 비트 수만큼 이동시켜 원하는 부분의 비트 데이터를 얻어 내는 연산자

① << left shift 연산자

✓ 대상 변수 값을 2진 비트로 바꾼 후 왼쪽으로 특정 비트 수만큼 이동

✓ 빈자리는 0값으로 채움

ex) 1 << 3

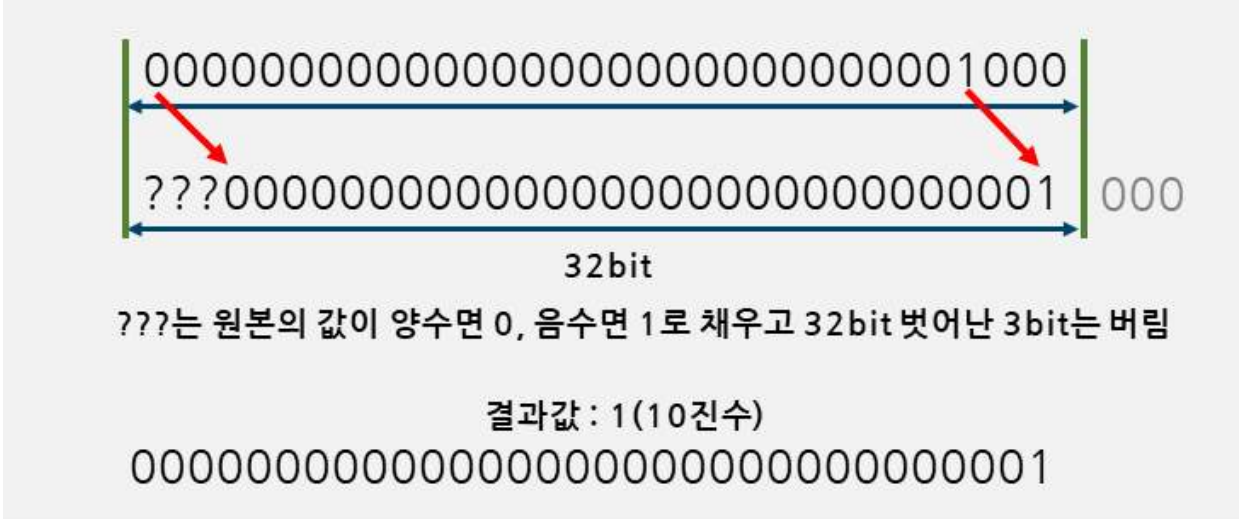


② >> right shift 연산자

✓ 대상 변수 값을 2진 비트로 바꾼 후 왼쪽으로 특정 비트 수만큼 이동

✓ 빈자리는 0값으로 채움

ex) 8 >> 3



③ >>> unsigned right shift 연산자

✓ '>>' 와 기본적으로 같음

✓ 그러나 원본데이터가 음수일 경우에도 빈 비트를 0으로 채움

ex) -3 >> 3 = -1

-3 >>> 3 = 536870911

[실습]

```
1  package tommy.java.exam03;
2
3  public class OperEx3 {
4          public static void main(String[] ar) {
5                  int i = -10;
6                  int j = i >>> 2;
7                  System.out.println("i = " + i + ", j = " + j);
8          }
9  }
```

6. 관계연산자

① >, <, >=, <= 비교 관계 연산자

② ==, != 항등 관계 연산자

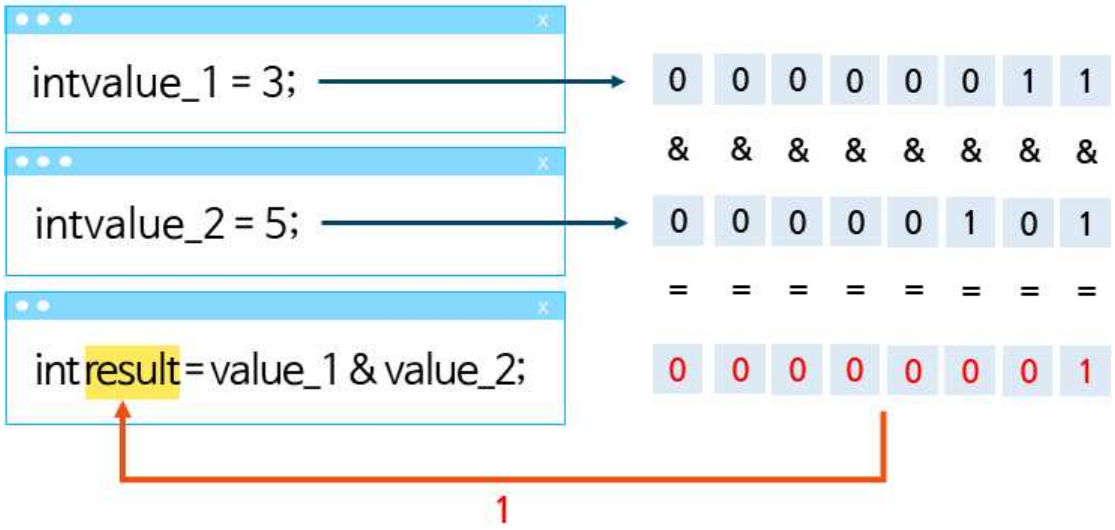
[실습]

```
1  package tommy.java.exam04;
2
3  public class OperEx4 {
4          public static void main(String[] ar) {
5                  int a = 10;
6                  int b = 5;
7                  boolean c = a < b;
8                  System.out.println("a < b :" + c);
9                  c = a != b;
10                 System.out.println("a != b : " + c);
11         }
12 }
```

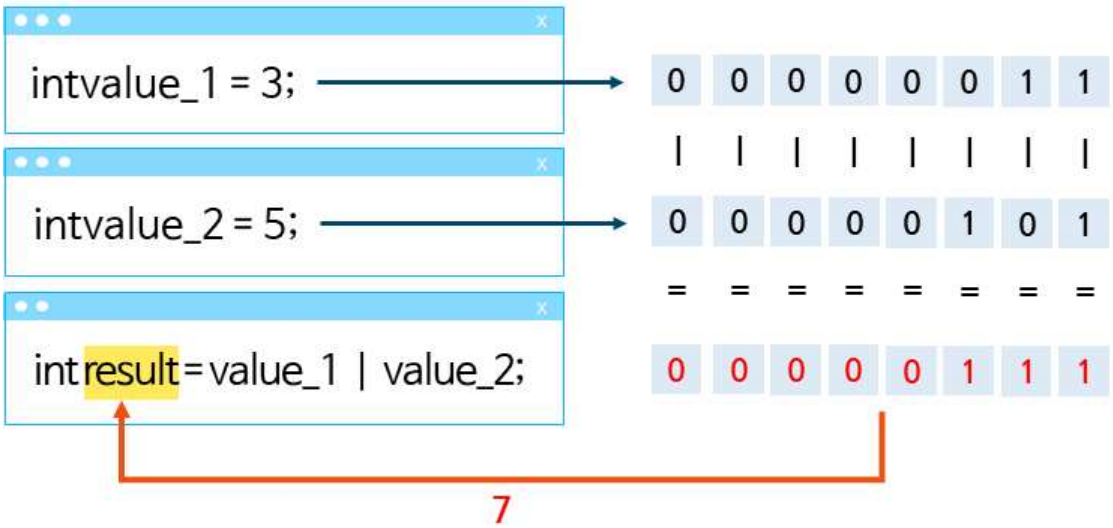
7. 비트 연산자

값1	값2	&(AND) 연산자	|(OR) 연산자	^(Exclusive OR) 연산자
0	0	0	0	0
1	0	0	1	1
0	1	0	1	1
1	1	1	1	0

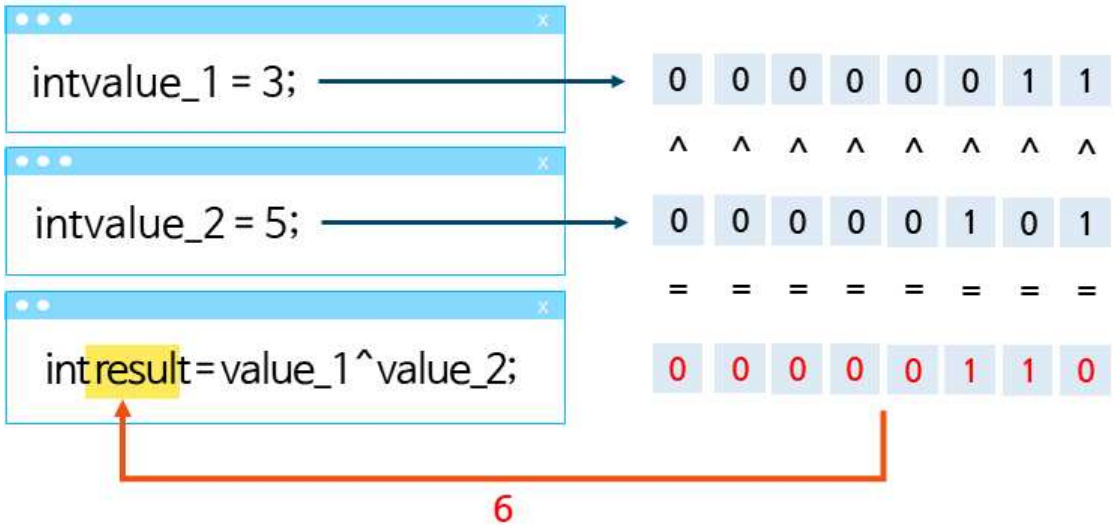
① 비트 AND 연산자



② 비트 OR 연산자



③ 비트 XOR 연산자



8. 논리연산자

✓ && (AND) ¦¦ (OR) ☞ 앞의 것만으로도 판단가능이면 판단함

✓ short circuit : &&일 경우 false출력 ¦¦일 경우 true출력 고로 실행속도가 빠르다.
& ¦ 사용 단 실행 후 결과 출력 실행속도가 느리다.

[실습]

```
1  package tommy.java.exam05;
2
3  public class OperEx5 {
4          public static void main(String[] ar) {
5                  boolean a;
6                  boolean b;
7                  if ((a = 4 > 3) ¦¦ (b = 5 > 7)) {
8                          System.out.println("a = " + a);
9                          System.out.println("b = " + b);
10                 }
11         }
12 }
```

9. 삼항연산자

✓ 조건항 ? 항1 (true일 때) : 항2 (false일 때)

[실습]

```
1  package tommy.java.exam06;
2
3  public class OperEx6 {
4          public static void main(String[] args) {
5                  int a = 20, b = 30, max;
6                  max = a > b ? ++a : ++b;
7                  System.out.println("max : " + max);
8                  System.out.println("a : " + a);
9                  System.out.println("b : " + b);
10         }
11 }
```

10. 배정연산자, 대입연산자

✓ *=, /=, %=, +=, -=, <<=, >>=, >>>= ^=, &=, !=, =

[실습]

```
1  package tommy.java.exam07;
2
3  public class OperEx7 {
4          public static void main(String[] ar) {
5                  int a = 10;
6                  int res = 0;
7                  res += a;
8                  System.out.println("res = " + res);
9                  res *= a;
10                 System.out.println("res = " + res);
11                 res -= a;
12                 System.out.println("res = " + res);
13                 res %= a;
14                 System.out.println("res = " + res);
15         }
16 }
```

11. 후위형 증감연산자 : ++/--

12. 콤마 연산자 : ,

[실습] 정수와 실수의 특수한 연산

```
1  package tommy.java.exam08;
2
3  public class OperEx8 {
4          public static void main(String[] ar) {
5                  int i = 10;
6                  int j = 0;
7                  System.out.println("i / j = " + (i / j));
8          }
9  }
```