

# 컴퓨터 프로그래밍 (Computer Programming)

이 선 순



## 5. 연산자



# 목차

1. 연산자
2. 산술연산자
3. 관계연산자
4. 논리연산자
5. 비트연산자
6. 연산자 우선순위

**05**

# 비트연산자

## 05 비트연산자

### ■ 비트 연산자

- 정수나 문자 등을 2진수로 변환한 다음 각 자리의 비트끼리 연산을 수행

표 4-6 비트 연산자의 종류

비트 연산자	설명	의미
&	비트 논리곱 연산자(AND)	둘 다 1이면 1이다.
	비트 논리합 연산자(OR)	둘 중 하나만 1이면 1이다.
^	비트 배타적 논리합 연산자(XOR)	둘이 같으면 0이고, 둘이 다르면 1이다.
~	비트 부정 연산자	1은 0으로 바꾸고, 0은 1로 바꾼다.
<<	왼쪽 시프트 연산자	비트를 왼쪽으로 시프트한다.
>>	오른쪽 시프트 연산자	비트를 오른쪽으로 시프트한다.

## 05 비트연산자

### ■ 비트 논리곱 연산자 &

- &(AND) 연산자
- 두 비트가 모두 1인 경우만 1 아니면 0

```
int num1 = 5;  
int num2 = 10;  
int result = num1 & num2;
```



```
num1 : 00000101  
&num2 : 00001010  


---

result: 00000000
```

## 05 비트연산자

### ■ 비트 논리곱 연산자 &

#### ■ '10 & 7'

10진수를 2진수로 변환한 다음 각 비트마다 AND 연산을 수행

2진수로는 00102, 10진수로는 2

A	B	A&B
0	0	0
0	1	0
1	0	0
1	1	1

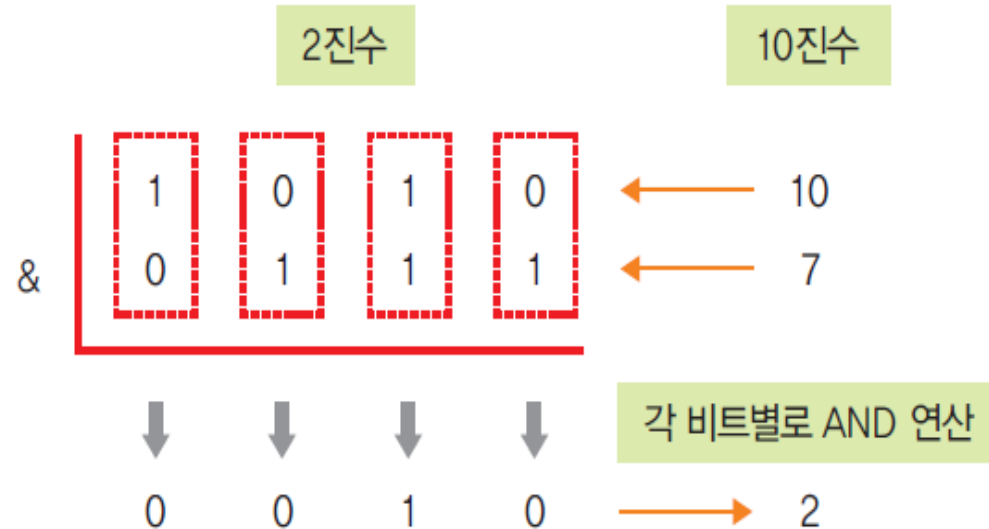


그림 4-13 비트 논리곱의 예

## 05 비트연산자

### 실습 4-8 비트 논리곱 연산자 사용 예

```
01 public class Ex04_08 {  
02     public static void main(String[] args) {  
03         System.out.printf(" 10 & 7 = %d \n", 10 & 7);  
04         System.out.printf(" 123 & 456 = %d \n", 123 & 456);  
05         System.out.printf(" 0xFFFF & 0000 = %d \n ", 0xFFFF & 0000);  
06     }  
07 }
```

----- 10과 7의 비트 논리곱을 수행한다.  
----- 123과 456의 비트 논리곱을 수행한다.  
----- 16진수 FFFF와 0의 비트 논리곱을 수행한다.

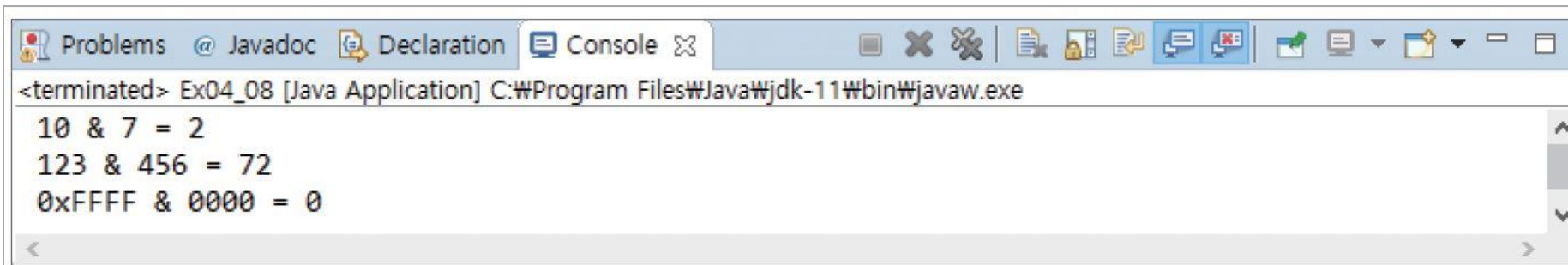


그림 4-14 실행 결과



## 05 비트연산자

### ■ 비트 논리합 연산자 |

- | (OR) 연산자
- 두 비트가 모두 0 인 경우만 0 아니면 1

```
int num1 = 5;  
int num2 = 10;  
int result = num1 | num2;
```



```
num1 : 00000101  
| num2 : 00001010  
-----  
result: 00001111
```

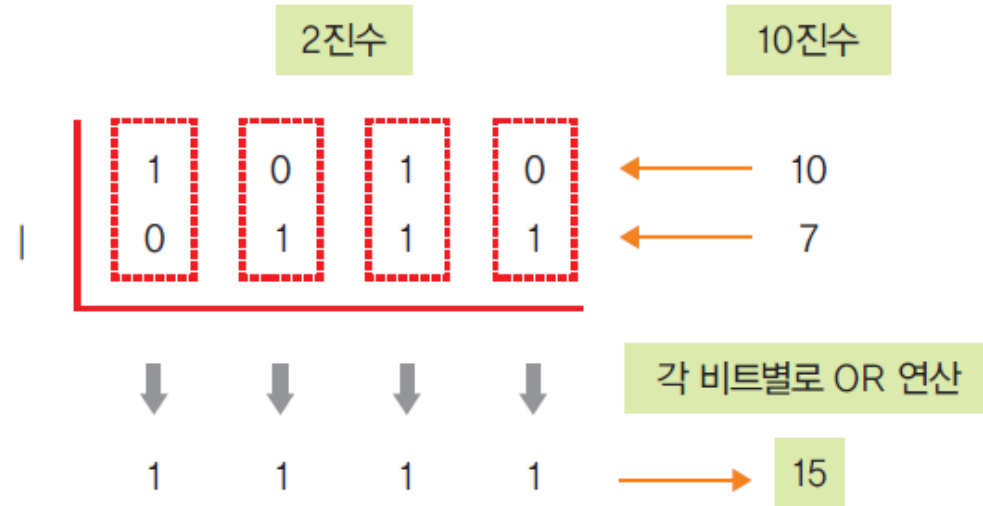
## 05 비트연산자

### ■ 비트 논리합 연산자 |

- '10 | 7'

A	B	A B
0	0	0
0	1	1
1	0	1
1	1	1

그림 4-15 비트 논리합의 예

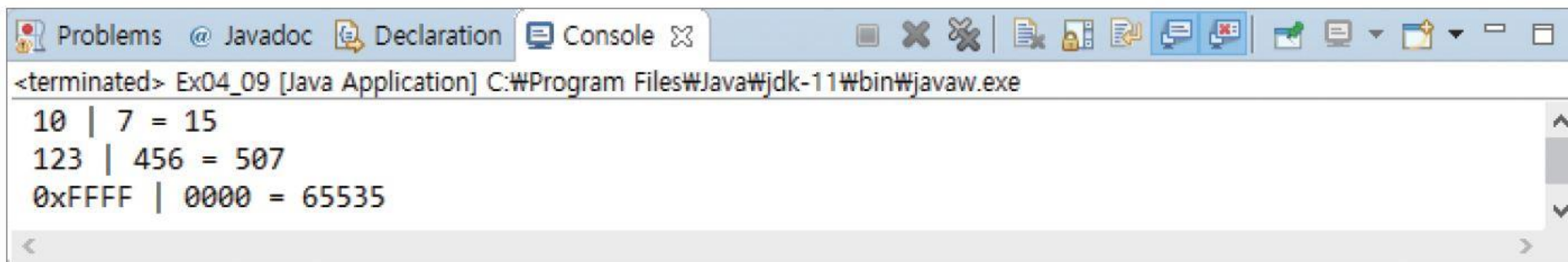


- 비트 논리합의 결과는  $1111_2$ 이고, 이는 10진수로 15

## 05 비트연산자

### 실습 4-9 비트 논리합 연산자 사용 예

```
01 public class Ex04_09 {
02     public static void main(String[] args) {
03         System.out.printf(" 10 | 7 = %d \n", 10 | 7); ----- 10과 7의 비트 논리합을 수행한다.
04         System.out.printf(" 123 | 456 = %d \n", 123 | 456); ----- 123과 456의 비트 논리합을
                                                                수행한다.
05         System.out.printf(" 0xFFFF | 0000 = %d \n ", 0xFFFF | 0000); -----
06     }                                                                16진수 FFFF와 0의 비트 논리합을 수행한다.
07 }
```



```
<terminated> Ex04_09 [Java Application] C:\Program Files\Java\jdk-11\bin\javaw.exe
10 | 7 = 15
123 | 456 = 507
0xFFFF | 0000 = 65535
```

그림 4-16 실행 결과

## 05 비트연산자

### ■ 비트 배타적 논리합 연산자 ^

- ^ (XOR) 연산자
- 두 값이 다르면 1, 같으면 0이 됨.
- 즉  $1^1$ 이나  $0^0$ 이면 결과가 거짓(0)이고,  $1^0$ 이나  $0^1$ 이면 결과가 참(1)

```
int num1 = 5;  
int num2 = 10;  
int result = num1 ^ num2;
```



```
num1 : 00000101  
^ num2 : 00001010  
-----  
result : 00001111
```

## 05 비트연산자

### ■ 비트 배타적 논리합 연산자 ^

- 두 값이 다르면 1, 같으면 0이 됨. 즉  $1^1$ 이나  $0^0$ 이면 결과가 거짓(0)이고,  $1^0$ 이나  $0^1$ 이면 결과가 참(1)
- $10^7$

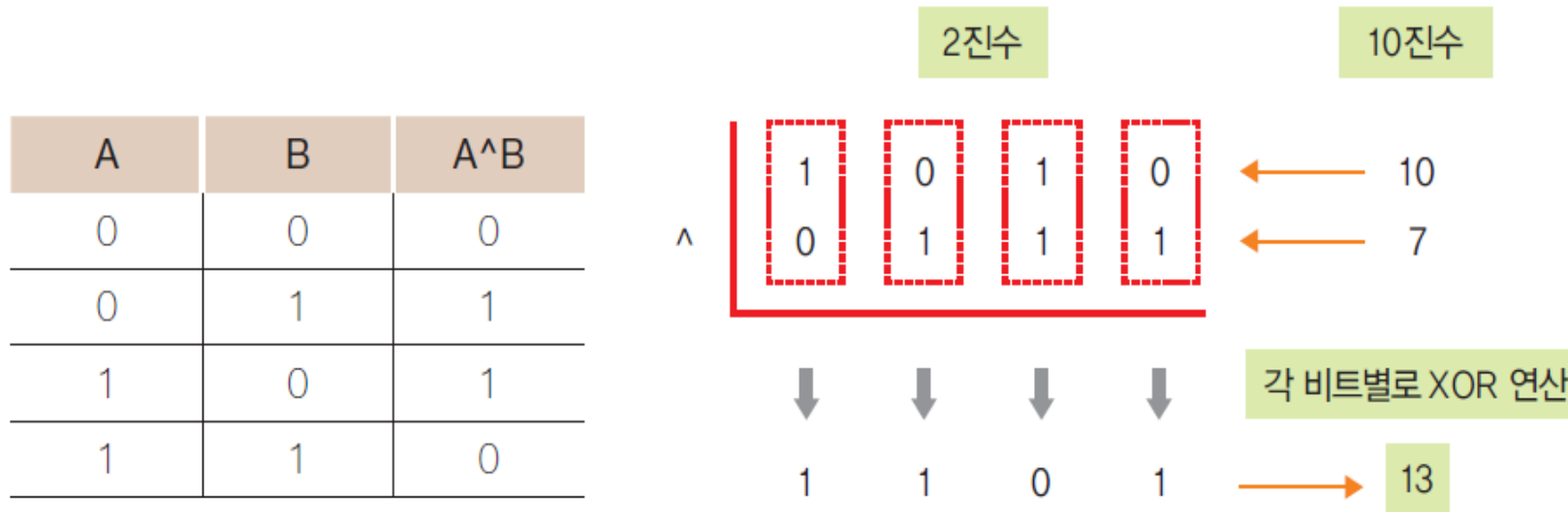


그림 4-17 비트 배타적 논리합의 예

- 비트 배타적 논리합 결과는  $1101_2$ 이고, 이는 10진수로 13

## 05 비트연산자

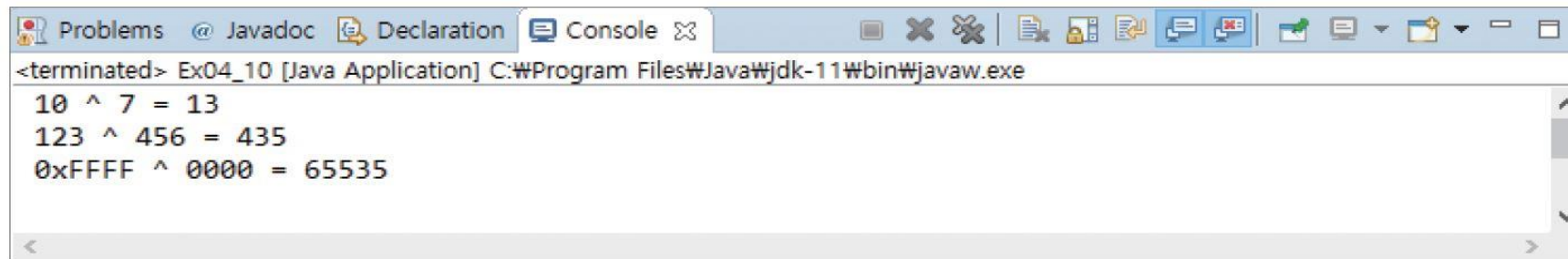
### 실습 4-10 비트 배타적 논리합 연산자 사용 예

```
01 public class Ex04_10 {
02     public static void main(String[] args) {
03         System.out.printf(" 10 ^ 7 = %d \n", 10 ^ 7);
04         System.out.printf(" 123 ^ 456 = %d \n", 123 ^ 456);
05         System.out.printf(" 0xFFFF ^ 0000 = %d \n ", 0xFFFF ^ 0000);
06     }
07 }
```

----- 10과 7의 비트 배타적 논리합을 수행한다.

----- 123과 456의 비트 배타적 논리합을 수행한다.

----- 16진수 FFFF와 0의 비트 배타적 논리합을 수행한다.



```
<terminated> Ex04_10 [Java Application] C:\Program Files\Java\jdk-11\bin\javaw.exe
10 ^ 7 = 13
123 ^ 456 = 435
0xFFFF ^ 0000 = 65535
```

그림 4-18 실행 결과

### ■ 비트 부정 연산자 ~

- 각 비트를 반대로 만드는 연산자.
- 즉 0은 1로 바꾸고, 1은 0으로 바꿈.
- 이렇게 반전된 값을 1의 보수라 하며, 그 값에 1을 더한 값을 2의 보수라 하는데 2진수에서 2의 보수는 음수를 나타냄
- 비트 부정 연산자는 해당 값의 음수(-)값을 찾고자 할 때 사용.
- 예. 정수값에 비트 부정을 수행한 다음 1을 더하면 그 값의 음수값을 얻을 수 있음

## 05 비트연산자

### ■ 비트 부정 연산자 ~

#### 실습 4-12 비트 부정 연산자 사용 예

```
01 public class Ex04_12 {  
02     public static void main(String[] args) {  
03         int a = 12345;  
04  
05         System.out.printf(" %d \n", ~a + 1); ----- a 값의 2의 보수를 구한다.  
06     }  
07 }
```

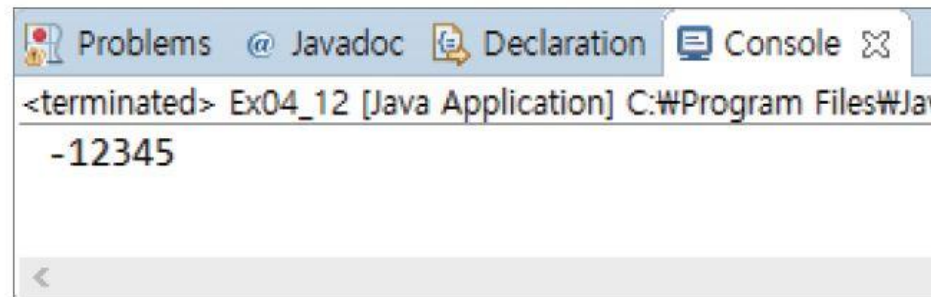


그림 4-22 실행 결과



## 05 비트연산자

### ■ 왼쪽 시프트 연산자 <<

- 나열된 비트를 왼쪽으로 시프트(shift), 이동연산자
- 26을 왼쪽으로 두 칸 시프트 연산
- 왼쪽으로 1회 시프트  $2^1$ , 2회 시프트  $2^2$ , 3회 시프트  $2^3$ 을 곱한 것과 같음

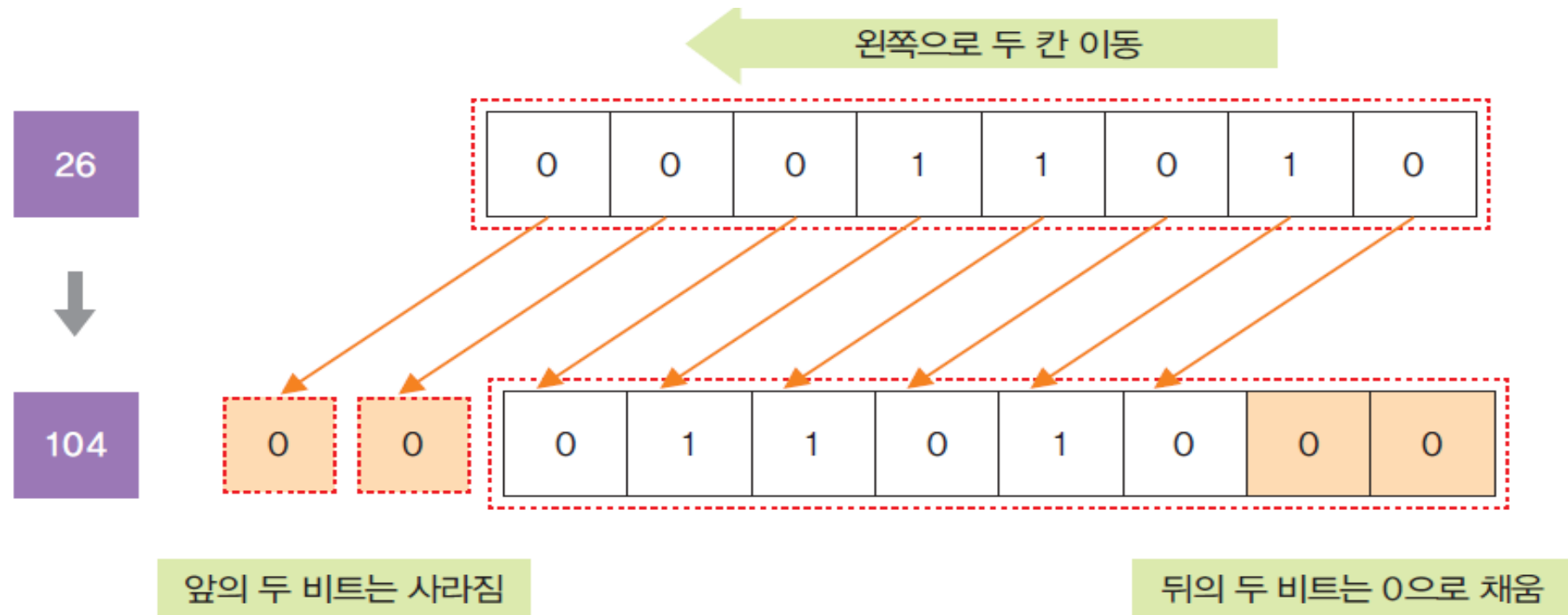


그림 4-23 26을 왼쪽으로 두 칸 시프트 연산

## 05 비트연산자

### 실습 4-13 왼쪽 시프트 연산자 사용 예

```
01 public class Ex04_13 {  
02     public static void main(String[] args) {  
03         int a = 10;  
04         System.out.printf("%d 를 왼쪽 1회 시프트하면 %d 이다.\n", a, a<<1);  
05         System.out.printf("%d 를 왼쪽 2회 시프트하면 %d 이다.\n", a, a<<2);  
06         System.out.printf("%d 를 왼쪽 3회 시프트하면 %d 이다.\n", a, a<<3);  
07     }  
08 }
```

왼쪽  
시프트한  
결과를  
출력한다.

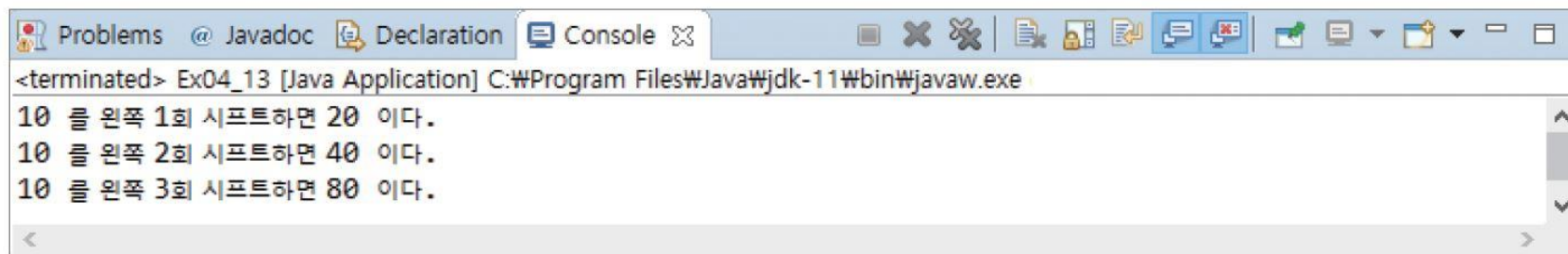


그림 4-24 실행 결과

## 05 비트연산자

### ■ 오른쪽 시프트 연산자 >>

- 나열된 비트를 오른쪽으로 시프트하는 연산자
- 26을 오른쪽으로 두 칸 시프트 연산
- 오른쪽으로 1회 시프트  $2^1$ , 2회 시프트  $2^2$ , 3회 시프트  $2^3$ 을 나눈것과 같음



그림 4-25 26을 오른쪽으로 두 칸 시프트 연산

## 05 비트연산자

### 실습 4-14 오른쪽 시프트 연산자 사용 예

```
01 public class Ex04_14 {  
02     public static void main(String[] args) {  
03         int a = 10;  
04         System.out.printf("%d 를 오른쪽 1회 시프트하면 %d 이다.\n", a, a >> 1);  
05         System.out.printf("%d 를 오른쪽 2회 시프트하면 %d 이다.\n", a, a >> 2);  
06         System.out.printf("%d 를 오른쪽 3회 시프트하면 %d 이다.\n", a, a >> 3);  
07         System.out.printf("%d 를 오른쪽 4회 시프트하면 %d 이다.\n", a, a >> 4);  
08     }  
09 }
```

오른쪽  
시프트한  
결과를  
출력한다.

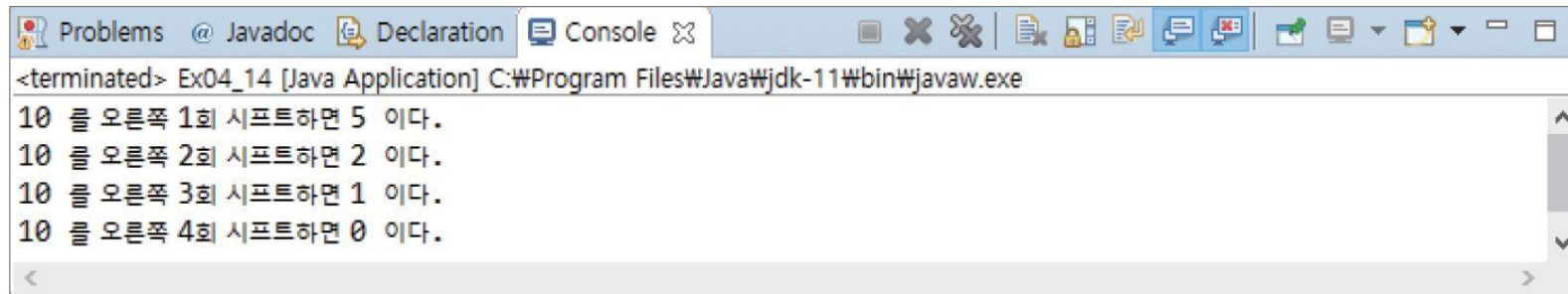


그림 4-26 실행 결과

**06**

## 연산자 우선순위

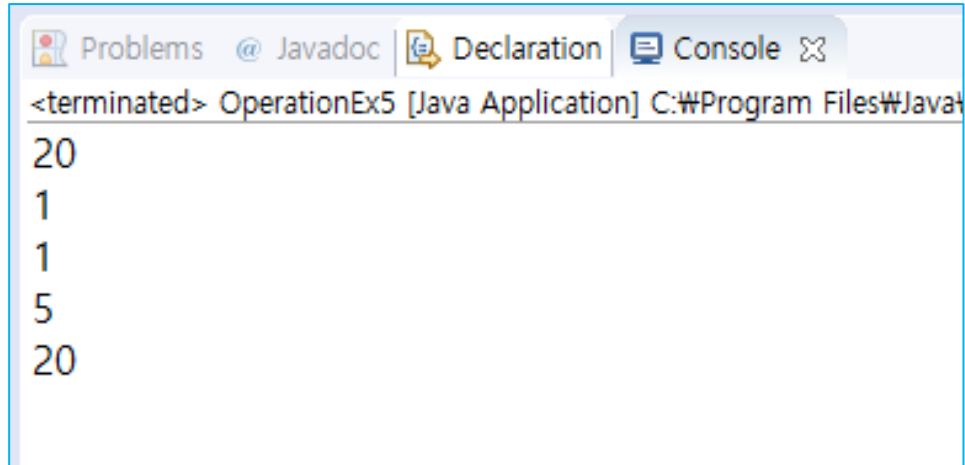
## 06 연산자 우선순위

표 4-7 연산자 우선순위

우선순위	연산자	설명	순위가 같을 경우 진행 방향
1	() [] .	1차 연산자	→
2	+ - ++ -- ~ ! (type)	단항 연산자[변수(또는 상수) 앞에 붙음]	←
3	* / %	산술 연산자	→
4	+ -	산술 연산자	→
5	<< >> >>>	비트 시프트 연산자	→
6	< <= > >= instanceof	비교 연산자	→
7	= !=	동등 연산자	→
8	&	비트 연산자	→
9	^	비트 연산자	→
10		비트 연산자	→
11	&&	논리 연산자	→
12		논리 연산자	→
13	?:	조건 삼항 연산자	→
14	= += -= *= /= %= %= ^=  = <<= >>=	대입 연산자	←

## Self study5-3

1. 정수 5에 대하여 비트 이동연산자를 이용하여 다음의 결과가 출력되도록 프로그램을 작성하고 그 결과를 출력하시오.



```
<terminated> OperationEx5 [Java Application] C:\Program Files\Java\
20
1
1
5
20
```

2. 다음 문제를 푸시오.

- a. (10진수)  $10^8$
- b. (16진수)  $0xFF \wedge 0xFF$
- c. (2진수)  $0101 \& 1100$
- d. (2진수)  $0101 \mid 1100$
- e. (2진수)  $1111 \& 0000$

질문은 이메일을 이용해주세요.  
[ds.june2@gmail.com](mailto:ds.june2@gmail.com)



**감사합니다**