연산자(operator)

연산자

연산자 종류와 우선순위

종류	구분	세부구분	연산자	우선순위
최우선연산자			( ) . [ ]	1
단항연산자			+ - ! (자료형) ++ -- ~	2
이항연산자	산술연산자		* / %	3
			+ -	4
	쉬프트연산자		>> << >>>	5
	비교연산자		> < >= <=	6
			== !=	7
	논리연산자	비트논리연산자	&	8
			^	9
			|	10
		일반논리연산자	&&	11
			||	12
삼항연산자			(조건식)?참일 때 사용할 값:거짓일 때 사용할 값	13
대입연산자	순수대입		=	14
	복합대입	산술대입	+= -= *= /= %=	
		쉬프트대입	<<= >>= >>>=	
		비트논리대입	&= ^= |=	
나열연산자			,	15

산술 연산자

산술연산자(+,-,*,/,%)

→ 기본 수학에서 사용하는 연산자와 유사하며 계산의 산술의 우선순위도 동일

+ : 더하기
- : 빼기
*** : 곱하기**
/ : 나누기의 몫
% : 나누기의 나머지

※ 주의해야 할 사항은 정수/정수를 수행하면 몫만 연산 → 3/2 = 1

산술 연산자

산술연산자(+,-,*,/,%)
→ 결과를 먼저 예상하고 실제로 작성해보세요.

예제 1)

```
int a = 10;
int b= 3;
System.out.println(“a+b = ”+(a+b));
System.out.println(“a-b = ”+(a-b));
System.out.println(“a*b = ”+(a*b));
System.out.println(“a/b = ”+(a/b));
System.out.println(“a%b = ”+(a%b));
```

대입 연산자

대입 연산자(=,+=,-=,*=,/=,%=)

→ 산술연산 후 변수에 바로 값을 대입하는 연산자

= : 오른쪽 값을 왼쪽 공간(변수)에 대입
+= : 왼쪽과 오른쪽 값을 더한 후 값을 왼쪽 공간(변수)에 대입
-= : 왼쪽과 오른쪽 값을 뺀 후 값을 왼쪽 공간(변수)에 대입
***= : 왼쪽과 오른쪽 값을 곱한 후 값을 왼쪽 공간(변수)에 대입**
/= : 왼쪽 값을 오른쪽 값으로 나누고 몫을 왼쪽 공간(변수)에 대입
%= : 왼쪽 값을 오른쪽 값으로 나누고 나머지를 왼쪽 공간(변수)에 대입

대입 연산자

대입 연산자(=,+=,-=,*=,/=,%=)

예제 1)

```
int a = 10;
int b= 3;
int c = 6;
System.out.printf("Result : %d, %d, %d \n",a,b,c);
a += 3;
b *= 4;
c %=5;
System.out.printf("Result : %d, %d, %d \n",a,b,c);
```

증감 연산자

산술연산자(++,--)

→ 값을 하나 증가, 하나 감소시키는 경우에 사용되는 단항 연산자
1. 전위 연산 : 변수에 저장된 값을 증/감 시킨 후 연산
2. 후위 연산 : 연산 수행 후 변수에 저장된 값을 증/감

++a : 값을 1 증가 후 연산을 진행(선 증가, 후 연산)
a++ : 연산을 진행한 후 값을 1 증가(선 연산, 후 증가)
--a : 값을 1 감소 후 연산을 진행(선 감소, 후 연산)
a-- : 연산을 진행한 후 값을 1 감소(선 연산, 후 감소)

증감 연산자

산술연산자(++,--)

→ 결과를 먼저 예상하고 실제로 작성해보세요.

예제 1)

```
int a = 10;
int b= 10;
System.out.println(a);
System.out.println(a++);
System.out.println(a);

System.out.println(b);
System.out.println(++b);
System.out.println(b);
```

예제 2)

```
int a = 10;
int b= (a--) + 2;
System.out.println(a);
System.out.println(b);
```

증감 연산자

산술연산자(++,--)

→ 결과를 먼저 예상하고 실제로 작성해보세요.

예제 3)

```
int a = 10;
int b= 10;
int c;
c = (a++) + (++b) + a;
System.out.println(a);
System.out.println(b);
System.out.println(c);
```

관계 연산자

관계연산자(<,>,==,!=,<=,>=)

→ 두 개의 관계를 따지는 연산자로 비교연산자 라고도 함
→ 관계 연산자는 조건을 만족하면 true, 만족하지 못하면 false를 반환(boolean형 결과값)

a<b : a가 b보다 작으면 true
a>b : a가 b보다 크면 true
a<=b : a가 b보다 작거나 같으면 true
a>=b : a가 b보다 크거나 같으면 true
a==b : a와 b가 같으면 true
a!=b : a와 b가 같지 않으면 true

※ 관계 연산자는 두 값을 비교만 하는 것이 아니라, 비교한 결과에 해당하는 논리값을 반환

관계 연산자

관계연산자(<,>,==,!=,<=,>=)

예제 1)

```
int a = 10;
int b= 20;
boolean result1, result2, result3;
result1 = (a==b);
result2 = (a<=b);
result3 = (a>b);
System.out.println( “result1 : ”+result1);
System.out.println( “result2 : ”+result2);
System.out.println( “result3 : ”+result3);
```

논리 연산자

논리연산자(&&, ||, !)

→ 여러 조건을 동시에 검사할 때 주로 사용하는 연산자
→ and(그리고), or(또는), not(그러나)를 표현하는 연산자

a&&b : a와 b가 모두 true인 경우 true
a||b : a또는 b가 true인 경우 true(a나 b 둘 중 하나라도 true면 true)
!a : a가 true면 false, a가 false면 true

a	b	a && b	a || b
true	true	true	true
true	false	false	true
false	true	false	true
false	false	false	false

논리 연산자

논리연산자(&&, ||, !)

예제 1)

```
int a = 10;
int b= 12;
boolean result1, result2, result3;
result1 = (a==10 && b==12);
result2 = (a<12 || b>12);
result3 = !(a==10);
System.out.println( “result1 : ”+result1);
System.out.println( “result2 : ”+result2);
System.out.println( “result3 : ”+result3);
```

비트 연산자

비트 연산자(&,|,^,~,<<,>>)

→ 2진수로 변환하여 비트단위의 연산을 수행하는 연산자
→ 정수형 타입에서만 사용이 가능
→ 주로 하드웨어를 직접 제어하거나 한글 등을 처리할 때 사용

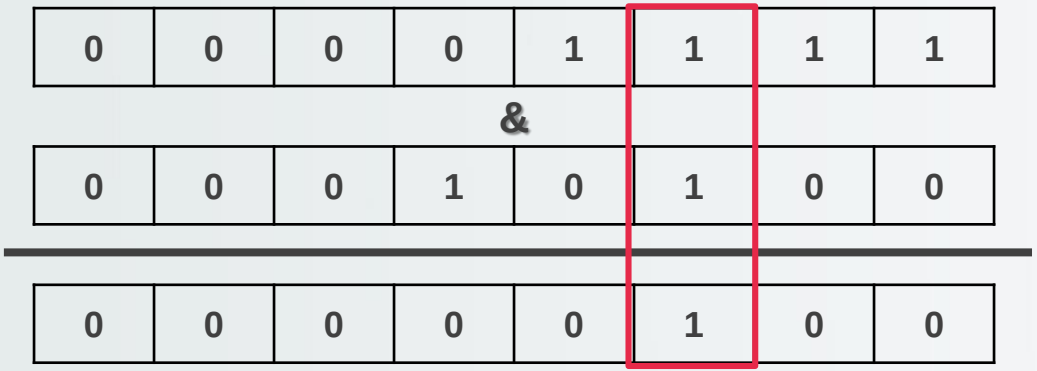
a&b : 비트단위 AND연산(둘 다 1인 경우 1)
a|b : 비트단위 OR연산(둘 중 1개라도 1인 경우 1)
a^b : 비트단위 XOR연산(두 비트가 같으면0, 다르면 1)
~a : 비트 반전연산(0이면1로, 1이면 0으로)
a<<1 : 왼쪽 쉬프트 연산(비트를 왼쪽으로 1칸씩 이동)
a>>1 : 오른쪽 쉬프트 연산(비트를 오른쪽으로 1칸씩 이동)

비트 연산자

비트단위 AND연산

→ 둘 다 1인 경우 1로 연산

```
int a = 15;                 //15를 2진수로 변환하면 → 00001111
int b = 20;                 //20을 2진수로 변환하면 → 00010100
int c = a & b;
System.out.println(c);      //결과는 4
```



비트 연산자

비트단위 OR연산

→ 둘 중 하나라도 1인경우 1

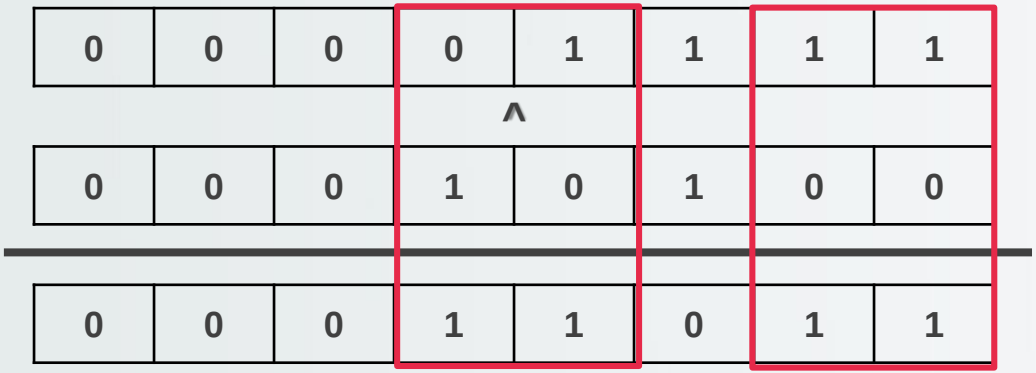
```
int a = 15;                 //15를 2진수로 변환하면 → 00001111
int b = 20;                 //20을 2진수로 변환하면 → 00010100
int c = a | b;
System.out.println(c);      //결과는 31
```

비트 연산자

비트단위 XOR연산

→ 두비트가 같으면 0 다르면 1

```
int a = 15;                 //15를 2진수로 변환하면 → 00001111
int b = 20;                 //20을 2진수로 변환하면 → 00010100
int c = a ^ b;
System.out.println(c);      //결과는 27
```



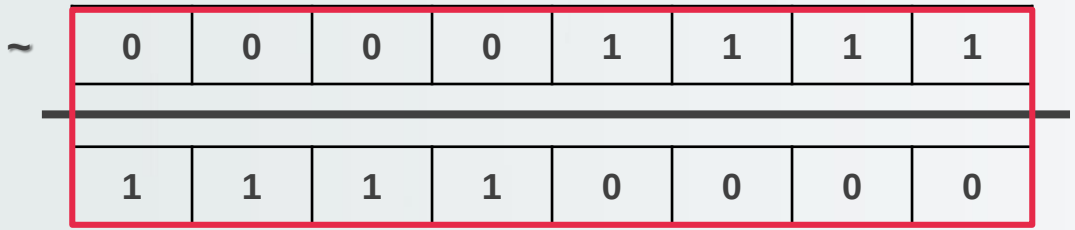
※ 대문자 → 소문자, 소문자 → 대문자로 변환 시 XOR를 사용하면 쉽게 가능함

비트 연산자

비트단위 반전연산

→ 0이면 1로, 1이면 0으로 변경

```
int a = 15;                 //15를 2진수로 변환하면 → 00001111
int b = ~a;
System.out.println(b);      //결과는 -16
```

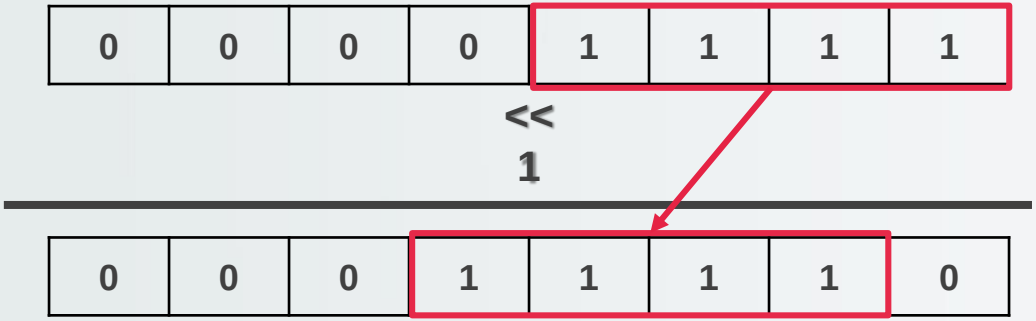


비트 연산자

왼쪽 쉬프트 연산

→ 비트의 자리를 모두 왼쪽으로 정해진 숫자만큼 이동

```
int a = 15;                    //15를 2진수로 변환하면 → 00001111
int b = a<<1;
System.out.println(b);         //결과는 30
```



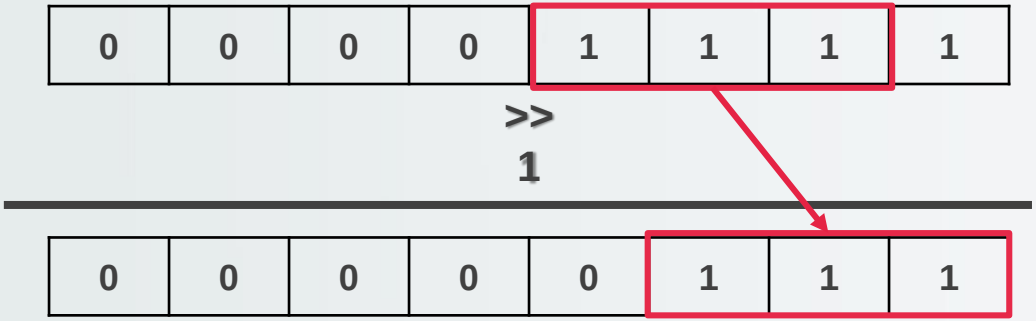
※ 빈공간은 0으로 채우며, 왼쪽으로 한칸씩 이동할 때마다 숫자가 2배로 커짐

비트 연산자

오른쪽 쉬프트 연산

→ 비트의 자리를 모두 오른쪽으로 정해진 숫자만큼 이동

```
int a = 15;                    //15를 2진수로 변환하면 → 00001111
int b = a>>1;
System.out.println(b);         //결과는 7
```



※ 남은 숫자는 버리며, 왼쪽으로 한칸씩 이동할 때마다 숫자가 반으로 줄어 듬

삼항 연산자

삼항 연산자

→ 조건식을 이용하여 결과값에 따라 연산을 처리하는 방식

변수 = 조건식? 조건식 true일 때 값:조건식 false일 때 값

예제 1)

```
int a = 10;
int b= 20;
int result = a<b?a:b;
System.out.println( “result : ”+result);
```

※ a와 b를 비교하여 a가 b보다 작은 경우 조건식이 true이므로 result에 a의 값이 대입되고, a가 b보다 큰 경우 조건식이 false가 되므로 b의 값이 대입된다. 결국 result에는 a와 b를 비교하여 둘 중 작은 수가 항상 대입된다.

실습문제

연산자 실습

문제 1

두수를 입력 받고 더한 수, 뺀 수, 곱한 수, 나눈 수가 출력 되도록 만들어 보시오. 단, 입력 받은 두 수는 정수형 변수에 저장하고 나눈 수는 소수점 둘째자리 까지 출력되야 함

```
첫번째 수 입력 : 10
두번째 수 입력 : 3
========== 결과 ==========
두 수를 더한 수 : 13
두 수를 뺀 수 : 7
두 수를 곱한 수 : 30
두 수를 나눈 수 : 3.33
```

문제 2

다음 코드를 보고 각각 무엇이 출력 될 지 맞추어라.(결과를 생각 한 수 코드를 작성하여 확인해볼 것)

```java
int a = 40, b = 20;
a += b;
System.out.printf("a의 값: %d , b의 값 : %d \n", a, b);
a -= b;
System.out.printf("a의 값: %d , b의 값 : %d \n", a, b);
b *= a;
System.out.printf("a의 값: %d , b의 값 : %d \n", a, b);
b /= a;
System.out.printf("a의 값: %d , b의 값 : %d \n", a, b);
```

연산자 실습

문제 3

다음 코드를 보고 각각 첫번째 결과값과 두번째 결과값이 무엇이 출력될지 맞추어라.(결과를 생각한 후 코드를 작성하여 확인 해볼것)

```
int a = 10, b = 20, c = 30, d = 40;
boolean result1, result2;
result1 = ((a < 20 && b > 10) && (c == 20 || d == 40));
result2 = ((a == 10 && b != 2 * 10) || (c == 30 && d != 40));
System.out.printf("첫번째 결과값 : %b \n", result1);
System.out.printf("두번째 결과값 : %b \n", result2);
```

문제 4

나이를 입력 받아 나이가 19세보다 많으면 “성인입니다. 어서오세요”, 19세보다 어리면 “미성년자는 입장불가입니다.”라고 출력

```
나이를 입력하세요 : 20
성인입니다.어서오세요
```

```
나이를 입력하세요 : 15
미성년자는 입장불가입니다
```

연산자 실습

문제 5

새 프로젝트 생성(프로젝트 명 자유롭게 대신 표기법은 확인)

1. **실행용 클래스**
   - 패키지명 : kh.java.operator.run
   - 클래스명 : Main
   - 내용 : 기능제공 클래스의 sample1() 메소드 실행

2. **기능제공 클래스**
   - 패키지명 : kh.java.test.function
   - 클래스명 : Exmaple
   - 메소드 명 : public void exam1(){}
   - 내용 : 국어, 영어, 수학 점수를 입력 받고 합계와 평균을 계산 및 출력 하고 합격/불합격 처리
     조건 : 각점수가 40점이상이며 평균이 60점이상인경우 합격, 그 외 불합격
   - 메소드 명 : public void exam2(){}
   - 내용 : 정수 하나를 입력받고 짝수면 “짝수!” 홀수면 “홀수!” 출력

실행결과 예시

exam1() 메소드 실행 결과

```
국어 점수 입력 : 60
영어 점수 입력 : 60
수학 점수 입력 : 60
합계 : 180
평균 : 60.00
합격
```

```
국어 점수 입력 : 60
영어 점수 입력 : 60
수학 점수 입력 : 50
합계 : 170
평균 : 56.67
불합격
```

exam2() 메소드 실행 결과

```
정수 입력 : 10
짝수!
```

```
정수 입력 : 11
홀수!
```