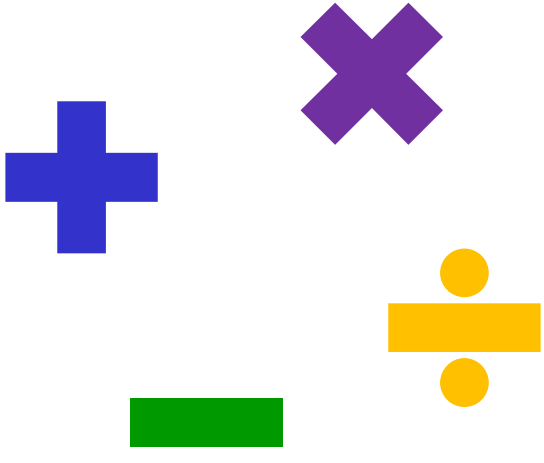
4강

JAVA_PROGRAMMING

연산자

❖ 연산자

- 데이터를 가공하기 위해 사용하는 기호 집합
- 기존의 데이터를 연산자를 통해 계산하여 원하는 값을 얻기 위해 사용



연산자 우선순위

- ❖ 프로그램의 일반적인 실행순서
  - 왼쪽에서 오른쪽, 위에서 아래
- ❖ 연산자가 사용되면 실행 순서가 달라질 수 있다.
- ✓ ex)

```
boolean bool = 4 + 5 > 2 - 1 * 7 && (12 & 3 * 2) > 7 || -2 != 2;
```

bool 변수에 저장되는 값은??

bool =

이유??

연산자의 우선순위

❖ 연산자의 우선순위

최우선 연산자	.(점) , [] , ()
단항 연산자	! , ~ , +/- , ++/-- , (cast 자료형) ,instanceof
산술 연산자	+, -, *, /, %
Shift 연산자	<< , >> , >>>
관계 연산자	< , > , <= , >= , == , !=
비트 연산자	& , | , ^
논리 연산자	&& , ||
삼항 연산자	(조건항) ? 참 항 : 거짓 항
배정 대입 연산자	=, *=, /=, %=, +=, -=, <<=, >>=, >>>=
증감 후위 연산자	++/--
순차 연산자	,(콤마)

최우선 연산자(. [] ())

- ✓ .(점) : 참조연산자 – 클래스의 멤버를 사용하기 위해 사용됨
- ✓ [] : 배열참조 – 배열의 위치 번째를 나타낼 때 사용됨
- ✓ () : 괄호 – 특정연산자를 묶어 먼저 처리할 수 있게 만들어 주는 연산자

```
public class Exec_01 {
    public static void main(String[] ar) {
            System.out.println("Test");
            int x = 5 + 3 * 2;
            int y = (5 + 3) * 2;
            System.out.println("x = " + x);
            System.out.println("y = " + y);
    }
}
```

단항 연산자[!]

- ✓ ! (논리 부정)
  - 논리 자료형의 데이터 값을 부정하는 연산자
  - 반드시 **논리 자료형에만 사용**이 가능하다.

```
public class test {
        public static void main(String []args){
                boolean bool1 = false;
                boolean bool2 = !bool1;
                System.out.println("bool1 = "+bool1);
                System.out.println("bool2 = "+bool2);
        }
}
```

단항 연산자[~]

✓ ~ (비트 부정)

- 비트 값으로 존재하는 모든 자료를 비트단위로 부정한다.
- boolean, float, double 형은 사용 불가
- byte, short, int, char 형은 연산 결과를 int, long에 담을 수 있다.
- long은 연산 결과를 long에만 담을 수 있다.

```
public class test {
        public static void main(String []args){
                byte A = 12;
                // byte B = ~A; 컴파일 에러
                int  C = ~A;
                long D = 12l;
                // int E = ~D; 컴파일 에러
                long F = ~D;
        }
}
```

단항 연산자[+ - ++ --]

- +/-
  - 양수와 음수를 판별하는 연산자
- ++/-- (증감 연산자)
  - 값을 하나 증가시키거나 감소시키는 연산자
  - 전위 연산이 우선순위가 높음(++a,--a);
  - 후위 연산은 우선순위가 가장 낮음(a++,a--,'콤마'보다는 높음)

```
public class test {
        public static void main(String []args){
                int a = 10;
                int b = ++a; //전위 연산(순위 높음)
                int c = a++; //후위연산(순위 낮음)
                System.out.println("b = "+b);
                System.out.println("c = "+c);
        }
}
```

산술 연산자(+ - * / %)

❖ 산술 연산자

산술 연산자	사용 예	의 미
+	a+b	두수의 합
-	a-b	두수의 차
*	a*b	두수의 곱
/	a/b	나누기 몫
%	a%b	나누기 나머지

산술 연산 결과

❖ byte, short, int 의 산술 연산 결과는 int형이다.

byte a = 10;

short s = 20;

위 두 자료를 연산하면 a + s = 30 (int형)

❖ long, float, double 자료가 산술 연산에 포함 될 경우

*50 + 2L * 3 => long형 결과로 56*

1.2f + 50L – 12 => float형 결과로 39.2

✓ 크기 순으로 long < float < double 이며 이 중 큰 자료형으로 결과나 나타난다.

❖ float과 double은 되도록 연산을 피한다.

✓ float과 double의 정밀도의 차이로 제대로 된 값이 나오지 않는다.

float f = 1.2f;

double d = 1.2;

double dd = d – f;

System.out.println("d – f = "+dd);

Shift 연산자[<< >> >>>]

❖ shift연산은 대상 필드를 2진 비트로 바꾼 후 비트를 이동 시키는 연산

- int 보다 작은 자료형을 shift연산 하면 int형으로 변환되어 연산
- long형은 64bit로 연산
- boolean, float, double형은 연산 불가

✓ < Left Shift >

int a = 1;

a<<2; (a의 값을 비트로 바꾸어서 왼쪽으로 2번 이동)

이동 전 =>	0000 0000 0000 0000 0000 0000 0000 0001
이동 후 =>	0000 0000 0000 0000 0000 0000 0000 0100

결과 = 4

Shift 연산자[<< >> >>>]

✓ < Right Shift >

int a = 4

a>>1;

(a의 값을 비트로 바꾸어서 오른쪽으로 1번 이동)

이동 후 => 0000 0000 0000 0000 0000 0000 0000 0010

결과 = 2

✓ <Unsigned Right Shift >

int a = -8;

a>>>3;

(a의 값을 오른쪽으로 3번 이동 후 빈 곳은 무조건 0으로 채운다.)

이동 전 => 1111 1111 1111 1111 1111 1111 1111 1000

이동 후 => 0001 1111 1111 1111 1111 1111 1111 1111

결과 = 536870911

관계 연산자[< <= > >= == !=]

❖ 비교관계 연산자

- <, <=, >, >=
- 왼쪽과 오른쪽 항의 값을 비교하여 참(true), 거짓(false)의 값을 나타낸다.
- 논리형 결과가 필요한 곳에 사용한다.

비교관계 연산자	사용 예	결과(논리형)
a < b	20 < 20	false
a <= b	10 <= 10	true
a > b	20 > 10	true
a >= b	20 >= 30	false

관계 연산자[< <= > >= == !=]

❖ 항등 관계연산자

- ==, !=
- 비교 관계 연산자 보다 낮은 우선순위를 가진다.
- 오른쪽 왼쪽의 항이 같은지 다른지를 판단하여 참(true), 거짓(false)을 판별한다.

항등 관계 연산자	사용 예	결과
==	10 == 20	false
!=	20 != 10	true

비트 연산자[& ^ |]

- ❖ 비트연산자는 shift연산자 처럼 데이터를 비트(2진수)로 변환하여 비트간 연산
- ❖ 자바에서는 일반 논리 연산도 가능 (우선 순위 & > ^ > |)
- ❖ int보다 작은 자료형 연산 결과는 int, long결과는 long
- ✓ &(AND 연산)
  - 연산되는 비트가 모두 1이어야 1값을 나타내는 연산
- ✓ ^(Exclusive OR 연산)
  - 연산되는 비트가 다르면 1 같으면 0을 나타내는 연산
- ✓ |(OR 연산)
  - 연산되는 비트가 하나라도 1이면 1값을 나타내는 연산

```
byte a = 7;
char b = 4;
int c = a ( & ^ | )  b;
          0000 0000  0000 0000  0000 0000  0000 0111
(& ^ |)   0000 0000  0000 0000  0000 0000  0000 0100
```

논리 연산자[&& ||]

- 논리 연산자는 양쪽 데이터의 논리결과에 따라 true/false를 나타낸다.
- && (AND연산)
  - 양쪽 결과가 모두 true이어야 true를 나타낸다.
- || (OR 연산)
  - 양쪽 결과 중 하나만 true라도 true를 나타낸다.

```
int x = 10;
int y = 20;
boolean bool = ++x > y && x < ++y;
```

✓ <결과>

bool =

x =

y =

```
boolean bool = ++x > y & x < ++y;
```

✓ <결과>

bool =

x =

y =

삼항 연산자(? :)

- ❖ 조건식에 따라 참인 경우 앞부분의 식을 거짓인 경우 뒷부분을 수행
- ❖ 사용 예
  - (조건식) ? 참인 경우 : 거짓인 경우 ;
  - 조건식이 참인 경우 ?뒤의 내용을 거짓일 경우 : 뒤의 내용을 실행

ex)

```
short a = 10;
short b = 20;
short c = (a > b) ? a : b;
System.out.println("c = "+c);
```

출력결과 =

예제

❖ 삼항 연산자 사용 시 주의

```
int a = 10;
(a == 10) ? System.out.println("같다.") : System.out.println("다르다.");
```

✓ 위 예제는 에러

```
int a = 10;
System.out.println((a == 10) ? "같다" : "다르다" );
```

✓ 위처럼 사용해야 한다.

배정 대입연산자

- 단순 대입은 오른쪽 값을 왼쪽으로 대입한다.
- 배정 대입은 연산 후의 결과를 오른쪽으로 대입한다.
- 수치형 결과를 나타내는 연산자들에 자료를 유지시키면서 연산 가능

배정 대입 연산자	사용 예	의 미
+=	a+=b	a=a+b
-=	a-=b	a=a-b
*=	a*=b	a=a*b
/=	a/=b	a=a/b
%=	a%=b	a=a%b

- '<<=' '>>=' '^=' '&=' '|=' 등의 연산도 가능하다.

Quiz

```
public class test{
    public static void main(String []args){
        int a = 5, b = 6, c = 10, d = 0;
        boolean bo = false;
        d = ++a * b--;
        System.out.printf("a = %d, b = %d, d = %d₩n",a,b,d);
        d = a++ + ++c - b--;
        System.out.printf("a = %d, b = %d, c = %d, d = %d₩n",a,b,c,d);
        a = 1;
        b = 0;
        bo = a++ > 0 || 1 < ++b * d-- / ++c;
        System.out.printf("a = %d, b = %d, c = %d, d = %d₩n",a,b,c,d);
        bo = b++ > 0 && 1 < ++a / ++c * d++;
        System.out.printf("a = %d, b = %d, c = %d, d = %d₩n",a,b,c,d);
    }
}
```

연산자의 우선순위

❖ 연산자의 우선순위

최우선 연산자	.(점) , [] , ()
단항 연산자	! , ~ , +/- , ++/-- , (cast 자료형) ,instanceof
산술 연산자	- +, -, *, /, %
Shift 연산자	<< , >> , >>>
관계 연산자	< , > , <= , >= , == , !=
비트 연산자	& , | , ^
논리 연산자	&& , ||
삼항 연산자	(조건항) ? 참 항 : 거짓 항
배정 대입 연산자	=, *=, /=, %=, +=, -=, <<=, >>=, >>>=
증감 후위 연산자	++/--
순차 연산자	,(콤마)

Quiz

- 연산자의 우선순위를 적어본다(필히 암기해야 함)

- 값을 받아들여 1이면 합격을 2이면 불합격이라는 글자를 출력하는 프로그램을 만드시오
  - – 합격여부를 입력하세요.(합격 : 1, 불합격 : 2) : 1
  - – 당신은 합격 입니다.

- 문자 하나를 입력 받아서 그 문자가 산술 연산자인지 아닌지를 판단하는 프로그램을 만드시오.
  - – 산술연산자를 입력하세요(+,-,*,/,%) : +
  - – + 는 산술연산자입니다.
  - – 산술연산자를 입력하세요(+,-,*,/,%) : a
  - – a 는 산술연산자가 아닙니다.