

# **COMP217**

# **Java Programming**

## **Spring 2021**

*Week 2, 3*  
*Expressions*

# Goals

- 각종 연산자(Operators)
- 수식의 계산(Arithmetic expressions)

# Expressions

Operator / Operand

Arithmetic operators / expressions

Relational operators / expressions

operator	precedence
in/decrement, postfix	expr++ expr--
unary	++expr --expr +expr ~ !
multiplicative	* / %
addition/subtraction	+ -
bit shift	<< >>> >>
relational	< > <= >= instanceof
equivalence test	== !=
bitwise AND	&
bitwise XOR	^
bitwise OR	
logical AND	&&
logical OR	
conditional	? :
assignment	= += -= *= /= %= &= ^=  = <<= >>= >>>=

# 수식

- 수식
  - 프로그래밍 언어에서 연산자와 피연산자의 조합으로 구성된 연산식
  - 표현식은 항상 하나의 결과 값이 있음
- 연산자와 피연산자
  - 연산자(operator)
    - $+$ ,  $-$ ,  $*$  기호와 같이 이미 정의된 연산을 수행하는 문자 또는 문자 조합 기호
  - 피연산자(operand)
    - 연산(operation)에 참여하는 변수나 상수
- 피연산자의 수에 따라
  - 단항(unary operator), 이항(binary operator), 삼항 연산자(ternary operator)

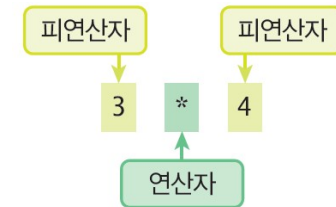
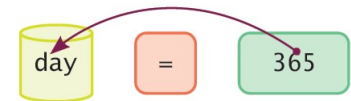


그림 3-1 • 연산자와 피연산자

# 대입연산자, 산술연산자

- 대입 연산자(assignment operator)
  - 연산자의 오른쪽 값을 왼쪽 변수에 저장하는 연산자
  - 대입 연산자의 왼쪽은 반드시 값을 저장할 수 있는 변수
    - 대입 연산자는 할당 또는 치환 연산자라고도 부름

```
int day = 365;
```



- 산술연산자 +, -, \*, /, %

구분	연산자	연산자 의미	예	결과	비고
부호 연산자	+	양수 부호	+3	3	
	-	음수 부호	-7	-7	
산술 연산자	+	더하기	3 + 7	10	문자열 연결 연산자 "java" + "lang"
	-	빼기	7 - 3	4	
	*	곱하기	7 * 3	21	별표 * 기호 <small>asterisk</small>
	/	나누기	7 / 2	3	정수와의 나누기는 결과도 정수
	%	나머지	7 % 2	1	백분율이 아님

# 산술 연산자(Arithmetic operators)

```
1 public class ArithmeticOperator {  
2     public static void main (String args[] ) {  
3         System.out.println(3+2);  
4         System.out.println(3-2);  
5         System.out.println(3*2);  
6         System.out.println(3/2);  
7         System.out.println(3%2);  
8         System.out.println(3.0/2.0);  
9         System.out.println(3.0/2);  
10        System.out.println(3/2.0);  
11        System.out.println(3.5%2);  
12    }  
13 }  
14
```

# Problem: Converting Temperatures

- Write a program that converts a Fahrenheit degree to Celsius using the formula:

$$celsius = (\frac{5}{9})(fahrenheit - 32)$$


- Be careful with integer division

```
/* wrong */  
celsius = (5 / 9) * (fahrenheit - 32);  
// is equivalent to 0 * (fahrenheit-32);
```


```
/* correct */  
celsius = (5.0 / 9) * (fahrenheit - 32);
```

# Increment and Decrement Operators, cont.

```
int i = 10;  
int newNum = 10 * i++;
```

Same effect as 

```
int i = 10;  
int newNum = 10 * (++i);
```

Same effect as 

- Example) Today is Saturday. What day is in 10 days? You can find that day is Tuesday using the following expression:



# 단항 연산자(Unary operators)

연산자	설명
+X	no operation
-X	reverses the sign
++X	increment x first, then use it
X++	use the value of x, and increment x
--X	decrement x first, then use it
X--	use the value of x, and decrement x

- ++
  - 변수의 값을 1 증가
- --
  - 변수의 값을 1 감소

```
1 public class UnaryOperator {
2     public static void main (String args[] ) {
3         int x = 1;
4         int y = -1;
5         int z;
6
7         z = +x;      //z에 x대입 (+연산자는 의미 없음)
8         System.out.println(z);
9         z = +y;      //z에 y대입 (+연산자는 의미 없음)
10        System.out.println(z);
11        z = -x;      //z에 x의 부호를 바꿔 대입
12        System.out.println(z);
13        z = -y;      //z에 y의 부호를 바꿔 대입
14        System.out.println(z);
15        z = ++x;      //x를 하나 증가시킨 후 z에 대입
16        System.out.println(z);
17        z = y++;      //y를 z에 대입시킨 후 y 하나 증가시킴
18        System.out.println(z);
19        z = y;        //z에 y대입
20        System.out.println(z);
21    }
22 }
```

## 그 외의 연산자 타입

복합대입연산자	의미
	$x = x + y$
	$x = x - y$
	$x = x * y$
	$x = x / y$
	$x = x \% y$

관계 연산자	의미
$x == y$	true if the values in x and y are equal
$x != y$	true if the values in x and y are NOT equal
$x > y$	true if x is greater than y
$x < y$	true if x is less than y
$x >= y$	true if x is greater than or equal to y
$x <= y$	true if x is less than or equal to y

- 관계 연산자
  - 결과 값은 boolean 값인 true 또는 false
- 우선 순위
  - 단항 연산자-> 산술 연산자-> 관계 연산자-> 대입연산자

# 관계 연산자(Relational Operator)

```
1 public class ComparisonOperator {  
2     public static void main (String args[] ) {  
3         int x, y;  
4  
5         x = 3; y = 4;  
6         System.out.println(x == y); //false  
7         System.out.println(x != y); //true  
8         System.out.println(x > y);  //false  
9         System.out.println(x < y);  //true  
10        System.out.println(x <= y); //true  
11  
12        x = 3; y = -4;  
13        System.out.println(x == y); //false  
14        System.out.println(x != y); //true  
15        System.out.println(x > y);  //true  
16        System.out.println(x < y);  //false  
17        System.out.println(x <= y); //false  
18    }  
19 }
```

# 논리 연산자(Logical Operators)

논리 연산자	의미
x && y	Logical AND: x ,y 모두 참일때 true
x    y	Logical OR: x 또는 y가 참일때 true
!x	true ↔ false

```
1 public class LogicalOperator {
2     public static void main (String args[] ) {
3         int x, y;
4
5         x = 3; y = 4;
6
7         System.out.println("실행 결과 : ");
8         System.out.println((x==3) && (y==4));
9         System.out.println((x==3) && (y==7));
10        System.out.println((x==3) || (y==4));
11        System.out.println((x==5) || (y==4));
12        System.out.println(!(x==3)); // (x==3)이 true인데 그것을 부정하여 false
13        System.out.println(!(x==5)); // (x==5)이 false인데 그것을 부정하여 true
14    }
15 }
16
```

# 우선 순위

```
result = x * y % z - a / b
```

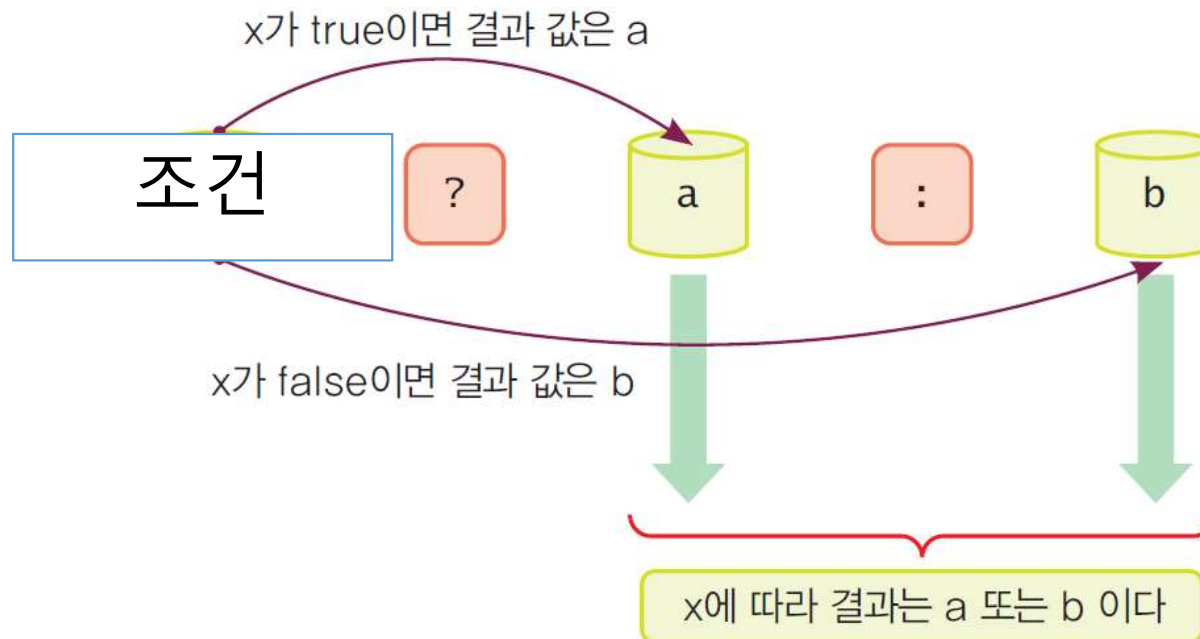
```
m = ( x + y + z ) / 3
```

```
result = x % y * z;
```

```
x = y = w = z ;
```

# 삼항 연산자(Ternary operator)

- 조건 연산자(conditional operator)
  - 조건의 논리 값에 따라 2개의 피연산자 중 하나가 결과 값
  - 유일한 삼항 연산자



# 삼항 연산자-Example

```
1 import java.util.Scanner;
2 public class TernaryOperator{
3     public static void main(String args []){
4         int a, b;
5         Scanner sc = new Scanner(System.in);
6         System.out.println("정수 두개를 입력하세요.");
7         a = sc.nextInt();
8         b = sc.nextInt();
9
10        System.out.println("max : "+((a>b)?a:b));
11
12
13
14
15    }
16 }
```

```
C:\Users\W썩꼰레\Desktop\COMP217>java TernaryOperator
정수 두개를 입력하세요.
5 1
max : 5
입력하신 두 수는 5, 1이며, 최대는 앞의 수입니다.
```

```
C:\Users\W썩꼰레\Desktop\COMP217>java TernaryOperator
정수 두개를 입력하세요.
2 4
max : 4
입력하신 두 수는 2, 4이며, 최대는 뒤의 수입니다.
```

```
C:\Users\W썩꼰레\Desktop\COMP217>java TernaryOperator
정수 두개를 입력하세요.
1
1
max : 1
입력하신 두 수는 1, 1이며, 두 수는 동일합니다.
```

# 비트 연산자(&, |, ^, ~)

피연산자는 int형

비트연산자	의미	Examples
~x	bitwise negation	~(0x0FFF) == 0xFFFFF000 [Q] why not 0xF000?
x & y	bitwise AND	(0x0FFF & 0xFFF0) == 0x0FF0 [Q] why not 0xFFFFF0FF0?
x ^ y	bitwise XOR	(0x0FFF ^ 0xFFF0) == 0xF00F [Q] Where are first 16 bits?
x   y	bitwise OR	(0x0FFF   0xFFF0) == 0xFFFF [Q] why not 0xFFFFFFFF?
x << n	n bits shift to left	0x0FFF << 4 == 0xFFF0
x >> n	n bits shift to right	0xFFF0 >> 4 == 0x0FFF

```
public class BitOperator {  
    public static void main (String args[] ) {  
        int x, y;  
  
        x = 0x0fff; y = 0xffff0;  
        System.out.println("실행결과");  
        System.out.printf("x \t: %8x\n", x); //x : 전체 폭을 8칸으로 두고 16진수로 표현  
        System.out.printf("y \t: %8x\n", y);  
        System.out.printf("(x & y) : %8x\n", (x & y)); //AND  
        System.out.printf("(x | y) : %8x\n", (x | y)); //OR  
        System.out.printf("(x ^ y) : %8x\n", (x ^ y)); //Xor, 두 값이 같으면 0, 다르면 1  
        System.out.printf("~x\t: %8x\n", ~x); //NOT  
        // \t는 탭만큼 간격두기  
        System.out.printf("(x << 4) : %8x\n", (x << 4)); //왼쪽으로 4비트씩 이동  
        System.out.printf("(x >> 4) : %8x\n", (x >> 4)); //오른쪽으로 4비트씩 이동  
    }  
}
```

실행결과

x	:	fff
y	:	ffff0
(x & y)	:	ff0
(x   y)	:	ffff
(x ^ y)	:	f00f
~x	:	fffff000
(x << 4)	:	ffff0
(x >> 4)	:	ff

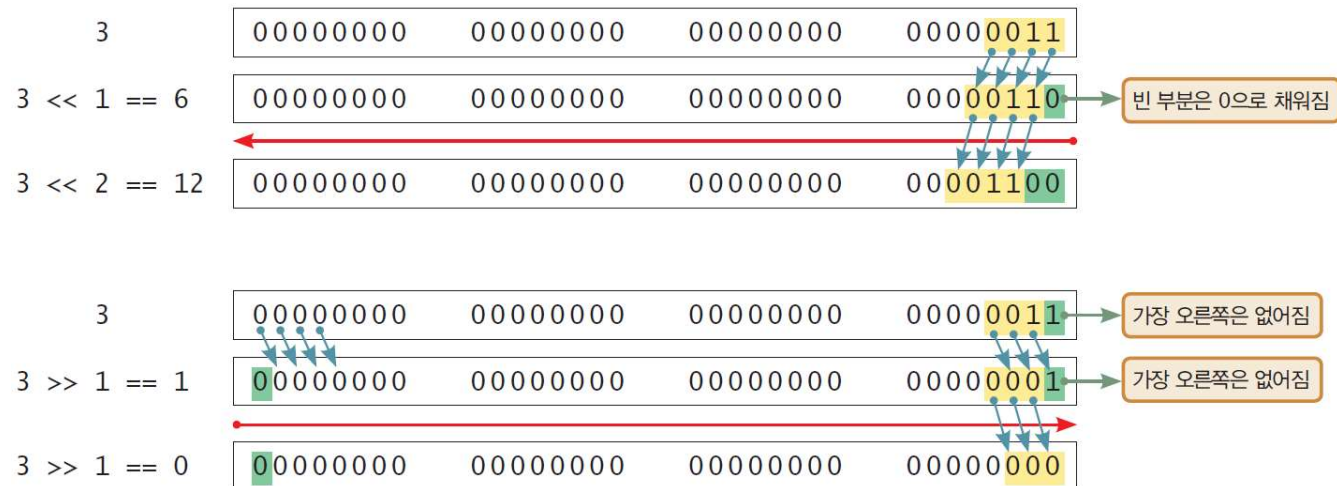


# 비트 연산자(shift)

## • 비트 이동 연산자(bit shift operators)

- >>, <<, >>>

연산자	이름	사용	연산 방법	새로 채워지는 비트
>>	Signed left shift	op1 >> op2	op1을 오른쪽으로 op2 비트만큼 이동	가장 왼쪽 비트인 부호 비트는 원래의 비트로
<<	Signed right shift	op1 << op2	op1을 왼쪽으로 op2 비트만큼 이동	가장 오른쪽 비트를 모두 0으로 채움
>>>	Unsigned right shift	op1 >>> op2	op1을 오른쪽으로 op2 비트만큼 이동	가장 왼쪽 비트인 부호 비트는 모두 0으로 채워짐



# 중간 점검 문제

1. 다음의 각 변수의 값을 적어보라.

```
int x = 1;
int y = 1;
int a = ++x * 2; // a의 값은 _____
int b = y++ * 2; // b의 값은 _____
```

2. 다음 수식의 값을 쓰시오.

$12/5 - 3$   
 $5 + 19\%3$

3. 다음의 수식에서 연산의 순서를 적으시오.

(1)  $x = y = 3 / 5 * 2 \% 6;$

(2)  $y = a * x * x + b * x + c;$

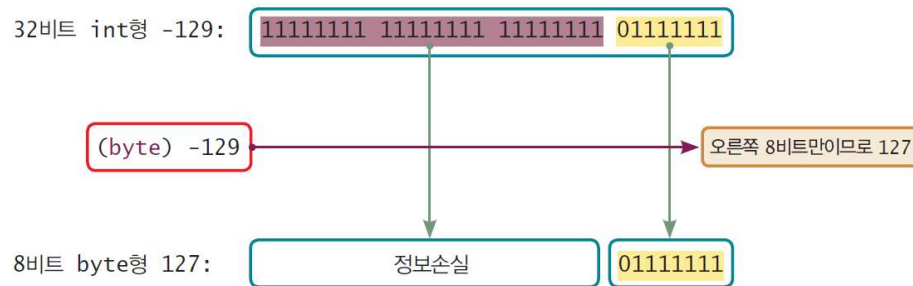
4. 변수 y, z, a, b의 값은?

```
int x = 0xff0f;
int y = x << 4;
int z = x >> 4;
int a = x & 0xf0ff;
int b = x | 0xf0ff;
```

# 형 변환(casting) 연산자

- 명시적 형 변환

- 실수를 정수로 변환하거나
- 범위가 큰 정수형에서 더 작은 정수형으로 변환하려면 명시적 형 변환(explicit type cast)이 필요



```
byte bt = (byte) -129;
```

- double에서 int로

- 자동 변환이 안되므로 오류발생
- 명시적 형변환이 필요

```
int n = 5.0 / 4.0;
```

Type mismatch: Cannot convert from **double** to int

자료형 불일치 오류 발생 !

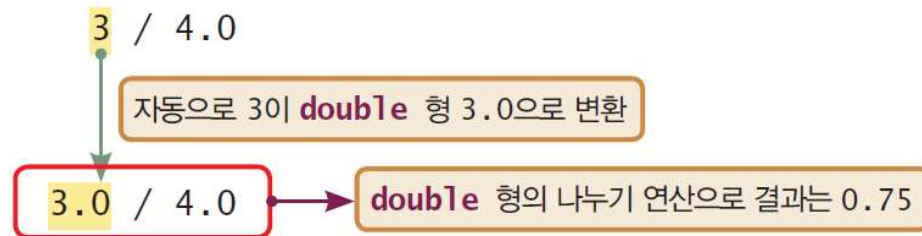
```
int n = (int) (5.0 / 4.0);
```

변수 n에 1이 저장

# 형변환 연산자

- 자동 형변환

- 자바 연산은 동일한 형의 피연산자로 연산을 수행
- 표현식  $3 / 4.0$ 
  - int 형 3이 자동으로 4.0인 double으로 변환
  - 표현 범위가 넓은 자료형으로 변환



# 형변환-Example

```
1 public class TypeCast {
2     public static void main (String args[] ) {
3         int i;
4         double f;
5         System.out.println("실행결과");
6         System.out.println("5/4 \t\t\t: "+(5/4));
7
8         f = 5/4;
9         System.out.println("f=5/4 \t\t\t: "+f);
10
11         /*서로 다른 자료형이 나올 경우 가장 큰 타입으로 자동 형변환*/
12
13         f = (double)5/4;
14         System.out.println("(double)5/4 \t\t: "+f);
15         f = 5/(double)4;
16         System.out.println("5/(double)4 \t\t: "+f);
17         f = (double)5/(double)4;
18         System.out.println("(double)5/(double)4 \t: "+f);
19         i = (int) 1.3 + (int) 1.8; //1+1
20         System.out.println("(int) 1.3 + (int) 1.8 \t: "+i);
21     }
22 }
```

실행결과	
5/4	: 1
f=5/4	: 1.0
<double>5/4	: 1.25
5/<double>4	: 1.25
<double>5/<double>4	: 1.25
<int> 1.3 + <int> 1.8	: 2

# 산술연산 주의점

- ArithmeticException이라는 예외
  - 정수를 0으로 나누면 실행 중에 발생
- Infinity
  - 실수인 0.0으로 나누면 무한대를 의미
- 표현식 0.0/0.0
  - NaN(Not a Number)가 출력
- 자료형 byte와 short의 산술 연산
  - 모두 int로 변환되어 연산을 수행

DevideByZero.java

```
01 package operator;
02
03 public class DevideByZero {
04     public static void main(String[] args) {
05         short data1 = 32766;
06         short data2 = 1;
07         //short data3 = data1 + data2; //오류발생
08         short data3 = (short)(data1 + data2);
09         short data4 = 32766 + 1;
10         System.out.println(data3 + " " + data4);
11
12         System.out.println(0.0 / 0.0); //NaN
13         System.out.println(3 / 0.0); //Infinity
14         System.out.println(3 / 0); //예외발생
15     }
16 }
```

더한 결과가 short의 범주인 -32768에서 32767 사이면 오류가 발생하지 않음.

결과 32767 32767

NaN

Infinity

Exception in thread "main" java.lang.ArithmeticException: / by zero  
at operator.DevideByZero.main(DevideByZero.java:14)

# APPENDIX

## Examples of Common Errors

# Common Errors and Pitfalls

- ➡ Common Error 1: Undeclared/Uninitialized Variables and Unused Variables
- ➡ Common Error 2: Integer Overflow
- ➡ Common Error 3: Round-off Errors
- ➡ Common Error 4: Unintended Integer Division
- ➡ Common Error 5: Redundant Input Objects
  
- ➡ Common Pitfall 1: Redundant Input Objects



# Common Error 1: Undeclared/Uninitialized Variables and Unused Variables

```
double interestRate = 0.05;
```

```
double interest = interestrate * 45;
```

# Common Error 2: Integer Overflow

```
int value = 2147483647 + 1;
```

```
// value will actually be -2147483648
```

# Common Error 3: Round-off Errors

```
System.out.println(1.0 - 0.1 - 0.1 - 0.1 - 0.1 - 0.1);
```

```
System.out.println(1.0 - 0.9);
```

# Common Error 4: Unintended Integer Division

```
int number1 = 1;  
int number2 = 2;  
double average = (number1 + number2) / 2;  
System.out.println(average);
```

(a)

```
int number1 = 1;  
int number2 = 2;  
double average = (number1 + number2) / 2.0;  
System.out.println(average);
```

(b)

# Common Pitfall 1: Redundant Input Objects

```
Scanner input = new Scanner(System.in);  
System.out.print("Enter an integer: ");  
int v1 = input.nextInt();
```

```
Scanner input1 = new Scanner(System.in);  
System.out.print("Enter a double value: ");  
double v2 = input1.nextDouble();
```

# LAB

- 어떤 도시가 메트로폴리스(거대도시)가 되려면 다음과 같은 2가지 조건 중의 하나를 만족하여야 한다고 가정하자.
  - ① 한 나라의 수도이고 인구가 100만 이상이어야 한다.
  - ② 연 소득이 1억 이상인 인구가 50만 이상이어야 한다.



## 실행결과

수도입니까?(수도: 1 수도아님: 0)1  
인구(단위: 백만)200  
부자의 수(단위: 백만)100  
메트로폴리스 여부: true

# Code for the exercise

```
/**
 * Check if a city is a metropolitan city
 */
import java.util.Scanner;          // Built-in Scanner class

public class Metropolis {
    public static void main (String args[] ) {
        boolean isCapital, isMetropolis;
        int citizen;
        int bourgeois;

        Scanner sc = new Scanner(System.in);
        System.out.print("It the city a capital? (capital:1 non-capital:0) ");
        isCapital = (sc.nextInt() == 1);
        System.out.print("Population? (in thousands) ");
        citizen = sc.nextInt();
        System.out.print("Bourgeois? (in thousands) ");
        bourgeois = sc.nextInt();

        isMetropolis = (isCapital && citizen >= 1000)
            || (bourgeois >= 500);

        System.out.println("Metropolis: " + isMetropolis);
    }
}
```

```
/* execution example
$ javac Metropolis.java
$ java Metropolis
It the city a capital? (capital:1 non-capital:0) 1
Population? (in thousands) 2000
Bourgeois? (in thousands) 1000
Metropolis: true
$ java Metropolis
It the city a capital? (capital:1 non-capital:0) 0
Population? (in thousands) 1000
Bourgeois? (in thousands) 500
Metropolis: true
*/
```

# 원 넓이 구하는 클래스 작성

- 클래스명 `CircleArea`
- 사용자로부터 반지름(실수) 입력받기
  - Scanner클래스의 `nextDouble()` 메서드 사용
  - 예) 

```
Scanner sc = new Scanner(System.in);  
double d = sc.nextDouble();  
int i = sc.nextInt();
```
- 원 넓이(실수) 계산 :  $\pi \times r^2$
- 원 넓이 출력하기
  - 소수점 아래 둘째 자리까지
  - `printf` 메서드 사용



# 원 넓이 구하는 클래스

```
1 import java.util.Scanner; //Scanner클래스를 사용할 것을 알림
2                             //사용자로부터 값을 입력받을 수 있도록
                             //도와주는 클래스
3 public class CircleArea{    //클래스 헤더 : 클래스 이름 알림
4                             //클래스 이름은 파일명과 일치시킴
5     public static void main(String args [] ){ //메인메서드 헤더
6         final float PI = 3.141592f; //상수 (final 키워드가
        //없으면 변수라서 PI=PI*2;와 같이 값을 바꿀 수 있지만
        //상수라서 바꿀수 없음)
7         double r; //8바이트 실수형으로 반지름 선언
8         double area; //넓이를 저장할 변수 선언
9         Scanner s = new Scanner(System.in); //스캐너 클래스 생성
10        System.out.print("반지름 : ");
11        r = s.nextDouble(); //사용자가 입력한 값을 반지름
        //변수에 대입
12        area = PI * r * r; //넓이 계산
13
14        System.out.printf("넓이 : %.2f cm^2\n", area); //넓이를
        //출력하는데 소수점 아래 2째 자리까지만 출력
15    }
16 }
```

## \* 난수 생성

- Math class: `java.lang.Math`
  - `random()`: 0이상 1미만의 임의의 double형 실수를 반환
  - 정수형 난수 생성은?? 별도의 함수 제공하지 않음
- Random class: `java.util.Random`
  - `nextDouble()`
  - `nextInt()`
  - `nextInt(int bound)`