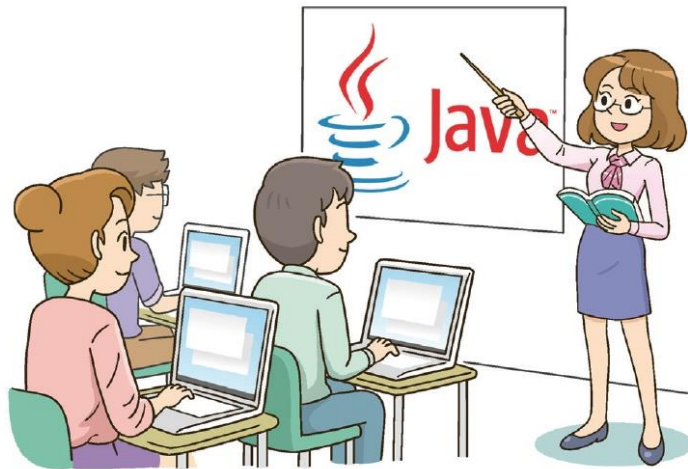


# 자바 프로그래밍



2주차  
자바의기본문법  
강의자 : 오 은 성

## 목차

- 자바프로그래밍의 기본 구조
- 식별자
- 자동타입변환 / 강제타입변환
- 자바에서 키 입력
- Scanner 이해하기

# 자바의 일반적인 구조

Add.java

```
01 public class Add {  
02  
03     public static void main(String[] args) {  
04         // 100과 200을 더하는 프로그램  
05         int x;  
06         int y;  
07         int sum;    // 합을 저장하는 변수  
08         x = 100;  
09         y = 200;  
10         sum = x + y;  
11         System.out.println("100과 200의 합="+sum);  
12     }  
13 }
```

클래스를 정의하는  
문장이다.

메소드를 정의하는  
문장이다.

실행결과

100과 200의 합=300

# 클래스

- ▶ 클래스(class)는 자바와 같은 객체 지향 언어의 기본적인 빌딩 블록
- ▶ 필요한 클래스를 하나씩 만들어 감으로써 전체 프로그램이 완성된다.



클래스는 자바 프로그램을  
이루는 기본적인  
빌딩블록 입니다.



# 클래스의 정의

이 클래스는 누구든지  
사용 가능

클래스를 선언하는 키워드

클래스 이름

클래스 시작

전체적인 구조



형식

```
public class Hello {
```

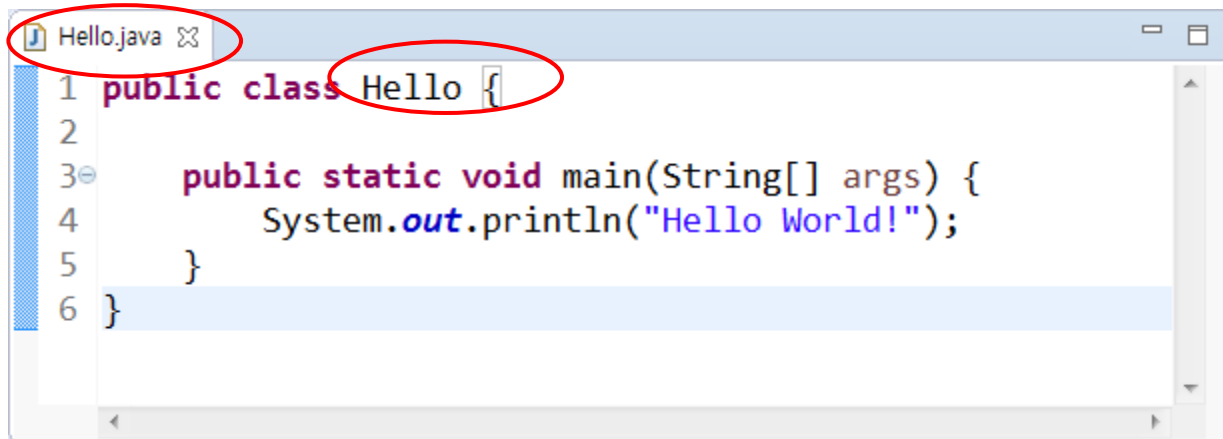
여기에 우리가 원하는  
문장을 추가한다.

클래스 종료

```
}
```

# 소스 파일과 클래스 이름

- ▶ 소스 안에 public 클래스가 있다면 반드시 소스 파일의 이름은 public 클래스의 이름과 일치하여야 한다.
- ▶ 하나의 소스 파일 안에 public 클래스가 2개 이상 있으면 컴파일 오류가 발생한다.



```
1 public class Hello {  
2  
3     public static void main(String[] args) {  
4         System.out.println("Hello World!");  
5     }  
6 }
```

# 주석

- ▶ `/* TEXT */`

- ▶ 주석의 시작과 끝을 `/*`와 `*/`로 표시

- ▶ `// TEXT`

- ▶ `//`에서 줄의 끝까지가 주석이다. 한 줄짜리 주석만 가능하다.

- ▶ `/** DOCUMENTATION */`

- ▶ `/**`에서 `*/`까지가 주석이 된다



Your comment  
here...

# 예제 2-1 자바프로그램의 기본 구조

8

```
/*  
* 소스 파일 : Hello.java  
*/
```

```
public class Hello {
```

```
    public static int sum(int n, int m) {  
        return n + m;  
    }
```

메소드

```
// main() 메소드에서 실행 시작
```

```
public static void main(String[] args) {  
    int i = 20;  
    int s;  
    char a;
```

```
    s = sum(i, 10); // sum() 메소드 호출
```

```
    a = '?';
```

```
    System.out.println(a); // 문자 '?' 화면 출력
```

```
    System.out.println("Hello"); // "Hello" 문자열 화면 출력
```

```
    System.out.println(s); // 정수 s 값 화면 출력
```

```
}
```

```
}
```

클래스

```
?  
Hello  
30
```

메소드



# 예제 2-1 코드 설명

9

## □ 클래스 만들기

- Hello 이름의 클래스 선언

```
public class Hello {  
}
```

- class 키워드로 클래스 선언
- public 선언하면 다른 클래스에서 접근 가능
- 클래스 코드는 {} 내에 모두 작성

## □ 주석문

- // 한 라인 주석
- /\* 여러 행 주석 \*/

## □ main() 메소드

- 자바 프로그램은 main()에서 실행 시작

```
public static void main(String[] args) {  
}
```

- public static void으로 선언
- String[] args로 실행 인자를 전달 받음

## □ 메소드

- C/C++에서의 함수를 메소드로 지칭

```
public static int sum(int n, int m) {  
    ...  
}
```

- 클래스 바깥에 작성할 수 없음

## □ 메소드 호출

- sum() 메소드 호출

```
int i = 20;  
s = sum(i, 10);
```

- sum() 호출 시 변수 i의 값과 정수 10을 전달
- sum()의 n, m에 각각 20, 10 값 전달
- sum()은 n과 m 값을 더한 30 리턴
- 변수 s는 정수 30을 전달받음

# 예제 2-1 설명 (계속)

10

## □ 변수 선언

- 변수 타입과 변수 이름 선언

```
int i=20;  
char a;
```

- 메소드 내에서 선언된 변수는 지역 변수

- 지역 변수는 메소드 실행이 끝나면 자동 소멸

## □ 문장

- ;로 한 문장의 끝을 인식

```
int i=20;  
s = sum(i, 20);
```

## □ 화면 출력

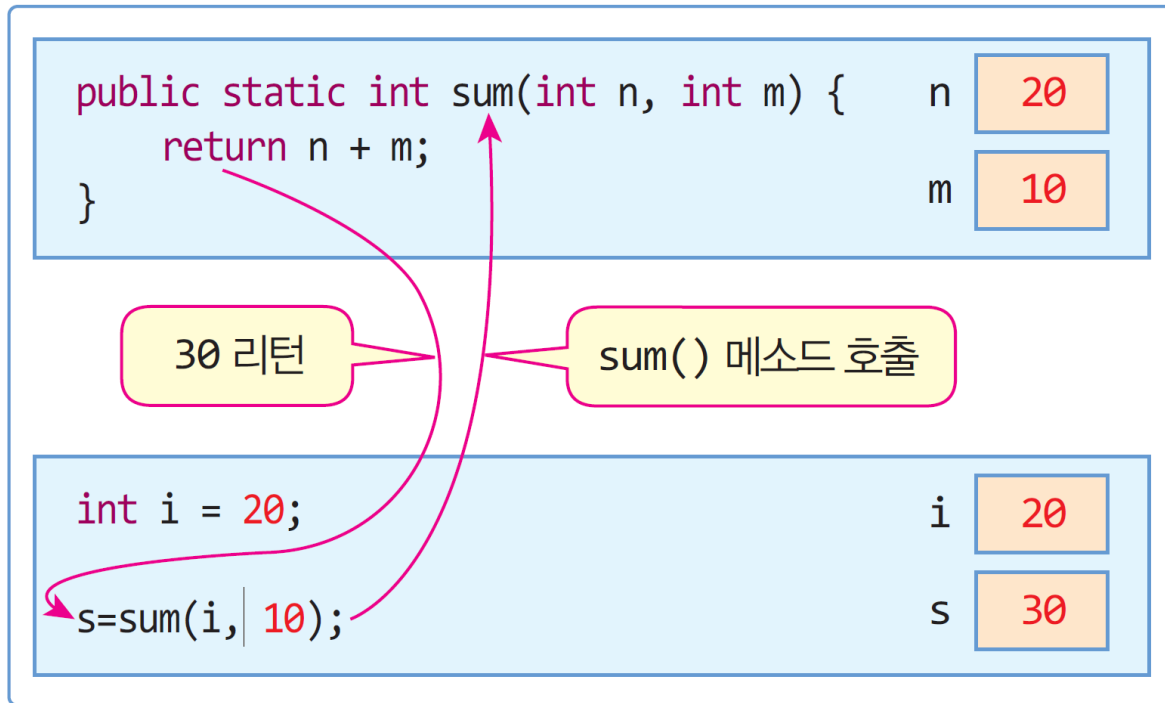
- 표준 출력 스트림에 메시지 출력

```
System.out.println("Hello"); // "Hello" 화면 출력
```

- 표준 출력 스트림 System.out의 println() 메소드 호출
- println()은 여러 타입의 데이터 출력 가능
- println()은 출력 후 다음 행으로 커서 이동

# sum() 메소드 호출과 리턴

11



클래스 Hello

# 식별자 (identifier)

12

- 식별자란?
  - ▣ 클래스, 변수, 상수, 메소드 등에 붙이는 이름
- 식별자의 원칙
  - ▣ '@', '#', '!'와 같은 특수 문자, 공백 또는 탭은 식별자로 사용할 수 없으나 '\_', '\$'는 사용 가능
  - ▣ 유니코드 문자 사용 가능. 한글 사용 가능
  - ▣ 자바 언어의 키워드는 식별자로 사용불가
  - ▣ 식별자의 첫 번째 문자로 숫자는 사용불가
  - ▣ '' 또는 '\$'를 식별자 첫 번째 문자로 사용할 수 있으나 일반적으로 잘 사용하지 않는다.
  - ▣ 불린 리터럴 (true, false)과 널 리터럴(null)은 식별자로 사용불가
  - ▣ 길이 제한 없음
- 대소문자 구별
  - ▣ Test와 test는 별개의 식별자

# 식별자 이름 사례

13

## □ 사용 가능한 예

```
int    name;
char   student_ID;           // '_' 사용 가능
void   $func() { }           // '$' 사용 가능
class  Monster3 { }          // 숫자 사용 가능
int     whatsyournamemynameiskitae; // 길이 제한 없음
int     barChart; int barchart; // 대소문자 구분. barChart와 barchart는 다름
int     가격;                 // 한글 이름 사용 가능
```

## □ 잘못된 예

```
int     3Chapter;             // 식별자의 첫문자로 숫자 사용 불가
class   if { }                 // 자바의 예약어 if 사용 불가
char    false;                 // false 사용 불가
void    null() { }             // null 사용 불가
class   %calc { }              // '%'는 특수문자
```

# 자바 키워드

14

abstract	continue	for	new	switch
assert	default	if	package	synchronized
boolean	do	goto	private	this
break	double	implements	protected	throw
byte	else	import	public	throws
case	enum	instanceof	return	transient
catch	extends	int	short	try
char	final	interface	static	void
class	finally	long	strictfp	volatile
const	float	native	super	while

# 좋은 이름 붙이는 언어 관습

15

- 기본 : 가독성 높은 이름
  - ▣ 목적을 나타내는 이름 붙이기 : s 보다 sum
  - ▣ 충분히 긴 이름으로 붙이기 : AVM보다 AutoVendingMachine
- 자바 언어의 이름 붙이는 관습 : 헝가리언 이름 붙이기
  - ▣ 클래스 이름
    - 첫 번째 문자는 대문자로 시작
    - 각 단어의 첫 번째 문자만 대문자
- 변수, 메소드 이름
  - 첫 단어 이후 각 단어의 첫 번째 문자는 대문자로 시작
- 상수 이름
  - 모든 문자를 대문자로 표시

```
public class HelloWorld { }  
class AutoVendingMachine { }
```

```
final static double PI = 3.141592;
```

```
int myAge;  
boolean isSingle;  
public int getAge() { }
```

# 문자열

16

- 문자열은 기본 타입이 아님
- String 클래스로 문자열 표현
  - ▣ 문자열 리터럴 - "JDK", "한글", "계속하세요"

```
String toolName="JDK";
```

- ▣ 문자열이 섞인 + 연산 -> 문자열 연결

```
toolName + 1.8          -> "JDK1.8"
 "(" + 3 + "," + 5 + ")" -> "(3,5)"
System.out.println(toolName + "이 출시됨"); // "JDK1.8이 출시됨" 출력
```



# 상수

## ▶ 상수 선언

### ▶ final 키워드 사용

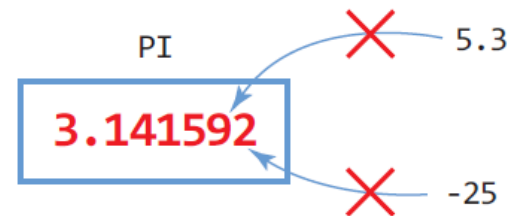
**final** **double** **PI** = **3.141592**;

상수 선언

데이터 타입

상수 이름

초기화



## 예제 2-2 : 변수, 리터럴, 상수 활용

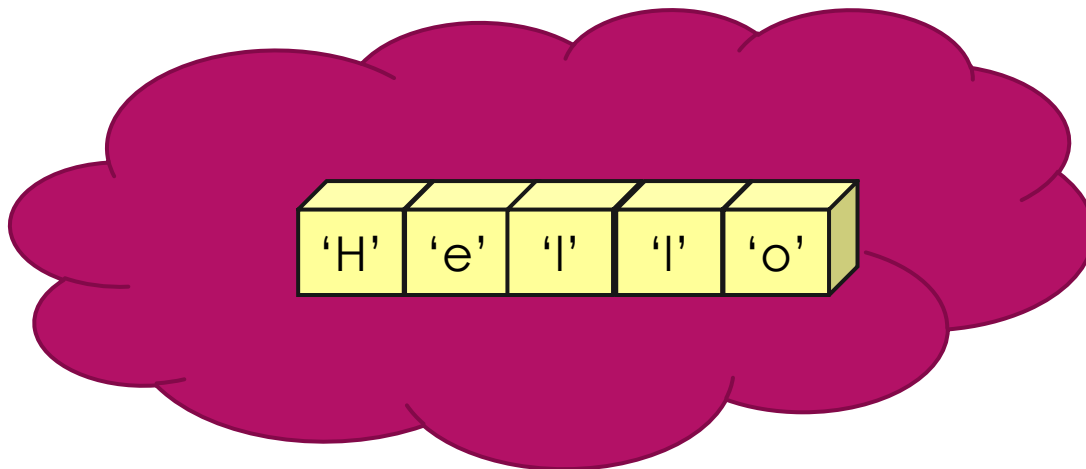
원의 면적을 구하는 프로그램을 작성해보자.

```
public class CircleArea {  
  
    public static void main(String[] args) {  
        final double PI = 3.14; // 원주율을 상수로 선언  
  
        double radius = 10.0; // 원의 반지름  
        double circleArea = radius*radius*PI; // 원의 면적 계산  
  
        // 원의 면적을 화면에 출력한다.  
        System.out.println("원의 면적 = " + circleArea);  
    }  
}
```

원의 면적 = 314.0

# 문자열

- ▶ 자바에서 문자열(**string**)은 문자들의 모임이다. 예를 들어서 문자열 "Hello"는 H, e, l, l, o 등의 5개의 유니코드 문자로 구성되어 있다.
- ▶ String 클래스가 제공된다.



## 예제2-3: 문자열 합치기

```
public class StringTest {  
    public static void main(String args[]) {  
        String s1 = "Hello World!";  
        String s2 = "I'm a new Java programmer!";  
  
        System.out.println(s1 + "\n" + s2);  
    }  
}
```

Hello World!

I'm a new Java programmer!

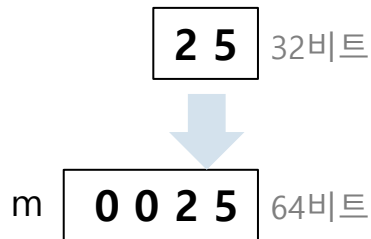
# 자동 타입 변환

21

## □ 자동 타입 변환

- ▣ 작은 타입은 큰 타입으로 자동 변환
  - 컴파일러에 의해 이루어짐
- ▣ 치환문(=)이나 수식 내에서 타입이 일치하지 않을 때

`long m = 25;` // 25는 int 타입. 25가 long 타입으로 자동 변환되는 사례



`double d = 3.14 * 10;` // 실수 연산을 하기 위해 10이 10.0으로 자동 변환  
// 다른 피연산자 3.14가 실수이기 때문

# 강제 타입 변환

22

- 자동 타입 변환이 안 되는 경우 : 큰 타입이 작은 타입으로 변환할 때

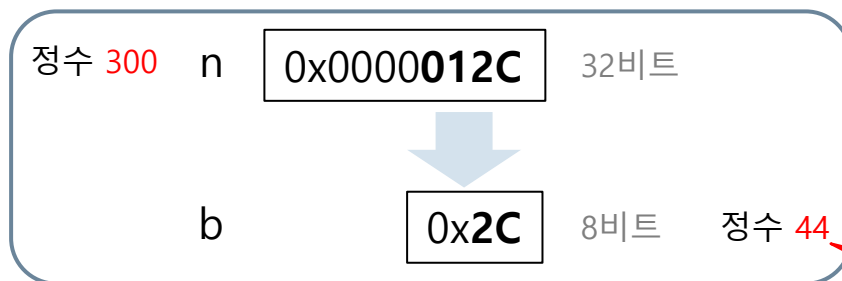
오류

```
int n = 300;  
byte b = n; // 컴파일 오류. int 타입이 byte로 자동 변환 안 됨
```

강제 타입 변환하려면, byte b = (byte)n; 로 수정

- 강제 타입 변환
  - ▣ 개발자가 필요하여 강제로 타입 변환을 지시
    - () 안에 변환할 타입 지정
  - ▣ 강제 변환은 값 손실 우려

byte b = (byte)n; 에 따른 손실



```
double d = 1.9;  
int n = (int)d; // n = 1
```

강제 타입 변환으로  
소숫점 이하 0.9 손실

$$44 = 300 \% 256$$

# 형 변환

```
int x = 3;  
double y = (double) x;
```



# 예제: 형 변환 예제

```
public class TypeConversion {  
    public static void main(String args[]) {  
        int i;  
        double f;  
  
        f = 5 / 4;  
        System.out.println(f);  
  
        f = (double) 5 / 4;  
        System.out.println(f);  
  
        i = (int) 1.3 + (int) 1.8;  
        System.out.println(i);  
    }  
}
```

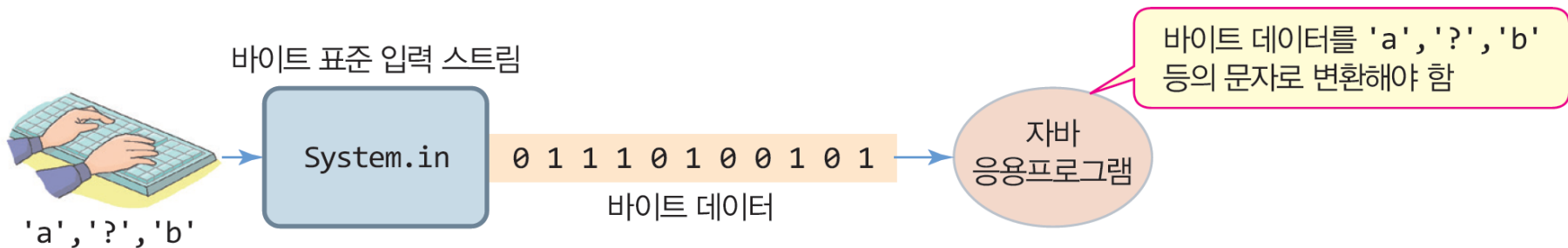
1.0  
1.25  
2



# 자바에서 키 입력

25

- System.in
  - ▣ 키보드로부터 직접 읽는 자바의 표준 입력 스트림
  - ▣ 키 값을 바이트(문자 아님)로 리턴
- System.in을 사용할 때 문제점
  - ▣ 키 값을 바이트 데이터로 넘겨주므로 응용프로그램이 문자 정보로 변환해야 함



# Scanner로 쉽게 키 입력

26

## □ Scanner 클래스

- System.in에게 키를 읽게 하고, 읽은 바이트를 문자, 정수, 실수, 불린, 문자열 등 다양한 타입으로 변환하여 리턴

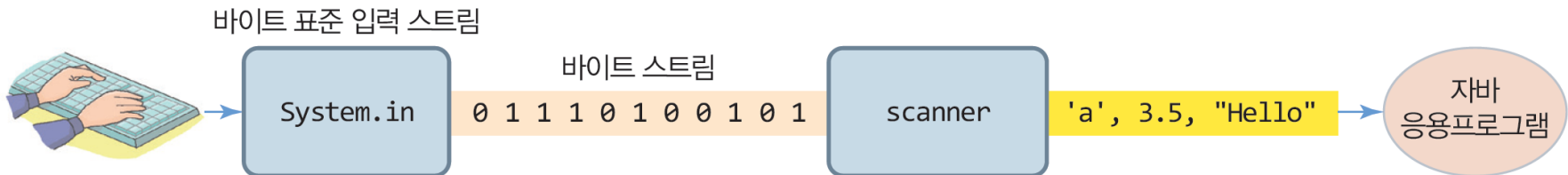
- java.util.Scanner 클래스

## □ 객체 생성

```
import java.util.Scanner; // import 문 필요
```

```
...
```

```
Scanner a = new Scanner(System.in); // Scanner 객체 생성
```

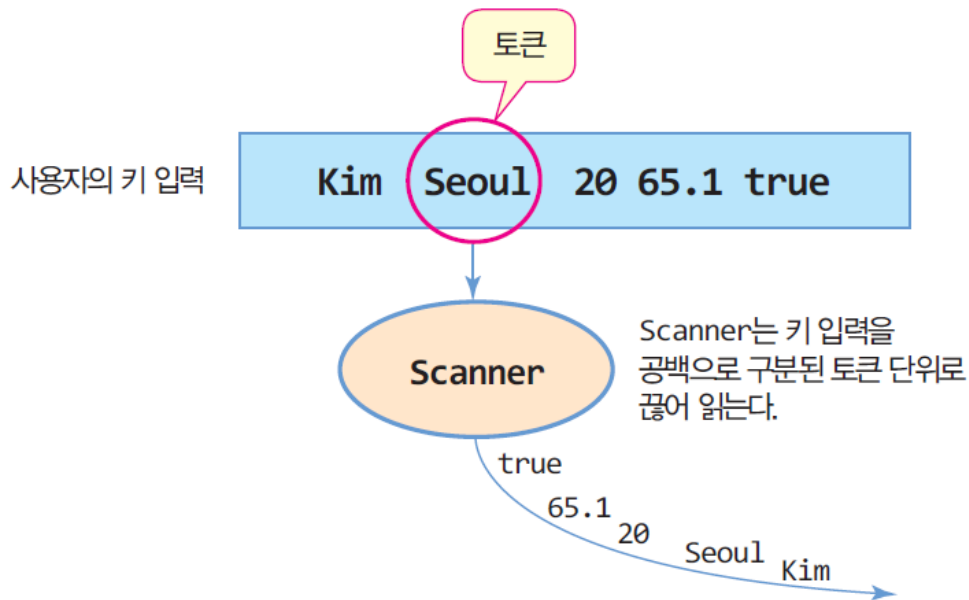


- System.in에게 키를 읽게 하고, 원하는 타입으로 변환하여 리턴

# Scanner를 이용한 키 입력

27

- Scanner에서 키 입력 받기
  - Scanner는 입력되는 키 값을 공백으로 구분되는 아이템 단위로 읽음
  - 공백 문자 : 'wt', 'wf', 'wr', ' ', 'wn'
- 개발자가 원하는 다양한 타입의 값으로 바꾸어 읽을 수 있음



```
Scanner scanner = new Scanner(System.in);

String name = scanner.next();           // "Kim"
String city = scanner.next();           // "Seoul"
int age = scanner.nextInt();            // 20
double weight = scanner.nextDouble();   // 65.1
boolean single = scanner.nextBoolean(); // true
```

# Scanner 주요 메소드

28

메소드	설명
<code>String next()</code>	다음 토큰을 문자열로 리턴
<code>byte nextByte()</code>	다음 토큰을 byte 타입으로 리턴
<code>short nextShort()</code>	다음 토큰을 short 타입으로 리턴
<code>int nextInt()</code>	다음 토큰을 int 타입으로 리턴
<code>long nextLong()</code>	다음 토큰을 long 타입으로 리턴
<code>float nextFloat()</code>	다음 토큰을 float 타입으로 리턴
<code>double nextDouble()</code>	다음 토큰을 double 타입으로 리턴
<code>boolean nextBoolean()</code>	다음 토큰을 boolean 타입으로 리턴
<code>String nextLine()</code>	'\n'을 포함하는 한 라인을 읽고 '\n'을 버린 나머지 문자열 리턴
<code>void close()</code>	Scanner의 사용 종료
<code>boolean hasNext()</code>	현재 입력된 토큰이 있으면 true, 아니면 입력 때까지 무한정 대기, 새로운 입력이 들어올 때 true 리턴. ctrl-z 키가 입력되면 입력 끝이므로 false 리턴

# 예제: 사용자로부터 받은 2개의 정수 더하기



첫번째 숫자를 입력하시오: 10  
두번째 숫자를 입력하시오: 20  
30



직접 입력  
하여 확인



## Add2.java

```
01 // 사용자가 입력한 두 개의 숫자를 더해서 출력한다.
02 import java.util.Scanner; // Scanner 클래스 포함
03
04 public class Add2 {
05     // 메인 메소드에서부터 실행이 시작된다.
06     public static void main(String args[]) {
07
08         Scanner input = new Scanner(System.in);
09         int x; // 첫 번째 숫자 저장
10         int y; // 두 번째 숫자 저장
11         int sum; // 합을 저장
12
13         System.out.print("첫번째 숫자를 입력하십시오: "); // 입력 안내 출력
14         x = input.nextInt(); // 사용자로부터 첫 번째 숫자를 읽는다.
15
16         System.out.print("두번째 숫자를 입력하십시오: "); // 입력 안내 출력
17         y = input.nextInt(); // 사용자로부터 두 번째 숫자를 읽는다.
18
19         sum = x + y; // 두 개의 숫자를 더한다.
20
21         System.out.println(sum); // 합을 출력한다.
22
23     }
24 }
```

## 예제 2-4 : Scanner를 이용한 키 입력 연습

31

Scanner를 이용하여  
이름, 도시, 나이, 체중,  
독신 여부를 입력 받고  
다시 출력하는  
프로그램을 작성하라.

```
import java.util.Scanner;

public class ScannerEx {
    public static void main(String args[]) {
        System.out.println("이름, 도시, 나이, 체중, 독신 여부를 빈칸으로 분리하여 입력하세요");
        Scanner scanner = new Scanner(System.in);

        String name = scanner.next(); // 문자열 읽기
        System.out.print("이름은 " + name + ", ");

        String city = scanner.next(); // 문자열 읽기
        System.out.print("도시는 " + city + ", ");

        int age = scanner.nextInt(); // 정수 읽기
        System.out.print("나이는 " + age + "살, ");

        double weight = scanner.nextDouble(); // 실수 읽기
        System.out.print("체중은 " + weight + "kg, ");

        boolean single = scanner.nextBoolean(); // 논리값 읽기
        System.out.println("독신 여부는 " + single + "입니다.");

    }
}
```

이름, 도시, 나이, 체중, 독신 여부를 빈칸으로 분리하여 입력하세요.

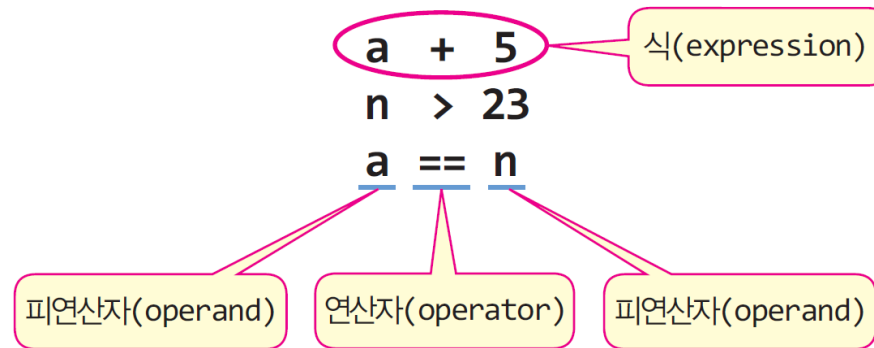
Kim Seoul 20 65.1 true

이름은 Kim, 도시는 Seoul, 나이는 20살, 체중은 65.1kg, 독신 여부는 true입니다.

# 식과 연산자

32

- 연산 : 주어진 식을 계산하여 결과를 얻어내는 과정



연산의 종류	연산자	연산의 종류	연산자
증감	<code>++ --</code>	비트	<code>&amp;   ^ ~</code>
산술	<code>+ - * / %</code>	논리	<code>&amp;&amp;    ! ^</code>
시프트	<code>&gt;&gt; &lt;&lt; &gt;&gt;&gt;</code>	조건	<code>? :</code>
비교	<code>&gt; &lt; &gt;= &lt;= == !=</code>	대입	<code>= *= /= += -= &amp;= ^=  = &lt;&lt;= &gt;&gt;= &gt;&gt;&gt;=</code>



# 연산자 우선순위

33

<div> <div>높음</div> <div>↓</div> <div>낮음</div> </div>	++(postfix) --(postfix)
	+(양수 부호) -(음수 부호) ++(prefix) --(prefix) ~ !
	형 변환(type casting)
	* / %
	+(덧셈) -(뺄셈)
	<< >> >>>
	<> <= >= instanceof
	== !=
	& (비트 AND)
	^ (비트 XOR)
	(비트 OR)
	&& (논리 AND)
	(논리 OR)
	? : (조건)
	= += -= *= /= %= &= ^=  = <<= >>= >>>=

- 같은 우선순위의 연산자
  - ▣ 왼쪽에서 오른쪽으로 처리
  - ▣ 예외)오른쪽에서 왼쪽으로
    - 대입 연산자, --, ++, +,-(양수 음수 부호), !, 형 변환은 오른쪽에서 왼쪽으로 처리
- 괄호는 최우선순위
  - ▣ 괄호가 다시 괄호를 포함한 경우는 가장 안쪽의 괄호부터 먼저 처리

# 산술 연산자

34

## □ 산술 연산자

- ▣ 더하기(+), 빼기(-), 곱하기(\*), 나누기(/), 나머지(%)
- ▣ /와 % 응용

- 10의 자리와 1의 자리 분리

$69/10 = 6$	← 몫 6
$69\%10 = 9$	← 나머지 9

- n이 홀수인지 판단

```
int r = n % 2; // r이 1이면 n은 홀수, 0이면 짝수
```

연산자	의미	예	결과
+	더하기	25.5 + 3.6	29.1
-	빼기	3 - 5	-2
*	곱하기	2.5 * 4.0	10.0
/	나누기	5/2	2
%	나머지	5%2	1

## 예제2: 두개의 정수를 입력 받아 사칙연산을 수행해 보자

```
import java.util.*;
public class InputString
{
    public static void main(String[] args)
    {
        int a,b,result;
        double result1;
        Scanner s = new Scanner(System.in);

        System.out.println("첫번째 정수를 입력하세요");
        a=s.nextInt();
        System.out.println("두번째 정수를 입력하세요");
        b=s.nextInt();
        result = a + b;
        System.out.println(a + "+" + b + "=" + result);
        result = a - b;
        System.out.println(a + "-" + b + "=" + result);
        result = a * b;
        System.out.println(a + "*" + b + "=" + result);
        result1 = (double)a / b;
        System.out.println(a + "/" + b + "=" + result1);
        result = a % b;
        System.out.println(a + "%" + b + "=" + result);
    }
}
```

# 중감 연산자

연산자	의미
$++x$	$x$ 값을 먼저 증가한 후에 다른 연산에 사용한다. 이 수식의 값은 증가된 $x$ 값이다.
$x++$	$x$ 값을 먼저 사용한 후에, 증가한다. 이 수식의 값은 증가되지 않은 원래의 $x$ 값이다.
$--x$	$x$ 값을 먼저 감소한 후에 다른 연산에 사용한다. 이 수식의 값은 감소된 $x$ 값이다.
$x--$	$x$ 값을 먼저 사용한 후에, 감소한다. 이 수식의 값은 감소되지 않은 원래의 $x$ 값이다.

# 예제: 증감 연산자

직접 입력  
하여 확인



## UnaryOperator.java

```
01 public class UnaryOperator {
02     public static void main(String[] args) {
03         int x = 1;
04         int y = 1;
05
06         int nextx = ++x; // x의 값이 사용되기 전에 증가된다. nextx는 2가 된다.
07         int nexty = y++; // y의 값이 사용된 후에 증가된다. nexty는 1이 된다.
08         System.out.println(nextx);
09         System.out.println(nexty);
10     }
11 }
```

실행결과



```
2
1
```

## 예제 2-5 : /와 % 산술 연산

38

초 단위의 정수를 입력받고, 몇 시간, 몇 분, 몇 초인지 출력하는 프로그램을 작성하라.

```
import java.util.Scanner;

public class ArithmeticOperator {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);

        System.out.print("정수를 입력하세요: ");
        int time = scanner.nextInt();    // 정수 입력
        int second = time % 60;        // 60으로 나눈 나머지는 초
        int minute = (time / 60) % 60;  // 60으로 나눈 몫을 다시 60으로 나눈 나머지는 분
        int hour = (time / 60) / 60;    // 60으로 나눈 몫을 다시 60으로 나눈 몫은 시간

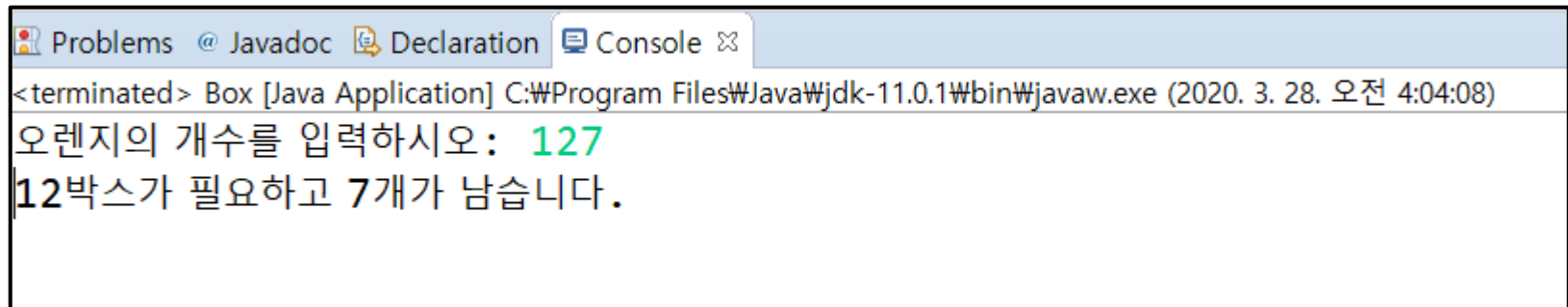
        System.out.print(time + "초는 ");
        System.out.print(hour + "시간, ");
        System.out.print(minute + "분, ");
        System.out.println(second + "초입니다.");

    }
}
```

정수를 입력하세요:5000  
5000초는 1시간, 23분, 20초입니다.

# 확인학습

1. 확인학습
2. 하나의 상자에 오렌지를 10개씩 담을 수 있다고 하자. 오렌지가 127개가 있다면 상자 몇 개가 필요한가? 또 몇 개의 오렌지가 남을 까?
3. 아래와 같이 출력하세요



```
Problems @ Javadoc Declaration Console
<terminated> Box [Java Application] C:\Program Files\Java\jdk-11.0.1\bin\javaw.exe (2020. 3. 28. 오전 4:04:08)
오렌지의 개수를 입력하시오: 127
12박스가 필요하고 7개가 남습니다.
```

# 관계 연산자

표 2-3 • 관계 연산자

연산자 기호	의미	사용예
==	x와 y가 같은가?	$x == y$
!=	x와 y가 다른가?	$x != y$
>	x가 y보다 큰가?	$x > y$
<	x가 y보다 작은가?	$x < y$
>=	x가 y보다 크거나 같은가?	$x >= y$
<=	x가 y보다 작거나 같은가?	$x <= y$



# 예제: 관계 연산자

직접 입력  
하여 확인



## ComparisonOperator.java

```
01 public class ComparisonOperator {  
02  
03     public static void main(String[] args){  
04         int x = 3;  
05         int y = 4;  
06         System.out.print((x == y) + " ");  
07         System.out.print((x != y) + " ");  
08         System.out.print((x > y) + " ");  
09         System.out.print((x < y) + " ");  
10         System.out.print((x <= y) + " ");  
11     }  
12 }
```

실행결과



false true false true true

# 논리 연산자

연산자 기호	사용예	의미
&&	x && y	AND 연산, x와 y가 모두 참이면 참, 그렇지 않으면 거짓
	x    y	OR 연산, x나 y중에서 하나만 참이면 참, 모두 거짓이면 거짓
!	!x	NOT 연산, x가 참이면 거짓, x가 거짓이면 참

# 예제: 논리 연산자

```
public class LogicalOperator {  
  
    public static void main(String[] args){  
        System.out.println((3 == 3) && (4 == 7)) ;  
        System.out.println((3 == 3 || 4 == 7)) ;  
    }  
}
```

false  
true

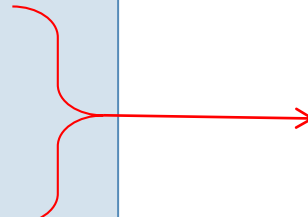
# 조건 연산자 ?:

44

- `condition ? opr2 : opr3`
  - ▣ 세 개의 피연산자로 구성된 삼항(ternary) 연산자
    - `condition`이 `true`이면, 연산식의 결과는 `opr2`, `false`이면 `opr3`
  - ▣ `if-else`을 간결하게 표현할 수 있음

```
int x = 5;
int y = 3;

int s;
if(x>y)
    s = 1;
else
    s = -1;
```



```
int s = (x>y)?1:-1;
```

## 예제 2-8 : 조건 연산

45

다음은 조건 연산자의 사례이다. 실행 결과는 무엇인가?

```
public class TernaryOperator {  
    public static void main (String[] args) {  
        int a = 3, b = 5;  
  
        System.out.println("두 수의 차는 " + ((a>b)?(a-b):(b-a)));  
    }  
}
```

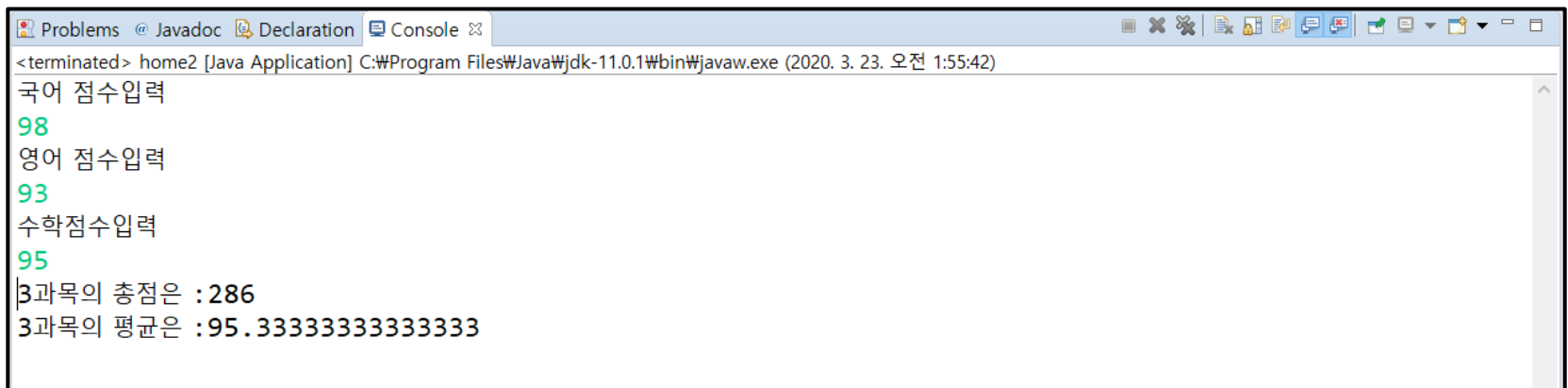
두 수의 차는 2

# 확인학습

1. 당신의 나이를 입력 받아 성년인지 미성년자인지 출력해주세요(단 조건연산자를 사용하여 프로그램 작성 19살이상이면 성년 그렇지 않으면 미성년 출력)

# 과제

1. 2주차\_수업 확인학습
2. 국어 영어 수학 점수를 입력 받아 변수에 저장하고 합계와 평균을 구해 출력해 주세요 반드시 평균은 소수점으로 형 변환 하기
3. 아래와 같이 출력하고 캡처하여 한글파일에 붙여 놓고 학번이름.hwp로 저장 후 LMS 과제란에 제출해 주세요



```
<terminated> home2 [Java Application] C:\Program Files\Java\jdk-11.0.1\bin\javaw.exe (2020. 3. 23. 오전 1:55:42)
국어 점수입력
98
영어 점수입력
93
수학점수입력
95
3과목의 총점은 : 286
3과목의 평균은 : 95.33333333333333
```