

2025년 상반기 K-디지털 트레이닝

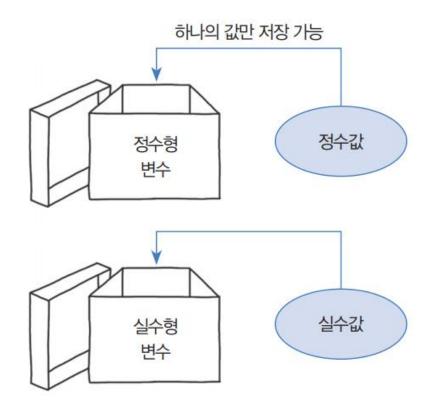
# 변수와 타입

[KB] IT's Your Life



#### 변수 variable

- 하나의 값을 저장할 수 있는 메모리 번지에 붙여진 이름
- o 자바의 변수는 한 가지 타입의 데이터만 저장 가능
  - 다양한 타입(정수형, 실수형 등)의 값을 저장할 수 없음



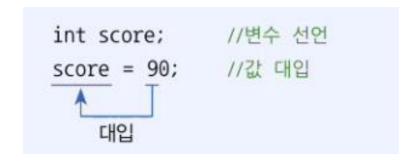
## ☑ 변수 선언

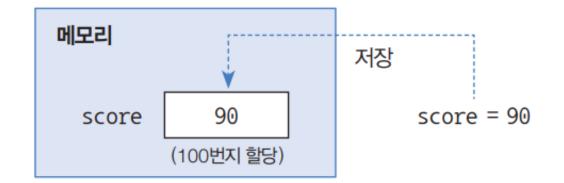
- 변수 변수를 사용하려면 변수 선언이 필요.
- 변수 선언은 저장할 데이터의 타입과, 변수 이름을 결정하는 것



#### 🕜 변수 선언

- 변수의 초기화
  - 변수에 최초로 값이 대입될 때 메모리에 할당 되고, 해당 메모리에 값이 저장 → 초기화





■ 선언과 초기 값 대입을 동시에 표현 가능

```
int <u>score</u> = 90;
대입
```

#### 변수 선언

- 변수의 초기화
  - 초기화 되지 않은 변수를 사용(읽기)하면 컴파일 에러

```
① int value;
                         //변수 value 선언
② int result = value + 10; //변수 value 값을 읽고 10을 더해서 변수 result에 저장
int value = 30;
             //변수 value가 30으로 초기화됨
int result = value + 10; //변수 value 값(30)을 읽고 10을 더해서 변수 result에 저장
```

## ch02.sec01.VariableInitializationExample.java

```
package ch02.sec01;
public class VariableInitializationExample {
 public static void main(String[] args) {
  //변수 value 선언
   int value;
   //연산 결과를 변수 result의 초기값으로 대입
   int result = value + 10; //에러
   //변수 result 값을 읽고 콘솔에 출력
   System.out.println(result); //에러
```

## ch02.sec01.VariableUseExample.java

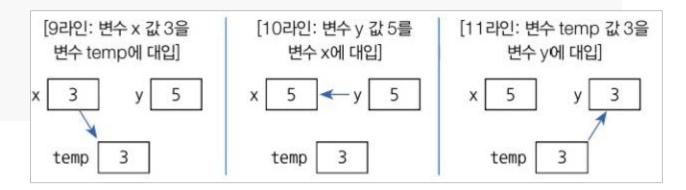
```
public class VariableUseExample {
  public static void main(String[] args) {
    int hour = 3;
    int minute = 5;
    System.out.println(hour + "시간 " + minute + "분");

  int totalMinute = (hour * 60) + minute;
    System.out.println("총 " + totalMinute + "분");
}
```

```
3시간 5분
총 185분
```

## ch02.sec01.VariableExchangeExample.java

```
package ch02.sec01;
public class VariableExchangeExample {
 public static void main(String[] args) {
   int x = 3;
   int y = 5;
   System.out.println("x:" + x + ", y:" + y);
   int temp = x;
   x = y;
   y = temp;
   System.out.println("x:" + x + ", y:" + y);
x:3, y:5
x:5, y:3
```



#### 🧿 자바의 데이터 타입

○ 기본 탁입 primitive type

값의 분류	기본 타입
정수	byte, char, short, int, long
실수	float, double
논리(true/false)	boolean

- 변수는 선언될 때의 타입에 따라 저장할 수 있는 값의 종류와 허용 범위가 달라짐
  - → 변수 선언이 필요한 이유

## 2 정수 타입

## 🛮 byte, short, char, int, long 타입

○ 정수 타입별 값의 범위

호환용 하위 혹은 다른 언어와의

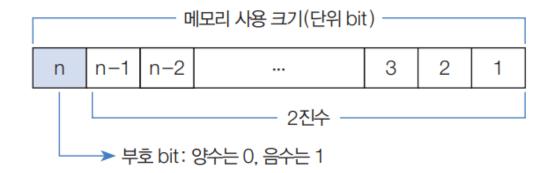
표준

종류	byte	short	int	long	
메모리 사용 크기(단위 bit)	8	16	32	64	

	타입	메모리 크기		저장되는 값의 허용 범위	
	byte	1byte*	8bit	$-2^{7} \sim (2^{7}-1)$	<b>−128</b> ~ <b>127</b>
1	short	2byte	16bit	$-2^{15} \sim (2^{15}-1)$	−32,768 ~ 32,767
	char	2byte	16bit	$0 \sim (2^{16}-1)$	0 ~ 65535 (유니코드)
	int	4byte	32bit	$-2^{31} \sim (2^{31}-1)$	-2,147,483,648 ~ 2,147,483,647
	long	8byte	64bit	$-2^{63} \sim (2^{63} - 1)$	-9,223,372,036,854,775,808 ~ 9,223,372,036,854,775,807

<sup>\* 1</sup>byte = 8bit, bit는 0과 1이 저장되는 단위

- o byte, short, char, int, long 타입
  - 메모리 크기를 n이라고 했을 때 정수 타입은 동일한 구조의 2진수로 저장

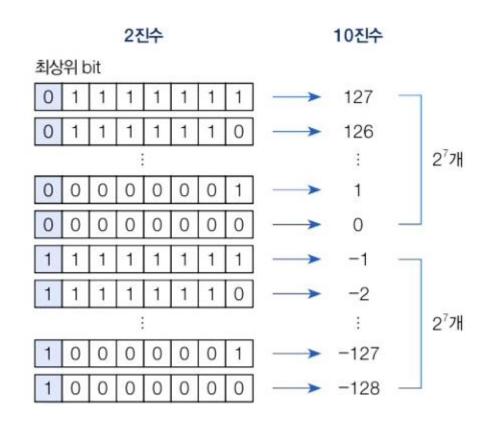


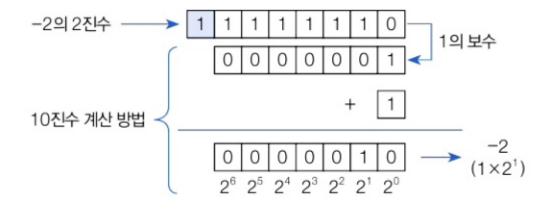
최상위 bit 값	값의 범위
0 n-1 개의 bit	$0 \sim (2^{n-1} - 1)$
1 n-1 개의 bit	$-2^{n-1} \sim -1$

## 2 <mark>정수 타입</mark>

## 🗸 byte, short, char, int, long 타입

- 정수 표현 방법
  - 최상위 비트는 부호( sign) 비트
  - 음수는 2의 보수로 표현





#### ☑ 정수 리터럴 표기법

2진수: 0b 또는 0B로 시작하고 0과 1로 작성

```
int x = 0b1011;  //10진수 값 = 1x2^3 + 0x2^2 + 1x2^1 + 1x2^0 = 11
int y = 0B10100;  //10진수 값 = 1x2^4 + 0x2^3 + 1x2^2 + 0x2^1 + 0x2^0 = 20
```

8진수: 0으로 시작하고 0~7 숫자로 작성

```
int x = 013;  //10진수 값 = 1x8^1 + 3x8^0 = 11
int y = 0206;  //10진수 값 = 2x8^2 + 0x8^1 + 6x8^0 = 134
```

#### 💟 정수 리터럴 표기법

10진수: 소수점이 없는 0~9 숫자로 작성

```
int x = 12;
int y = 365;
```

• 16진수: 0x 또는 0X로 시작하고 0~9 숫자나 A, B, C, D, E, F 또는 a, b, c, d, e, f로 작성

```
int x = 0xB3;  //10진수 값 = 11x16^1 + 3x16^0 = 179
int y = 0x2A0F;  //10진수 값 = 2x16^3 + 10x16^2 + 0x16^1 + 15x16^0 = 10767
```

## ch02.sec02.IntegerLiteralExample.java

```
package ch02.sec02;
public class IntegerLiteralExample {
 public static void main(String[] args) {
   int var1 = 0b1011; //2진수
   int var2 = 0206; //8진수
   int var3 = 365; //10진수
   int var4 = 0xB3; //16진수
   System.out.println("var1: " + var1);
   System.out.println("var2: " + var2);
   System.out.println("var3: " + var3);
   System.out.println("var4: " + var4);
```

```
var1: 11
var2: 134
var3: 365
var4: 179
```

## ch02.sec02.ByteExample.java

```
package ch02.sec02;
public class ByteExample {
 public static void main(String[] args) {
   byte var1 = -128;
   byte var2 = -30;
   byte var3 = 0;
   byte var4 = 30;
   byte var5 = 127;
   //byte var6 = 128; //컴파일 에러(Type mismatch: cannot convert from int byte)
   System.out.println(var1);
   System.out.println(var2);
   System.out.println(var3);
   System.out.println(var4);
                                      -128
   System.out.println(var5);
                                      -30
                                      30
                                      127
```

## ch02.sec02.LongExample.java

```
package ch02.sec02;
public class LongExample {
 public static void main(String[] args) {
   long var1 = 10;
   long var2 = 20L;
   //long var3 = 1000000000000; //컴파일러는 int로 간주하기 때문에 에러 발생
   long var4 = 1000000000000L;
   System.out.println(var1);
   System.out.println(var2);
   System.out.println(var4);
10
20
10000000000000
```

#### 문자 리터럴과 char 타입

- 문자 리터럴: 하나의 문자를 작은 따옴표(')로 감싼 것
- 문자 리터럴을 유니코드로 저장할 수 있도록 char 타입 제공

```
char var1 = 'A'; //'A' 문자와 매핑되는 숫자: 65로 대입
char var3 = '가'; //'가' 문자와 매핑되는 숫자: 44032로 대입
```

#### o char 타입도 정수 타입에 속함

```
char c = 65; //10진수 65와 매핑되는 문자: 'A'
char c = 0x0041; //16진수 0x0041과 매핑되는 문자: 'A'
```

자바는 기본으로 unicode utf-8을 활용함 영어는 1바이트, 한글은 3바이트.

### 문자 타입

## ch02.sec03.CharExample.java

```
package ch02.sec03;
public class CharExample {
 public static void main(String[] args) {
   char c1 = 'A'; //문자 저장
char c2 = 65; //유니코드 직접 저장
   char c3 = '가'; //문자 저장
   char c4 = 44032; //유니코드 직접 저장
   System.out.println(c1);
   System.out.println(c2);
   System.out.println(c3);
                                                char c = ''; //컴파일 에러
   System.out.println(c4);
                                                char c = ' '; //공백 하나를 포함해서 초기화
```

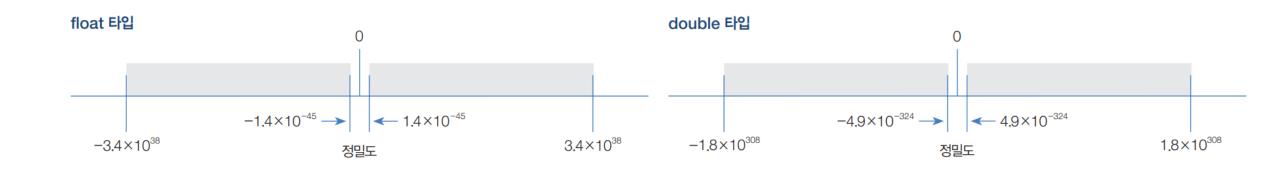
```
A
A
가
가
```

### float과 double 타입

○ 실수 타입에는 float과 double이 있음

타입	메모리 크기		저장되는 값의 허용 범위(양수 기준)	유효 소수 이하 자리
float	4 byte	32 bit	$1.4 \times 10^{-45} \sim 3.4 \times 10^{38}$	7자리
double	8 byte	64 bit	$4.9 \times 10^{-324} \sim 1.8 \times 10^{308}$	15자리

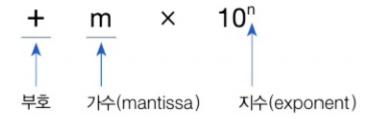
#### o double 타입이 float 타입보다 큰 실수를 저장할 수 있고 정밀도도 높음



### 4 실수 타입

### ▽ 부동 소수점 float-point

○ 실수를 나타내는 국제 표준 IEEE754



[float] 부호(1bit) + 지수(8bit) + 가수(23bit) = 32bit = 4byte

1bit 지수(8bit)	가수(23bit)
---------------	-----------

[double] 부호(1bit) + 지수(11bit) + 가수(52bit) = 64bit = 8byte

1bit	지수(11bit)	가수(52bit)
------	-----------	-----------

#### ☑ 실수 리터럴

• 10진수 리터럴

```
double x = 0.25;
double y = -3.14;
```

• e 또는 E가 포함된 10의 거듭제곱 리터럴

```
double x = 5e2; //5.0 \times 10^2 = 500.0
double y = 0.12E-2 //0.12 \times 10^{-2} = 0.0012
```

#### 실수 리터럴의 기본 타입은 double

```
double var = 3.14;
double var = 314e-2;
```

```
float var = 3.14f;
float var = 3E6F;
```

## ch02.sec04.FloatDoubleExample.java

```
package ch02.sec04;
public class FloatDoubleExample {
 public static void main(String[] args) {
   //정밀도 확인
   float var1 = 0.1234567890123456789f;
   double var2 = 0.1234567890123456789;
   System.out.println("var1: " + var1);
   System.out.println("var2: " + var2);
   //10의 거듭제곱 리터럴
   double var3 = 3e6;
   float var4 = 3e6F;
   double var5 = 2e-3;
   System.out.println("var3: " + var3);
                                         var1: 0.12345679
                                                                                   double 타입이 float 타입보다 약 2배
   System.out.println("var4: " + var4);
                                         var2: 0.12345678901234568 •-----
   System.out.println("var5: " + var5);
                                                                                   정도의 유효 자릿수를 가진다.
                                         var3: 3000000.0
                                         var4: 3000000 0
                                         var5: 0.002
```

#### 🥝 boolean 타입 변수에 대입되는 논리 타입

○ 참과 거짓을 의미하는 true와 false로 구성되며 boolean 타입 변수에 대입할 수 있음

```
boolean stop = true;
boolean stop = false;
```

○ 주로 두 가지 상태값을 저장하는 경우에 사용. 조건문과 제어문의 실행 흐름을 변경하는 데 사용

```
int x = 10; 연산식 boolean result = (x == 20); //변수 x의 값이 20인가? boolean result = (x != 20); //변수 x의 값이 20이 아닌가? boolean result = (x > 20); //변수 x의 값이 20보다 큰가? boolean result = (0 < x & x < 20); //변수 x의 값이 0보다 크고, 20보다 적은가? boolean result = (x < 0 | x > 200); //변수 x의 값이 0보다 적거나 200보다 큰가?
```

## ch02.sec05.BooleanExample.java

```
package ch02.sec05;
public class BooleanExample {
 public static void main(String[] args) {
   boolean stop = true;
   if(stop) {
    System.out.println("중지합니다.");
   } else {
    System.out.println("시작합니다.");
   int x = 10;
   boolean result1 = (x == 20); //변수 x의 값이 20인가?
   boolean result2 = (x != 20); //변수 x의 값이 20이 아닌가?
   System.out.println("result1: " + result1);
   System.out.println("result2: " + result2);
```

중지합니다. result1: false result2: true

//컴파일 에러

//컴파일 에러

char var1 = "A";

char var2 = "홍길동";

## 문자열과 String 타입

- 문자열: 큰따옴표("")로 감싼 문자들
- 문자열을 변수에 저장하려면 String 타입을 사용

```
String var1 = "A";
String var2 = "홍길동";
```

○ 이스케이프 문자: 문자열 내부에 역슬래쉬(₩)가 붙은 문자

이스케이프 문자	
\"	"문자 포함
\'	'문자 포함
\\	∖ 문자 포함
\u16진수	16진수 유니코드에 해당하는 문자 포함
\t	출력 시 탭만큼 띄움
\n	출력 시 줄바꿈(라인피드)
\r	출력 시 캐리지 리턴

## ch02.sec06.StringExample.java

```
package ch02.sec06;
public class StringExample {
 public static void main(String[] args) {
   String name = "홍길동";
   String job = "프로그래머";
   System.out.println(name);
   System.out.println(job);
   String str = "나는 \"자바\"를 배웁니다.";
   System.out.println(str);
   str = "번호\t이름\t직업 ";
   System.out.println(str);
                                        홍길동
   System.out.print("나는\n");
                                        프로그래머
   System.out.print("자바를\n");
                                        나는 "자바"를 배웁니다.
   System.out.print("배웁니다.");
                                        번호
                                                이름
                                                         직업
                                        나는
                                        자바를
                                        배웁니다.
```

### ◎ 텍스트 블록 문법

o Java 13부터 지원

```
String str = """
...
""";
```

## ch02.sec06.TextBlockExample.java

System.out.println("-----");

```
package ch02.sec06;
public class TextBlockExample {
 public static void main(String[] args) {
   String str1 = "" +
   "{\n" +
   "\t\"id\":\"winter\",\n" +
   "\t\"name\":\"눈송이\"\n" +
                                                             "id":"winter",
                                                             "name":"눈송이"
   String str2 = """
                                  삼중따옴표로 심플하게
    "id":"winter",
                                                             "id":"winter",
                                                             "name":"눈송이"
   1111111
   System.out.println(str1);
   System.out.println("-----
   System.out.println(str2);
```

## ch02.sec06.TextBlockExample.java

```
String str = """
나는 자바를 \
학습합니다.
나는 자바 고수가 될 겁니다.
System.out.println(str);
```

나는 자바를 학습합니다. 나는 자바 고수가 될 겁니다.

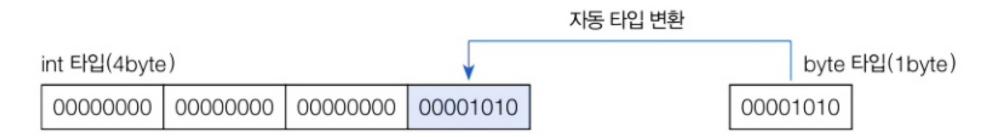
## 7 자동 타입 변환

#### 💟 자동 타입 변환

- 데이터 타입을 다른 타입으로 변환하는 것
- 값의 허용 범위가 작은 타입이 허용 범위가 큰 타입으로 대입될 때 발생



```
byte byteValue = 10;
int intValue = byteValue; //자동 타입 변환됨
```



#### 💟 자동 타입 변환

정수 타입이 실수 타입으로 대입--> 무조건 자동 타입 변환이 됨

```
long longValue = 5000000000L;
float floatValue = longValue; //5.0E9f로 저장됨
double doubleValue = longValue; //5.0E9로 저장됨
```

o char 타입이 int 타입으로 변환될 때 유니 코드 값이 대입

```
char charValue = 'A';
int intValue = charValue; //65가 저장됨
```

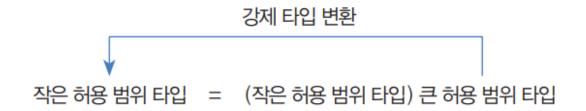
## ch02.sec07.PromotionExample.java

```
public class PromotionExample {
 public static void main(String[] args) {
   //자동 타입 변환
   byte byteValue = 10;
   int intValue = byteValue;
   System.out.println("intValue: " + intValue);
   char charValue = '가';
   intValue = charValue;
   System.out.println("가의 유니코드: " + intValue);
   intValue = 50;
   long longValue = intValue;
   System.out.println("longValue: " + longValue);
                                                        intValue: 10
   longValue = 100;
   float floatValue = longValue;
                                                        가의 유니코드: 44032
   System.out.println("floatValue: " + floatValue);
                                                        longValue: 50
                                                        floatValue: 100.0
   floatValue = 100.5F;
                                                        doubleValue: 100.5
   double doubleValue = floatValue;
   System.out.println("doubleValue: " + doubleValue);
```

#### 8

#### 💟 캐스팅 연산자로 강제 타입 변환하기

- 큰 허용 범위 타입을 작은 허용 범위 타입으로 쪼개어서 저장하는 것
- 캐스팅 연산자로 괄호()를 사용하며, 괄호 안에 들어가는 타입은 쪼개는 단위

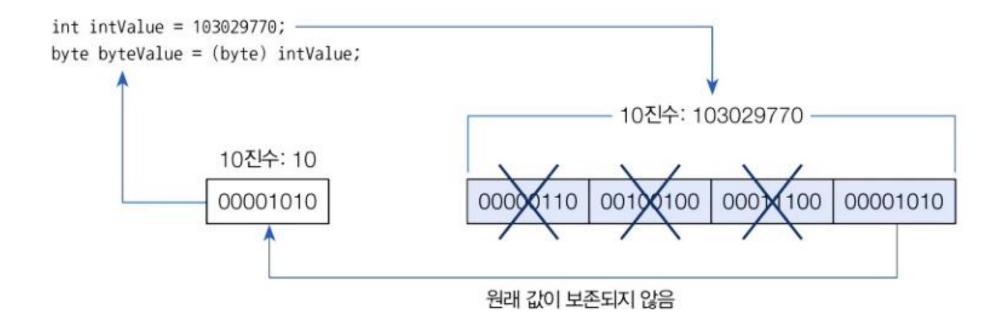


 $\circ$  int  $\rightarrow$  byte

```
int intValue = 10;
byte byteValue = (byte) intValue;
(1byte)
byte byteValue = (byte) intValue;
```

## ☑ 캐스팅 연산자로 강제 타입 변환하기

- $\circ$  int  $\rightarrow$  byte
  - 원래 값이 보존되지 않는 경우



#### 💟 캐스팅 연산자로 강제 타입 변환하기

 $\circ$  long  $\rightarrow$  int

```
long longValue = 300;
int intValue = (int) longValue; //강제 타입 변환 후에 300이 그대로 유지
```

 $\circ$  int  $\rightarrow$  char

```
int intValue = 65;
char charValue = (char) intValue;
System.out.println(charValue); //'A'가 출력
```

○ 실수 → 정수

```
double doubleValue = 3.14;
int intValue = (int) doubleValue; //intValue는 정수 부분인 3만 저장
```

# ch02.sec08.CastingExample.java

```
package ch02.sec08;
public class CastingExample {
 public static void main(String[] args) {
   int var1 = 10;
   byte var2 = (byte) var1;
   System.out.println(var2); //강제 타입 변환 후에 10이 그대로 유지
   long var3 = 300;
   int var4 = (int) var3;
   System.out.println(var4); //강제 타입 변환 후에 300이 그대로 유지
   int var5 = 65;
   char var6 = (char) var5;
                                               10
   System.out.println(var6); //'A'가 출력
                                                300
                                                Α
   double var7 = 3.14;
   int var8 = (int) var7;
   System.out.println(var8); //3이 출력
```

## 9 연산식에서 자동 타입 변환

#### ◎ 연산식에서 int 타입의 자동 변환

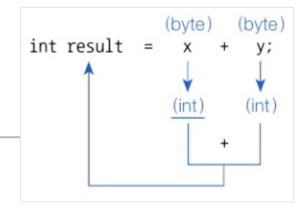
○ 정수 타입 변수가 산술 연산식에서 피연산자로 사용되면
→ int 타입보다 작은 byte, short 타입 변수는 int 타입으로 자동 변환되어 연산 수행

```
int result = byte 타입 char 타입 short 타입 int 타입 byte 타입 char 타입 int 타입
```

#### byte 타입 변수가 피연산자로 사용된 경우

```
byte x = 10;
byte y = 20;
byte result = x + y; //컴파일 에러
int result = x + y;
```

#### int 타입 변수가 피연산자로 사용된 경우



## 9 연산식에서 자동 타입 변환

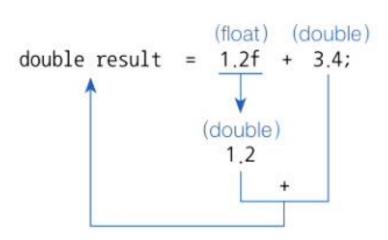
- 연산식에서 long 타입의 자동 변환
  - 다른 타입의 피연산자들이 long 타입으로 변환되어 계산



- 💟 연산식에서 실수 타입의 자동 변환
  - o float 만 있는 경우 float로 계산

```
float result = 1.2f + 3.4f; //컴파일: float result = 4.6f;
```

o double이 있는 경우 모두 double로 자동 변환 후 계산



#### 💟 연산식에서 실수 타입의 자동 변환

o int 타입과 double 타입이 연산에 있는 경우 → double 타입으로 자동변환

```
int intValue = 10;
double doubleValue = 5.5;
double result = intValue + doubleValue; //10.0 + 5.5
```

○ 정수 타입으로 계산을 원하는 경우 → 강제 형변환

```
int intValue = 10;
double doubleValue = 5.5;
int result = intValue + (int) doubleValue; //10 + 5
```

#### 연산식에서 실수 타입의 자동 변환

```
int x = 1;
int y = 2;
double result = x / y;
System.out.println(result); //0.5가 출력될까요?
```

→ 결과는 0.0

#### ○ 실수 연산을 원한다면 피연산자 중 하나를 double로 강제 형변환해야 함

방법 1	방법 2	방법 3
int x = 1;	int x = 1;	int x = 1;
int y = 2;	int y = 2;	int y = 2;
double result = (double)	<pre>double result = x / (double)</pre>	double result = (double)
x / y;	у;	x / (double) y;
<pre>System.out.println(result);</pre>	<pre>System.out.println(result);</pre>	<pre>System.out.println(result);</pre>

## ch02.sec09.OperationPromotionExample.java

```
public class OperationPromotionExample {
 public static void main(String[] args) {
   byte result1 = 10 + 20; //컴파일 단계에서 연산
   System.out.println("result1: " + result1);
   byte v1 = 10;
   byte v2 = 20;
   int result2 = v1 + v2; //int 타입으로 변환 후 연산
   System.out.println("result2: " + result2);
   byte v3 = 10;
   int v4 = 100;
   long v5 = 1000L;
   long result3 = v3 + v4 + v5; //long 타입으로 변환 후 연산
   System.out.println("result3: " + result3);
                                                          result1: 30
   char v6 = 'A';
                                                          result2: 30
   char v7 = 1:
                                                          result3: 1110
   int result4 = v6 + v7; //int 타입으로 변환 후 연산
                                                          result4: 66
   System.out.println("result4: " + result4);
                                                          result4: B
   System.out.println("result4: " + (char)result4);
```

## ch02.sec09.OperationPromotionExample.java

```
int v8 = 10;
int result5 = v8 / 4; //정수 연산의 결과는 정수
System.out.println("result5: " + result5);
int v9 = 10;
double result6 = v9 / 4.0; //double 타입으로 변환 후 연산
System.out.println("result6: " + result6);
int v10 = 1;
int v11 = 2;
double result7 = (double) v10 / v11; //double 타입으로 변환 후 연산
System.out.println("result7: " + result7);
                                                    result5: 2
                                                    result6: 2.5
                                                    result7: 0.5
```

### 🦁 String 타입 자동 변환

○ 정수 + 문자열 또는 문자열 + 정수 → 문자열

```
int value = 3 + 7; \rightarrow int value = 10;

String str = "3" + 7; \rightarrow String str = "3" + "7"; \rightarrow String str = "37";

String str = 3 + "7"; \rightarrow String str = "3" + "7"; \rightarrow String str = "37";
```

○ 주의사항: 정수 + 정수 + 문자열 → 정수 합계 계산 후 문자열로 결합

```
int value = 1 + 2 + 3; \rightarrow int value = 3 + 3; \rightarrow int value = 6;

String str = 1 + 2 + "3"; \rightarrow String str = 3 + "3"; \rightarrow String str = "33";

String str = 1 + "2" + 3; \rightarrow String str = "12" + 3; \rightarrow String str = "123";

String str = "1" + 2 + 3; \rightarrow String str = "12" + 3; \rightarrow String str = "123";
```

```
String str = "1" + (2 + 3); \rightarrow String str = "1" + 5; \rightarrow String str = "15";
```

# ch02.sec09.StringConcatExample.java

```
package ch02.sec09;
public class StringConcatExample {
 public static void main(String[] args) {
   //숫자 연산
   int result1 = 10 + 2 + 8;
   System.out.println("result1: " + result1);
   //결합 연산
   String result2 = 10 + 2 + "8";
   System.out.println("result2: " + result2);
   String result3 = 10 + "2" + 8;
   System.out.println("result3: " + result3);
   String result4 = "10" + 2 + 8;
   System.out.println("result4: " + result4);
   String result5 = "10" + (2 + 8);
   System.out.println("result5: " + result5);
```

```
result1: 20
result2: 128
result3: 1028
result4: 1028
result5: 1010
```

# 10 문자열을 기본 타입으로 변환

# String 타입 변환하기

변환 타입	사용 예
String → byte	<pre>String str = "10"; byte value = Byte.parseByte(str);</pre>
String → short	<pre>String str = "200"; short value = Short.parseShort(str);</pre>
String → int	<pre>String str = "300000"; int value = Integer.parseInt(str);</pre>
String → long	String str = "40000000000";  long value = Long.parseLong(str);
String → float	<pre>String str = "12.345"; float value = Float.parseFloat(str);</pre>
String → double	<pre>String str = "12.345"; double value = Double.parseDouble(str);</pre>
String → boolean	<pre>String str = "true"; boolean value = Boolean.parseBoolean(str);</pre>

# ch02.sec10.PrimitiveAndStringConversionExample.java

```
package ch02.sec10;
public class PrimitiveAndStringConversionExample {
 public static void main(String[] args) {
   int value1 = Integer.parseInt("10");
   double value2 = Double.parseDouble("3.14");
   boolean value3 = Boolean.parseBoolean("true");
   System.out.println("value1: " + value1);
   System.out.println("value2: " + value2);
   System.out.println("value3: " + value3);
   String str1 = String.valueOf(10);
   String str2 = String.value0f(3.14);
   String str3 = String.valueOf(true);
   System.out.println("str1: " + str1);
   System.out.println("str2: " + str2);
   System.out.println("str3: " + str3);
```

```
value1: 10
value2: 3.14
value3: true
str1: 10
str2: 3.14
str3: true
```

#### ♡ 변수 범위를 나타내는 중괄호 {} 블록

○ 조건문과 반복문의 중괄호 {} 블록 내에 선언된 변수는 해당 중괄호 {} 블록 내에서만 사용 가능

```
public static void main(String[] args) {
 if(...) | {
  int var2; ————
                     -----if 블록에서 선언
  //var1과 var2 사용 가능
                                         블록
                                                  메소드
 for(···) {
                                                  블록
                     ----- for 블록에서 선언
  int var3; ———
                                         for
  //var1과 var3 사용 가능
                                         블록
  //var2는 사용 못함
 //var1 사용 가능
 //var2와 var3는 사용 못함
```

## 11 <mark>변수 사용 범위</mark>

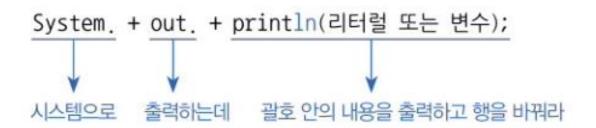
# ch02.sec11.VariableScopeExample.java

```
public class VariableScopeExample {
  public static void main(String[] args) {
    int v1 = 15;
    if(v1>10) {
      int v2 = v1 - 10;
    }
    int v3 = v1 + v2 + 5; //v2 변수를 사용할 수 없기 때문에 컴파일 에러 발생
  }
}
```

## 12 <mark>콘솔로 변수값 출력</mark>

## 🗸 println() 메소드로 변수값 출력하기

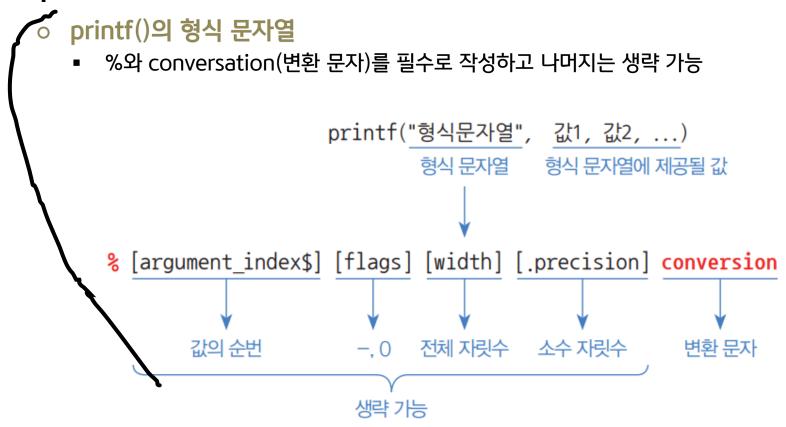
○ 모니터에 값을 출력하기 위해 System.out.println() 이용



○ 출력 방법에 따라 println() 이외에도 다음과 같이 print(), printf()를 사용할 수 있음

메소드	의미
println(내용);	괄호 안의 내용을 출력하고 행을 바꿔라.
print(내용);	괄호 안의 내용을 출력하고 행은 바꾸지 말이라.
printf("형식문자열", 값1, 값2, …);	형식 문자열에 맞추어 뒤의 값을 출력해라.

### 🗸 println() 메소드로 변수값 출력하기



## 12 콘솔로 변수값 출력

## 🗸 println() 메소드로 변수값 출력하기

o printf()의 형식 문자열



## oprintln() 메소드로 변수값 출력하기

# o printf()의 형식 문자열

형식화된 문지	열	설명	출력 형태
정수	%d	정수	123
	%6d	6자리 정수. 왼쪽 빈자리 공백	123
	%–6d	6자리 정수. 오른쪽 빈자리 공백	123
	%06d	6자리 정수. 왼쪽 빈자리 0 채움	000123
실수	%10.2f	정수 7자리+소수점+소수 2자리. 왼쪽 빈자리 공백	123.45
	%-10.2f	정수 7자리+소수점+소수 2자리. 오른쪽 빈자리 공백	123.45
	%010.2f	정수 7자리+소수점+소수 2자리. 왼쪽 빈자리 0 채움	0000123.45
문자열	%s	문자열	abc
	%6s	6자리 문자열. 왼쪽 빈자리 공백	abc
	%–6s	6자리 문자열. 오른쪽 빈자리 공백	abc
특수 문자	\t \n %%	탭(tab) 줄바꿈 %	%

## ch02.sec12.PrintfExample.java

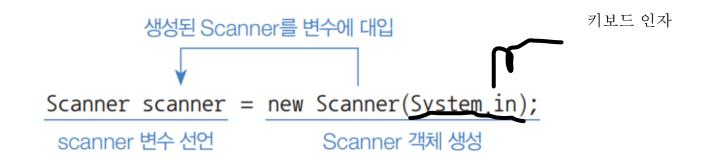
```
package ch02.sec12;
public class PrintfExample {
 public static void main(String[] args) {
   int value = 123;
   System.out.printf("상품의 가격:%d원\n", value);
   System.out.printf("상품의 가격:%6d원\n", value);
   System.out.printf("상품의 가격:%-6d원\n", value);
   System.out.printf("상품의 가격:%06d원\n", value);
   double area = 3.14159 * 10 * 10;
   System.out.printf("반지름이 %d인 원의 넓이:%10.2f\n", 10, area);
   String name = "홍길동";
   String job = "도적";
   System.out.printf("\%6d \mid \%-10s \mid \%10s \mid n", 1, name, job);
                                       상품의 가격:123원
                                       상품의 가격: 123원
                                       상품의 가격:123
                                       상품의 가격:000123원
                                       반지름이 10인 원의 넓이:
                                                              314.16
```

1 | 홍길동

도적

#### Scanner 타입 변수 활용하기

- o Scanner 타입 변수를 선언
- 대입 연산자 =를 사용해서 new 연산자로 생성한 Scanner 객체를 변수에 대입



o scanner.nextLine()을 실행하면 키보드로 입력된 내용을 문자열로 읽고 좌측 String 변수에 저장



## 13 키보드 입력 데이터를 변수에 저장

## ch02.sec13.ScannerExample.java

```
package ch02.sec13;
import java.util.Scanner;
public class ScannerExample {
 public static void main(String[] args) throws Exception {
   Scanner scanner = new Scanner(System.in);
   System.out.print("x 값 입력: ");
   String strX = scanner.nextLine();
   int x = Integer.parseInt(strX);
   System.out.print("y 값 입력: ");
   String strY = scanner.nextLine();
   int y = Integer.parseInt(strY);
                                                 x 값 입력: 3
   int result = x + y;
                                                  y 값 입력: 5
   System.out.println("x + y: " + result);
                                                  x + y: 8
   System.out.println();
```

# ch02.sec13.ScannerExample.java

같으면 true 다르면 false

```
while(true) {
     System.out.print("입력 문자열: ");
                                            자바는 문자열 내용 비교는 .eqauls 메소드 사용해라.
    String data = scanner.nextLine();
                                            ==이거는 참조 주솟값 비교
    if(data.equals("q")) {
      break;
    System.out.println("출력 문자열: " + data);
     System.out.println();
                                             입력 문자열: Hello
   System.out.println("종료");
                                             출력 문자열: Hello
                                             입력 문자열: 안녕하세요
                                             출력 문자열: 안녕하세요
boolean result = data . equals("문자열");
                                             입력 문자열: q
               변수의 문자열
                                             종료
                          비교
```