

## 2부 자바 기본 다루기

### - 3장 프로그래밍을 위한 자바의 자료형

최문환



# 3장 프로그래밍을 위한 자바의 자료형

1. 값이 변하지 않는 상수
2. 메모리 기억공간인 변수



# 1. 값이 변하지 않는 상수

## 1.1 소수점을 가지지 않는 정수형 상수 (수치형 상수)

| 상수의 종류 | 예                | 의미   |
|--------|------------------|--|
| 2진수    | 0b10             | 0은 영문자가 아니고 숫자.<br>0b는 2진수 상수로 인식                  |
| 10진수   | 4, 10, 80        |  |
| 8진수    | 04, 012, 0100    | 맨 앞에 숫자 '0'을 덧붙이면 8진수 상수로 인식                       |
| 16진수   | 0x4, 0xA, 0x6    | 맨 앞에 숫자 '0'과 영문자 'x', 즉 '0x'를 덧붙이면 16진수 상수로 인식합니다. |
| long형  | 10L, 034L, 0x2AL | 10진, 8진, 16진 상수 뒤에 L을 덧붙임                          |

# 1. 값이 변하지 않는 상수

## 1.2 소수점을 갖는 실수형 상수 (수치형 상수)

| 상수의종류  | 예                  | 의미                              |
|--------|--------------------|---------------------------------|
| 소수형    | 1234.5, 0.0000987  | 가장 일반적으로 사용하는 실수형 데이터           |
| 지수형    | 1.2345E3, 0.987E-5 | 영문자E를 기준으로 앞에는 가수부 뒤에는 지수부를 기술함 |
| float형 | 1234.5f, 0.00987f  | 실수형 상수 뒤에 f를 덧붙임                |

# 1. 값이 변하지 않는 상수

## 1.3 단일 따옴표로 표현하는 문자형 상수

▶ 예

'A', 'a', '2'



# 1. 값이 변하지 않는 상수

## ▶ 확장 특수 출력 문자(escape sequence)

| 종 류      | 의 미  |
|----------|--|
| \n<br>\t | 엔터 키의 기능을 갖는다. 줄을 바꾼다(new line).<br>수평 탭으로 일정한 간격을 띄운다(tab). |
|          |  |

# 1. 값이 변하지 않는 상수

## 1.4 참 혹은 거짓을 갖는 논리형 상수

- 참(true), 거짓(false) 둘 중의 하나의 값만을 저장할 수 있는 자료형
- 다른 자료형으로 변환하지 못한다.

# <예제> 자바에서 사용되는 상수종류

[파일 이름 : Data01.java]

```
01:public class Data01 {  
02:    public static void main(String[] args) {  
03:  
04:        //(1) 정수 : 소수점이 없는 수  
05:        System.out.println(1);  
06:        //(2) 실수 : 소수점이 있는 수  
07:        System.out.println(1.5);  
08:        //(3) 문자 : 단일 따옴표로 묶어줌  
09:        System.out.println('a');  
10:        //(4) 논리값 : true, false  
11:        System.out.println(true);  
12:    }  
13:}
```



# <예제>더 다양한 상수-[파일 이름 : Data02.java]

```
001:public class Data02 {  
002:    public static void main(String[] args)    {  
003:        //(1) long 형 상수 : 숫자 끝에 L 혹은 l을 붙임  
004:        System.out.println(1L);  
005:        //(2) float 형 상수 : 숫자 끝에 F 혹은 f를 붙임  
006:        System.out.println(1.5f);  
007:        //(3) 문자열 : 이중 따옴표로 묶어줌  
008:        System.out.println("abc");  
009:    }  
010:}
```

## 2. 메모리 기억공간인 변수 살피기

- 프로그램 실행 중에 변하는 값을 저장할 수 있는 메모리 기억공간
- 변수는 앞에 언급한 상수(값)를 저장하기 위한 공간을 의미
- 이러한 공간은 컴퓨터의 메모리(RAM)의 특정 위치(스택)에 만들어진다.

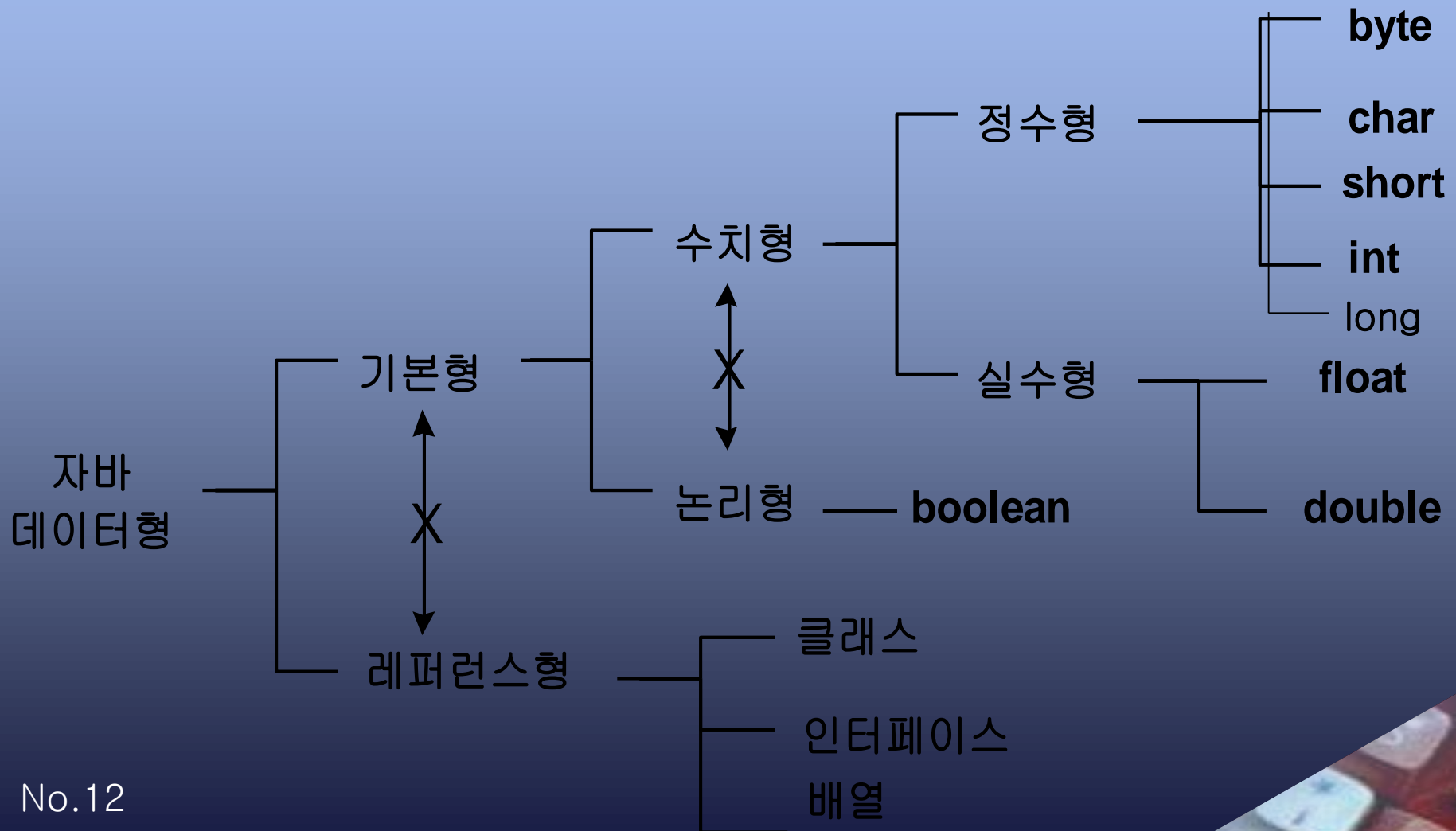


## 2.1 기억공간을 확보하는 변수 선언

- 변수는 상수처럼 그냥 사용하지 못하고 변수 선언 과정을 거친 후 사용할 수 있다.

자료형    변수\_이름;

## 2.2 기본 자료형



## 2.3 소수점이 없는 정수형

| 유 형   | 크 기            | 허 용 값                   |   |
|-------|----------------|-------------------------|---|
| byte  | 1 Byte( 8 bit) | $-2^7 \sim 2^7-1$       | $-128 \sim 127$                                 |
| short | 2 Byte(16 bit) | $-2^{15} \sim 2^{15}-1$ | $-32768 \sim 32767$                             |
| int   | 4 Byte(32 bit) | $-2^{31} \sim 2^{31}-1$ | $-2147483648 \sim 2147483647$                   |
| long  | 8 Byte(64 bit) | $-2^{63} \sim 2^{63}-1$ | $-9223372036854775808 \sim 9223372036854775807$ |

정수를 저장하기 위한 자료형을 4가지로 나눈 이유는 저장할 데이터(상수, 값)에 따라 기억공간의 크기를 다르게 할당하기 위한 것

# 변수의 이름을 명명하기 위한 규칙

- ① 영문자(A~Z, a~z)와 숫자(0~9)와 밑줄문자(\_)의 조합으로 만들어집니다.
- ② 첫 글자는 반드시 영문자나 '\_'로 시작하여야 합니다. 숫자로 시작해서는 안 됩니다.
- ③ 식별자는 철자(스펠링)가 같다고 해도 대소문자를 구분하기 때문에 조심해야 합니다.
- ④ 자바에서 사용되는 예약어는 식별자로 사용할 수 없습니다.
- ⑤ 식별자는 가급적이면 자기 역할에 맞는 이름을 부여합니다.

# <잠깐만>- 예약어와 식별자

## 예약어란?

자바에서 미리 정의하고 의미를 부여한 단어. 이미 정해진 자신의 역할이 있기 때문에 프로그래머는 예약어를 다른 용도로 사용할 수 없다.

int, char, class, if, else, switch, for, while, break

## 식별자란?

프로그래머가 특별히 의미를 부여하는 단어  
예를 들어 변수의 이름이나 메소드의 이름이나 클래스의 이름 등을 식별자라고 한다.

# <예제>정수 상수를 저장하는 정수형 변수

```
01:public class Data03{
02: public static void main(String[] args) {
03:     int a;           //변수 선언하고
04:     a=1;             //변수에 값을 저장
05:     System.out.println(a);
06:
07:     a=2;             //변수는 값을 변경할 수 있다.
08:     System.out.println(a); //마지막에 대입한 값만 유지
09: }
10:}
```



## 2.4 자료형의 역할

### 1. 저장되는 값의 형태를 결정

1. 소수점이 없는 값을 정수형 상수를 저장하기 위해서는 int형으로 변수 선언

```
int a=1;
```

2. 소수점이 있는 값을 실수형 상수를 저장하기 위해서는 double형으로 변수 선언

```
double m=2.4;
```

3. 참, 거짓 논리값을 저장하기 위해서는 boolean 형으로 변수 선언

```
boolean k=true;
```

## 2.4 자료형의 역할

### 2. 메모리의 사이즈를 결정

메모리 할당되는 사이즈에 따라 변수에 저장할 수 있는 값의 허용 범위가 달라진다.

```
byte a=1;  
a=128;           //컴파일 에러
```

```
short b=128;  
b=32768;         //컴파일 에러
```

```
int c=32768;  
int c=123456L;   //컴파일 에러
```

```
long d=123456L;
```

## 2.5 자료형 변환

```
long d=123456;    //암시적인 형 변환
```

```
int c=123456L;    //컴파일 에러
```

## 2.5.1 암시적인 형 변환

```
short t b=128;
```

```
int c=32768;
```

```
c=b;           // 암시적인 형 변환
```

## 2.5.2 명시적인 형 변환

```
short b=128;  
int c=32768;
```

```
b=c;           //컴파일 에러
```

```
b=(short)c;    //명시적인 형 변환
```

# <예제> 암시적인 형 변환과 명시적인 형 변환

```
01: public class Data05 {  
02:     public static void main(String[] args) {  
03:         byte a=1;  
04:         short b=128;  
05:         int c=32768;  
06:  
07:         b=a; //암시적인 형 변환  
08:         System.out.println(b);  
09:  
10:         b=(short)c; // 명시적인 형변환  
11:         //오버플로우가 발생되어 엉뚱한 값 출력  
12:         System.out.println(b);  
13:     }  
14: }
```

## 2.6 소수점이 있는 실수형

```
int a=23.7;           //컴파일 에러
```

| 종류  | 유형     | 크기     |
|-----|--------|--------|
| 실수형 | float  | 4 Byte |
|     | double | 8 Byte |

## 2.6 소수점이 있는 실수형

- `float b=23.7;`      `//컴파일 에러`
- `double c=23.7;`      `//올바른 표현`
- `float b=23.7f;`      `//올바른 표현`



# <예제> 실수 자료형 저장하기

```
01: public class Data06 {  
02:     public static void main(String[] args){  
03:         double    a=23.7;  
04:         float     b=23.7f;  
05:         System.out.println(a);  
06:         System.out.println(b);  
07:     }  
08: }
```

## 2.7 문자 한 개를 저장하는 문자형

| 종 류 | 유 형  | 크 기                           | 허 용 값     |
|-----|------|-------------------------------|-----------|
| 문자형 | char | 2 Byte (16 bit)<br>16비트 유니 코드 | 0 ~ 65535 |

- 대문자 'A' (65)
- 소문자 'a' (97)
- 정수형태의 문자 '0' (48)

# <예제> 문자 자료형 저장하기

```
01: public class Data07 {  
02:     public static void main(String[] args){  
03:         char x;  
04:         x='A';  
05:         System.out.printf("%c->%d\n",x,(int)x);  
06:         x='0';  
07:         System.out.printf("%c->%d\n",x,(int)x);  
08:         x=0;  
09:         System.out.printf("%c->%d\n",x,(int)x);  
10:         x='a';  
11:         System.out.printf("%c->%d\n",x,(int)x);  
12:     }  
13: }
```

20.27

# <잠깐만>-println, print, printf

## ① println

- ln은 라인의 약어로서 메소드 내에 기술한 내용을 출력한 후 자동으로 개행(줄을 바꿈)한다.

## ② print 메소드

- 메소드내에 기술한 내용만을 출력할뿐 줄 바꿈을 하지 않는다.

## ③ printf 메소드

- printf 지시자(형식지정자)를 기술하여 원하는 자료 형태로 출력할 수 있는 메소드.
- 문자 데이터를 문자 형태로 출력하기 위해서는 %c라는 형식 지정자를 사용한다.
- 형식 지정자는 %기호 다음에 영문자를 기술하는데 형식 지정자 %d는 정수형 10진수 형태로 출력하게 된다.

# 문자여러개를 집합으로 관리하는 문자열형 (String)

'AB' //잘못된 표현

"AB"

char x="AB"; //잘못된 표현

String y="AB";

String y= 'A'; //잘못된 표현

# <예제> 문자열 저장하기

```
01: public class Data08 {  
02:     public static void main(String[] args){  
03:         String y;  
04:         y="AB";  
05:         System.out.println(y);  
06:         y="A";  
07:         System.out.println(y);  
08:     }  
09: }
```

# <문제>

1. 잘못된 문장을 골라내고 그 이유를 설명하시오.

```
public class DataEx01 {  
    public static void main(String[] args) {  
        char        a='a';           //A.  
        char        b="a";           //B.  
        String      c="a";           //C.  
        String      d='a';           //D.  
        char        e= "ab" ;        //E.  
        String      f="ab";          //F.  
    }  
}
```

No.31

# <문제>

2. 잘못된 문장을 골라내고 그 이유를 설명하시오.

```
public class DataEx02{  
    public static void main(String[] args) {  
        byte var1=128; //A.  
        short var2=128; //B.  
        int var3=28L; //C.  
        long var4=128L; //D.  
        float var5=123456.789123; //E.  
        double var6=123456.789123; //F.  
    }  
}
```



# <문제>

3. 잘못된 문장을 고쳐서 에러가 발생하지 않도록 수정하시오.

```
001:public class DataEx03{
002:    public static void main(String[] args) {
003:        byte var1=127;
004:        short var2=128;
005:        int var3=128;
006:        long var4=128L;
007:        var4 = var1;
008:        System.out.println(var1+ " , " + var2);
009:        var1 = var3;
010:        System.out.println(var1+ " , " + var3);
011:
012:        float var5=123456.789123;
013:        double var6=123456.789123;
014:        var5 =var6;
015:        System.out.println(var5+ " , " + var6);
016:        var6 = var5;
017:        System.out.println(var5+ " , " + var6);
018:    }
019:}
```

# <문제>

4. char 형의 저장할 수 있는 데이터 값의 허용 범위는 얼마입니까?
5. 변수를 선언하기 위해서는 자료형과 변수의 이름을 결정해야 합니다. 변수 이름으로 사용할 수 있는 것을 고르시오.
- A. false
  - B. default
  - C. \_object
  - D. a-class

# <문제>

6. 다음 프로그램을 컴파일 했을 때 결과를 추측해보시오.

```
public class DataEx06 {  
    public static void main (String[] args) {  
        byte b = 127;  
        byte c = 126;  
        byte d = b + c;  
    }  
}
```

# <문제>

7. byte 형을 저장할 수 있는 데이터 값의 허용 범위는 얼마입니까?

8. float 형 변수 선언이 올바르게 된 것 두개만 고르시오.

```
public class DataEx09 {  
    public static void main (String[] args) {  
        float f1 = 1F;    //A.  
        float f2 = 1.0;    //B.  
        float f3 = '1';    //C.  
        float f4 = "1";    //D.  
        float f5 = 1.0d;    //E.  
    }  
}
```