

● 변수: 단 하나의 값을 저장할 수 있는 메모리 공간.

● 변수 이름: 메모리 공간에 이름을 붙여주는 것. 그 이름을 이용해서 저장공간(변수)에 값을 저장하고, 저장된 값을 읽어오기도 한다.

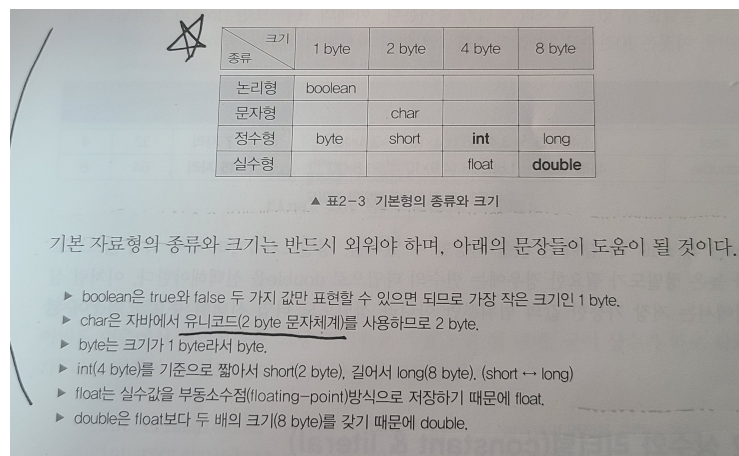
● 변수의 타입: 값(data)의 종류(type)에 따라 값이 저장될 공간의 크기와 저장형식을 정의한 것이 자료형(data type)이다.

↳ 기본형 변수(primitive type·원시타입 변수): 실제 값(data)을 저장한다. 논리형(boolean), 문자형(char-내부적으로 문자를 정수(유니코드)로 저장), 정수형(byte, short, int, long), 실수형(float, double) 계산을 위한 실제 값을 저장한다.

↳ 참조형(reference type): 객체의 주소를 저장한다. 8개의 기본형을 제외한 나머지 타입. 선언 시 변수의 타입으로 클래스의 이름을 사용하므로 클래스 이름이 참조변수의 타입이 된다. 그래서 새로운 클래스를 작성한다는 것은 새로운 참조형을 추가하는 셈이다. 참조형 변수는 null 또는 객체의 주소(JVM32bit - 4byte 정수)를 값으로 갖는다. null은 어떤 객체의 주소도 저장되어 있지 않음을 뜻한다.

- 참조형 변수 간의 연산을 할 수 없다. 실제 연산/변환에 사용되는 것은 모두 기본형이다(boolean 제외).

- 기본형은 값(data)을 저장하므로 값의 종류를 이야기할 때 자료형(data type), 참조형은 객체의 주소(4 byte 정수)를 저장하므로 객체의 종류를 이야기할 때 타입(type). 타입이 자료형을 포함하는 보다 넓은 의미이기도 하다.



종류 \ 크기	1 byte	2 byte	4 byte	8 byte
논리형	boolean			
문자형		char		
정수형	byte	short	int	long
실수형			float	double

▲ 표2-3 기본형의 종류와 크기

기본 자료형의 종류와 크기는 반드시 외워야 하며, 아래의 문장들이 도움이 될 것이다.

- ▶ boolean은 true와 false 두 가지 값만 표현할 수 있으면 되므로 가장 작은 크기인 1 byte.
- ▶ char은 자바에서 유니코드(2 byte 문자체계)를 사용하므로 2 byte.
- ▶ byte는 크기가 1 byte라서 byte.
- ▶ int(4 byte)를 기준으로 짧아서 short(2 byte), 길어서 long(8 byte). (short ↔ long)
- ▶ float는 실수값을 부동소수점(floating-point) 방식으로 저장하기 때문에 float.
- ▶ double은 float보다 두 배의 크기(8 byte)를 갖기 때문에 double.

- int는  $2 \times 10^9$  만큼 저장가능하다 (대략 10자리 수)

- 실수형은 정수형에 비해 훨씬 큰 값을 저장할 수 있지만 오차가 발생할 수 있는 단점이 있어 정밀도(precision - 높으면 발생할 수 있는 오차의 범위가 줄어듦)도 중요하다. float의 정밀도는 7자리인 반면 double은 15자리다.

● 변수의 초기화(initialization): 변수의 초기화란, 변수를 사용하기 전에 처음으로 값을 저장하는 것. 메모리는 여러 프로그램이 공유하는 자원이므로 전에 다른 프로그램에 의해 저장된 알 수 없는 값(쓰레기값, garbage value)이 남아있을 수 있기 때문이다.

- 지역변수는 사용되기 전에 초기화를 반드시 해야 하지만, 클래스 변수와 인스턴스 변수는 초기화를 생략 가능하다.

● 변수의 명명규칙

↳ 식별자(identifier): 프로그래밍에서 사용하는 모든 이름 ('변수의 이름'); 식별자는 같은 영역 내에서 서로 구분(식별)될 수 있어야 한다.

1) 대소문자가 구분되며 길이에 제한이 없다.

2) 예약어(key word, reserved word)를 사용해서는 안 된다. ex) break, char, class, do, double, final, public, true

3) 숫자로 시작해서는 안 된다.

4) 특수문자는 '\_'와 '\$'만 허용한다.

규칙(coding convention)

- 클래스 이름의 첫 글자는 대문자.

- 여러 단어로 이루어진 이름은 단어의 첫 글자를 대문자로.

- 상수의 이름은 모두 대문자로. 여러 단어로 이어진 경우 '\_'로 구분.

● 상수(constant): 값을 저장할 수 있는 공간으로 한 번 값을 저장하면 다른 값으로 변경할 수 없다. 선언하는 방법은 앞에 final을 붙여주면 되며, 반드시 사용하기 전에 초기화해야 한다(선언과 동시에 초기화 하는 것이 좋다). 이름은 모두 대문자로 하는 것이 관례이며, 여러 단어로 이어져있는 경우 '\_'로 구분한다. 리터럴에 의미 있는 이름을 붙여서 코드의 이해와 수정을 쉽게 만든다.

● 리터럴(literal): 변수/상수의 값. 변수/상수는 '값을 저장할 수 있는 메모리 공간'이고, 그 공간(상수/변수)의 값이 리터럴이다.

```
int year = 2022;
final int MAX_VALUE = 100;
// year은 변수, MAX_VALUE는 상수. 2022, 100은 리터럴.
```

└ 리터럴의 종류: long타입 (L/l), 실수형 float(f/F)

└ 정수형 리터럴의 중간에 구분자 '\_'를 넣을 수 있게 되어서 큰 숫자를 편하게 읽을 수 있다.

```
long big = 100_1000_100_000L;
```

└ 리터럴에 소수점이나 10의 제곱을 나타내는 기호 E또는 e(ex. 1e1, 1e-3), 그리고 접미사 f, F, d, D를 포함하고 있으면 실수형 리터럴로 간주한다.

└ 저장될 변수의 타입과 일치하는 것이 보통이지만, 타입이 달라도 저장범위가 넓은 타입에 좁은 타입의 값을 저장하는 것은 허용된다.

```
int i = 'A';
long l = 123;
double d = 3.14f;
```

└ 문자열 리터럴은 ""안에 아무것도 넣지 않은 것을 빈 문자열(empty string)이라고 한다. 하지만 문자 리터럴은 반드시 ""안에 하나의 문자가 있어야 한다.

└ 덧셈 연산자(+)는 피연산자가 모두 숫자일 때는 두 수를 더하지만, 피연산자 중 어느 한 쪽이 String이면 나머지 한 쪽을 먼저 String으로 변환한 다음 두 String을 결합한다.

● 형식화된 출력 printf(): 지시자(specifier)을 통해 변수의 값을 여러 가지 형식으로 변환하여 출력한다. 출력후 줄바꿈을 하지 않아 지시자 %n을 따로 넣어주어야 한다.

└ %b: boolean

└ %d: (decimal) 정수

- ↳ **%f: (floating point) 소수:** 기본적으로 소수점 아래 7자리에서 반올림하여 6자리까지만 출력. % 전체자리(소수 점포함, 빈자리는 공백으로, 0으로 채움).소수점아래자리f로 지정할 수도 있다.
  - ↳ **%e/E:** 지수(exponent) 표현식의 형식으로 출력
  - ↳ **%c: (charater) 문자**
  - ↳ **%s: (string) 문자열:** 원하는 만큼의 출력공간을 확보하거나 문자열 일부만 출력가능하다([%s] 문자의 길이만큼 출력공간 확보, [%xs] 최소 x글자 출력 공간 확보(우측정렬), [%-xs] 최소 x글자 출력 공간 확보(좌측정렬), [% .xs] 왼쪽에서 x글자만 출력). 지정된 숫자보다 문자열의 길이가 작으면 빈자리는 공백으로 출력된다(기본적으로 우측에, -을 붙이면 좌측에).
-