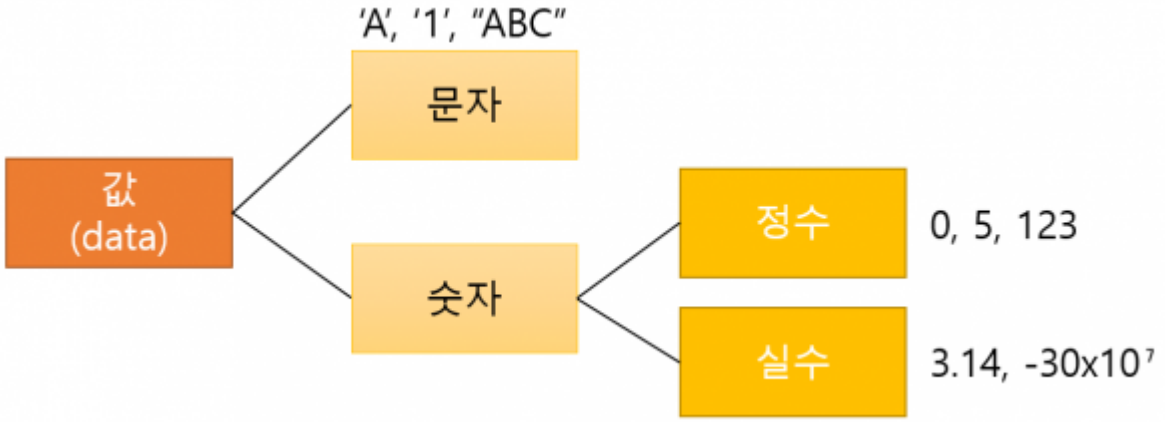
자바 데이터 타입, 변수 그리고 배열

자바의 프리미티브 타입, 변수 그리고 배열을 사용하는 방법을 익힙니다.

프리미티브 타입 종류와 값의 범위 그리고 기본 값



- 우리가 주로 사용하는 값의 종류는 크게 문자와 숫자로 나눌 수 있으며 여기서 숫자는 다시 정수와 실수로 나뉜다.
- 기본형은 모두 8가지의 타입(자료형)이 있으며, 크게 논리형, 문자형, 정수형, 실수형으로 구분된다.

타입 종류

분류	타입
논리형	Boolean
	true와 false 중 하나를 값으로 갖으며 조건식과 논리적 계산에 사용된다.
문자형	char
	문자를 저장하는데 사용되며, 변수에 하나의 문자만 저장이 가능하다.
정수형	short, byte, int, long
	정수를 저장하는데 사용되며, 주로 int가 사용된다. byte는 이진 데이터를 다룰 때 사용되며 short는 C언어와의 호환을 위해서 추가되었다.
실수형	float, double
	실수를 저장하는데 사용되며, 주로 double이 사용된다.

- 정수형은 가장 많이 사용되기에 타입이 4가지나 제공된다.
- 각 타입별로 범위가 다르기에 범위에 맞는 값을 사용하면 된다.

타입 범위

자료형	저장 가능한 값의 범위	크기	
		bit	byte
Boolean	false, true	8	1
char	'₩u0000' ~ '₩uffff'(0~2^{16}-1, 0~65535)	16	2
byte	-128 ~ 127(-2^7~2^7 -1)	8	1
short	-32,768 ~ 32,767(-2^{15}~2^{15}-1)	16	2
int	-2,147,483,648 ~ 2,147,483,647(-2^{31}~2^{31}-1, 약 ±20억)	32	4
long	-9,223,372,036,854,775,808 ~ 9,223,372,036,854,775,808(-2^{61}~2^{61}-1)	64	8
float	1.4E-45 ~ 3.4E38(1.4*10E-45f ~ 3.4*10E+38f)	32	4
double	4.94065645841246544E-324 ~ 1.79769313486231570E+308	64	8

- boolean은 true와 false 두 값만 표현하면 되기에 1바이트면 충분하다.
  - 기본 값 : `false`
- char는 자바에서 유니코드(2 byte문자 체계)를 사용하기에 2byte
  - 기본 값 : `\u0000`
- byte는 크기가 1byte이므로 byte.
  - 기본 값 : `0`
- int(4 byte)를 기준으로 짧게는 (2 byte) 길게는 (8 byte)를 취사선택한다.
  - 기본 값 : `0`
- float은 실수값을 부동소수점(floating-point)방식으로 저장하기 때문에 float
  - 기본 값 : `0.0F`
- double은 float보다 두 배의 크기(8byte)를 갖기 때문에 double
  - 기본 값 : `0.0`

실수형의 정밀도

자료형	저장 가능한 값의 범위	정밀도	크기	
			bit	byte
float	1.4E-45 ~ 3.4E38	7자리	32	4
double	4.9E-324 ~ 1.8E308	15자리	64	8

- 실수형은 정수형과 저장 방식이 다르기에 같은 크기라도 훨씬 큰 값을 표현할 수는 있지만, 오차가 발생할 수 있다. 그래서 정밀도(precision)가 중요한데, 정밀도가 높을수록 오차의 범위가 줄어든다.
- 위 표를 보면 `float`의 정밀도는 7자리로 10진수로 7자리의 수를 오차없이 저장할 수 있다는 의미다. 그렇기에 사용할 변수의 값의 범위가 7자리를 넘는다면 정밀도를 고려해 `double` 타입을 사용해야 한다.

프리미티브 타입과 레퍼런스 타입

자료형은 크게 '기본형(Primitive Type)' 과 참조형(Reference Type)으로 나눌 수 있다.

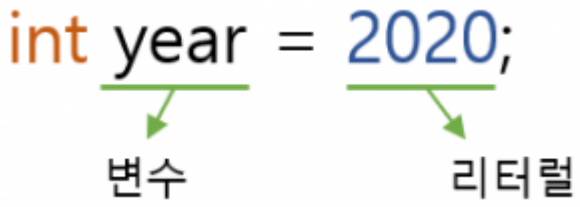
- 기본형(Primitive Type) : 논리형(boolean), 문자형(char), 정수형(byte, short, int, long), 실수형(float, double) 계산을 위한 실제 값을 저장한다.
- 참조형(reference type) : 객체의 주소를 저장한다. 기본적으로 `Java.lang.Object`를 상속받을 경우 참조형이 된다. 즉, 기본형을 제외하고는 참조형이라 생각해도 된다.

좀 더 얘기하자면 기본형은 메모리영역의 스택영역에 실제 값들이 저장된다면, 참조형은 실제 인스턴스는 힙영역에 생성되있고, 그 영역의 주소를 스택영역에서 저장하고 있다고 보면 된다.

리터럴

> 그 자체로 값을 의미하는 것

- 리터럴은 데이터 그 자체를 의미한다.
- 아래 그림에서 `2020`이 리터럴이다.



- 즉, 2020, 123, 3.14, “ABC” 와 같은 값들을 리터럴이라고 하는데 본래 이러한 값들은 상수라 불러야 하지만 프로그래밍에서는 상수를 **'값을 한 번 저장하면 변경할 수 없는 저장공간'**으로 정의했기 때문에 이와 구분하기 위해서 리터럴이라는 용어를 사용한다.
- 그러니 리터럴은 기존에 알고있던 상수의 다른 이름이라고 볼 수 있다.

> 인스턴스는 리터럴이 될 수 있을까?

- 인스턴스안의 값의 불변성(`Imutable`) 이 보장된다면 객체 리터럴이 될 수 있다.(불변 클래스(`imutable class`))
- 하지만 이렇게 불변성을 보장하도록 설계된 클래스를 제외하고 보통의 인스턴스는 동적으로 사용되고 내용이 변할 수 있기 때문에 객체 리터럴이 될 수 없다.
- Ex: `Java.lang.String` 이나 `java.awt.Color` 같은 클래스는 내용이 변해야 하는 상황이면 새로운 객체를 만들어 내용의 불변성이 보장되기에 객체 리터럴이라 부른다.

변수 선언 및 초기화하는 방법

변수의 스코프와 라이프타임

타입 변환, 캐스팅 그리고 타입 프로모션

1차 및 2차 배열 선언하기

타입 추론, var

Ref

https://catsbi.oopy.io/6541026f-1e19-4117-8fef-aea145e4fc1b
https://github.com/whiteship/live-study/issues/2

와프