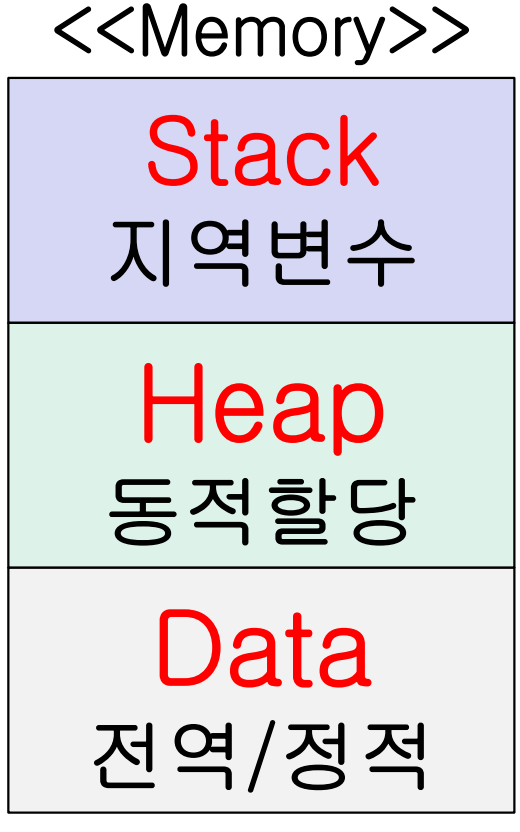
2강

JAVA_PROGRAMMING

메모리 구조

- ❖ 스택 영역(Runtime Stack)
  - ▪ 로컬변수를 할당, 선언된 블럭을 벗어나는 소멸됨.
  - ▪ 자동으로 초기화 되지 않으므로 초기화 해주어야 한다.
- ❖ 동적 할당 메모리영역
  - ▪ 배열로 만들어진 연속된 공간들이나 클래스에 포함된 멤버를 할당하는 영역.
  - ▪ Garbage Collector 에 의해 관리.(디폴트 초기화)
- ❖ 상수와 Method 영역
  - ▪ 프로그램의 시작 시 생성, 종료 시 소멸.
  - ▪ 고정데이터를 가진 상수나 추상화 형식을 지닌 Method의 경우 해당 내용을 기억해 두어야 할 필요성 때문에 만들어진 영역.(실행 시 위치 값을 가지게 된다.)
- ❖ Register 영역
  - ▪ JVM을 실행하면서 필요한 부분의 주소를 기억해 두는 곳.
  - ▪ 각 영역에 연결되어 있는 프로세서가 작동할 때 여기에 등록된 내용을 기초로 실행



상수(Constant)와 변수(Variable) 및 데이터 타입(Data Type)

- ❖ 상수 : 항상 일정한 값을 유지하는 데이터(리터럴 상수)
- ❖ 변수 : 특정 상황에 따라 변화하는 데이터
  - ✓ 상수와 변수 모두 메모리의 공간

- ❖ 자료형(Data type) : 데이터의 종류를 말한다.
  - ▪ 기본 자료형
    - ✓ 논리형(boolean), 정수형(byte, short, int, long), 실수형(float, double)
  - ▪ 클래스형(객체를 생성하기 위한 자료형)
    - ✓ 기본 자료형을 제외한 모든 Class(String, Random 등)

자바 사용자 정의 명칭

- ❖ 사용자가 정의한 이름(식별자)
- ❖ 폴더, 클래스, 메서드, 그리고 필드들의 이름 등을 정의할 때 사용

➢ 규칙

- ✓ 첫 글자는 $, _,영문 대소문자여야 한다.(한글도 가능하지만 권장하지는 않는다)
- ✓ 글자수에는 제한이 없다.
- ✓ 공백 문자는 포함할 수 없다.
- ✓ 특수문자(@, #, %, ^, &, !, ?)는 사용할 수 없다.
- ✓ 숫자는 첫 글자가 아닐 경우 사용이 가능하다
- ✓ 예약어는 사용할 수 없다.

➢ 권장사항

- ✓ Class : 첫 글자는 $, _,대문자를 사용하자
- ✓ Method(함수) : 첫 글자는 $, _, 소문자를 사용하자
- ✓ Field(변수) : 일반적으로 전부 대문자인 경우는 static final 필드이고
- ✓ 그 외는 첫 글자는 대문자, 나머지는 소문자이다
- ✓ 합성어의 첫 글자는 대문자로 한다(HelloJava)

논리 자료형(boolean)

❖ boolean

✓ 논리 자료형은 참/거짓의 값만 저장할 수 있다.

- 크기 = 1byte
- 범위 = true / false

ex)

- boolean bool = false;
- boolean형 변수 bool에는 false(거짓)이라는 값이 들어있다.

정수 자료형(byte)

❖ byte

- ✓ byte형은 정수형 자료형 중에 가장 작은 값을 저장할 수 있다.
- ✓ 배열이나 데이터 전송의 기본이 되는 자료형으로 많이 사용된다.
- ✓ 정수형 자료형은 모두 허용 범위 이상의 값으로 대입 할 수 없다.

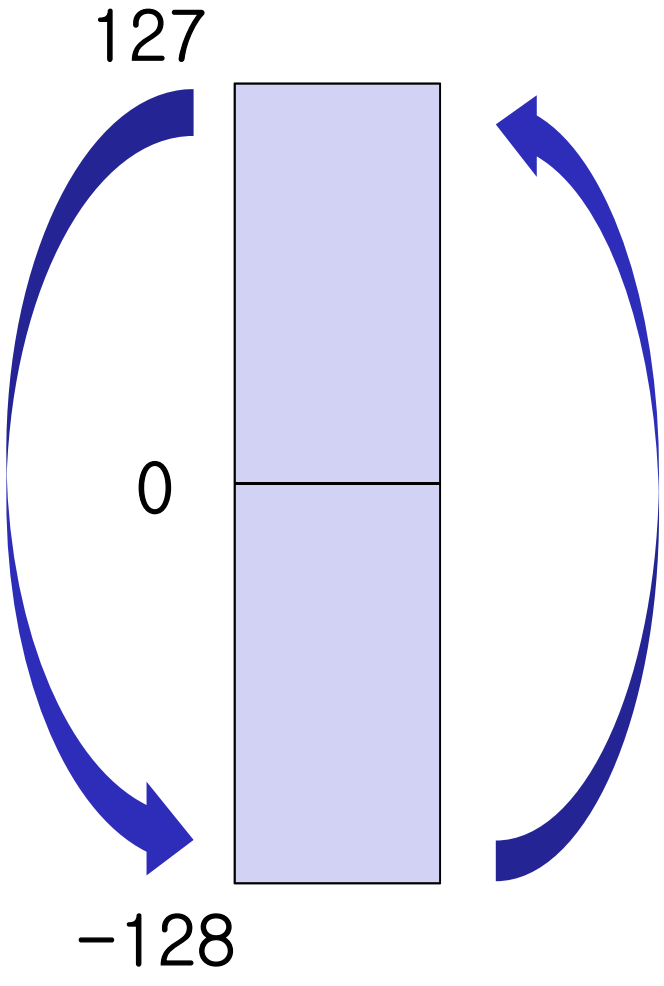
- 크기 = 1byte
- 범위 = -128~127

ex)

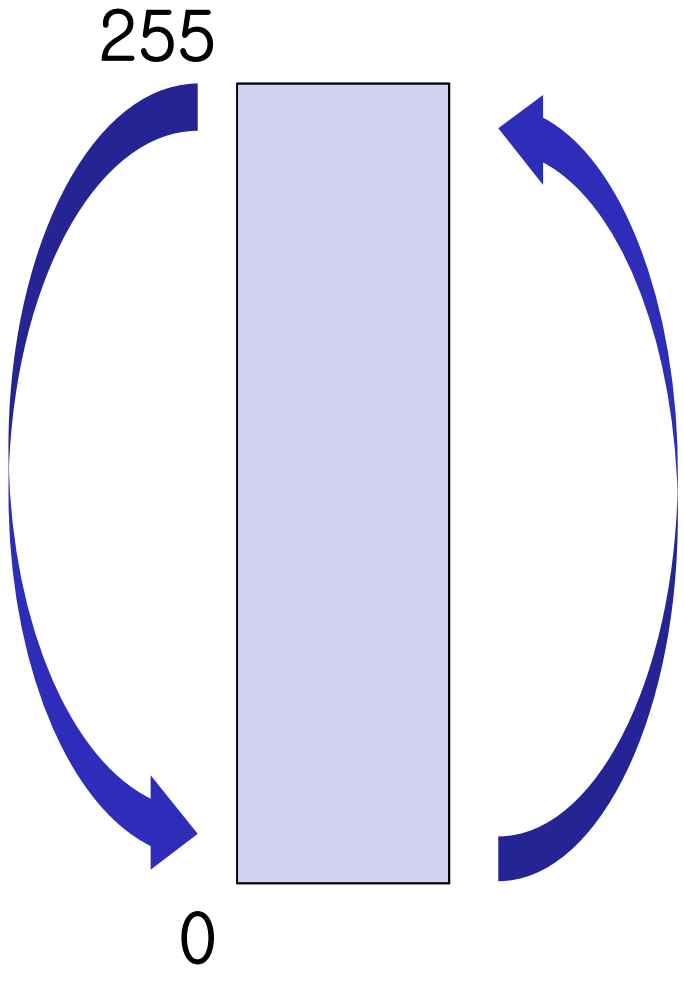
- byte b = 127;
- byte b = -128;
- byte b = 128; //에러!!!

데이터 범위 이해

✓ byte형 범위표현



< signed >



< unsigned >

정수 자료형(char)

❖ char

- ✓ 자바의 자료형 중 유일한 unsigned(부호없는)정수형이다.
- ✓ 아스키코드 + 유니코드를 포함하여 2byte의 범위를 가진다.
- ✓ 숫자 값이나 아스키코드,유니코드 등의 표현을 담을 수 있다.
- ✓ 아스키 코드 표 검색해 보기
  - 크기 = 2byte
  - 범위 = 0 ~ 65535

ex)

- char ch = 65; (‘A’의 아스키 코드 값)
- char ch = ‘A’;(‘ ’ 문자 하나를 묶어서 표현)
- char ch = ‘₩u0041’; (‘₩u’는 유니코드 명시방법, 41은 ‘A’의 16진수 값)

정수 자료형(short, int, long)

❖ short

✓ 정수 자료형으로 잘 사용되지는 않는 자료형이다

- 크기 = 2byte
- 범위 = -32768 ~ 32767

❖ int

✓ 모든 정수의 기본 자료구조가 int형이다.

✓ 정수 연산시에 발생하는 결과 값은 모두 int형으로 표현

- 크기 = 4byte
- 범위 = -2,147,483,648~2,147,483,647

❖ long

✓ int형의 두 배를 저장하는 정수 자료형

✓ 초기화 할때 l이나 L을 붙혀야 한다.

✓ 예) long lo = 123356L;

- 크기 = 8byte
- 범위 = -9,223,372,036,854,775,808 ~ 9,223,372,036,854,775,807

실수 자료형(float, double)

❖ float(단정도 실수형)

- 크기 = 4byte, 범위 = 1.4E-45 ~ 3.4028235E38;
- 실수를 표현하는 자료형이다.
- int와 같은 크기의 공간을 사용하지만 int보다 큰 값을 표현할 수 있다.
- 저장할 때 값 뒤에 f or F를 붙여야 된다.
- float f = 3.14F;

❖ double(배정도 실수형)

- 크기 = 8byte, 범위 = 4.9E-324 ~ 1.79769313486231570E308
- java에서 실수의 기본 자료형이다.
- long보다 큰 데이터를 저장할 수 있다.
- double d = 3.14;

❖ 실수형 사용 시 예약어

- 실수/0 연산 시 Infinity 결과 값(무한대)
- 실수(0)/0 연산 시 NaN 결과 값(수가 아닌 값)

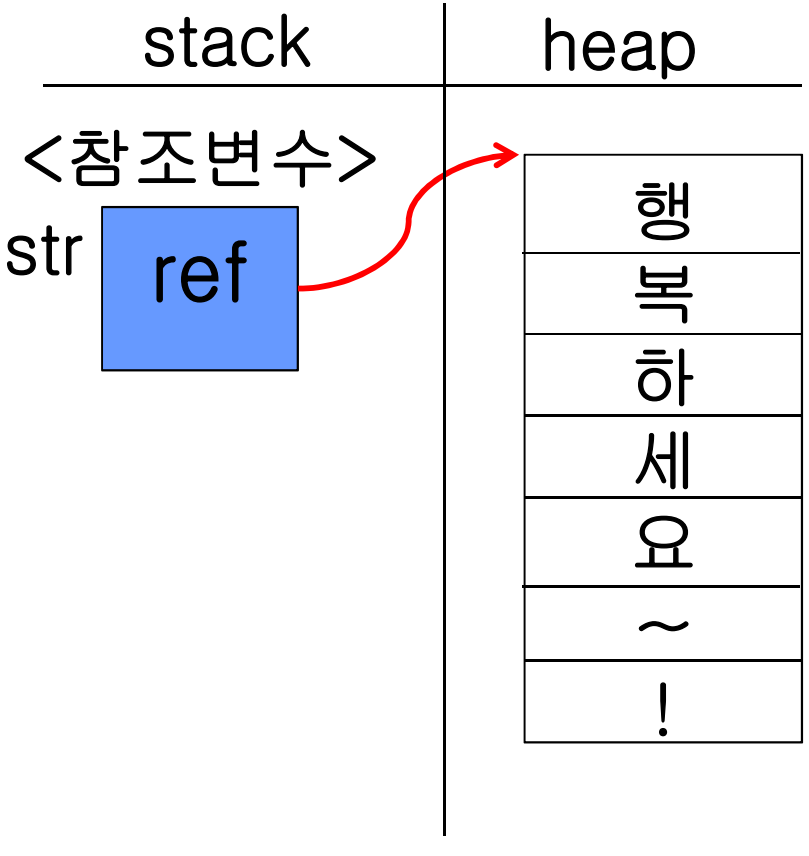
클래스형 자료형

❖ String

- ✓ String은 클래스
- ✓ 클래스는 자료형처럼 사용할 수 있다.
- ✓ String은 기본 자료형과 같은 개념으로 문자열을 처리하기 위한 클래스
  - 크기 = 4byte (클래스 참조 변수)
  - 범위 = 무한대
  - String str = “행복하세요~!”;

❖ 문자열 크기가 4byte인 이유?

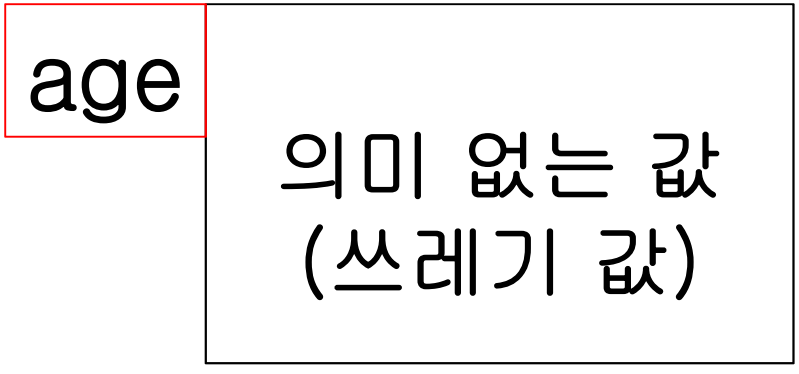
❖ 범위가 무한대인 이유?



변수 선언

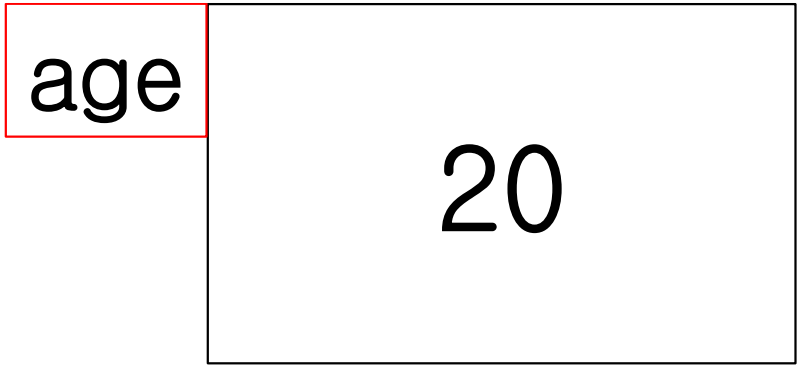
❖ 자료형 변수명; (선언)

```
int age;
```



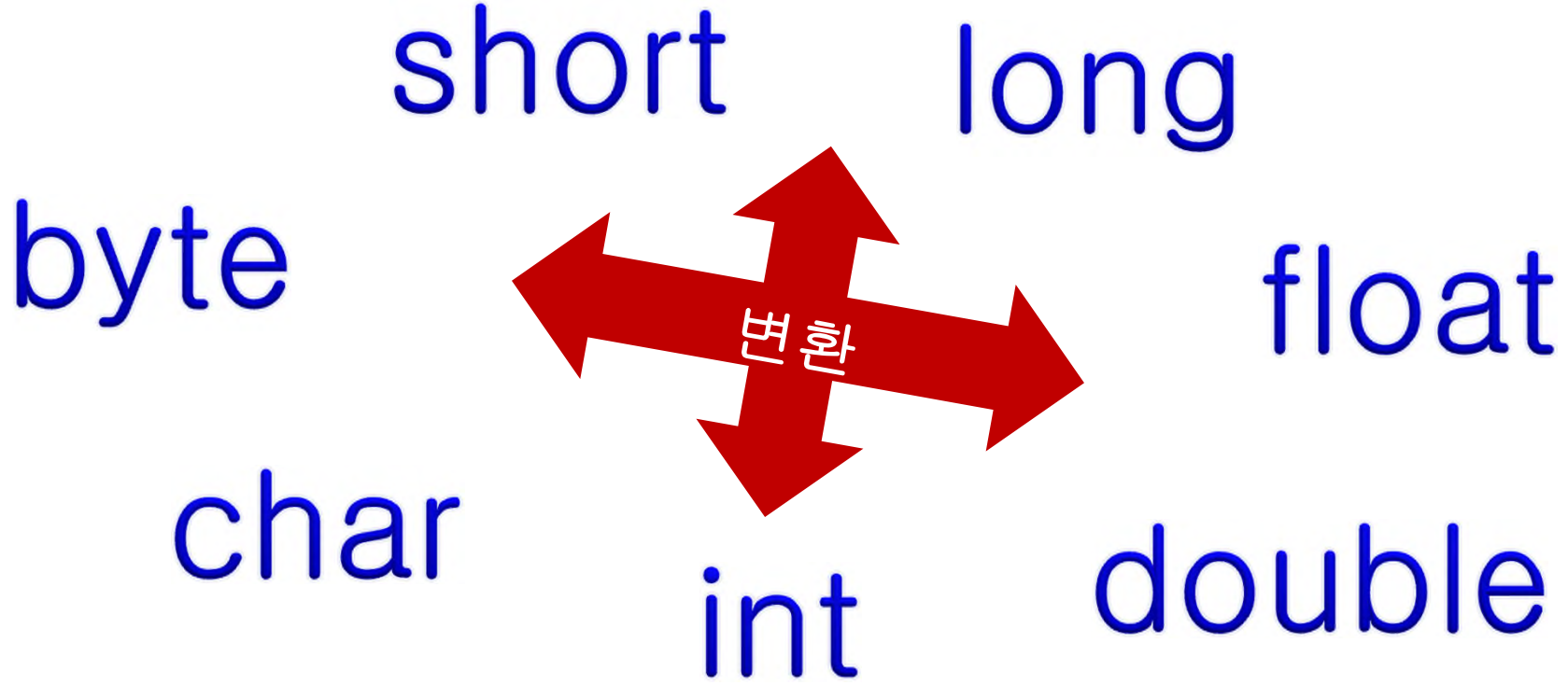
❖ 자료형 변수명 = 초기값; (초기화)

```
int age = 20;
```



Promotion/Casting 형 변환

❖ 자료형 변환
- 특정 자료형의 값을 다른 자료형에 대입하는 방식
- boolean, String은 형 변환 불가능



Promotion / Casting

❖ Promotion (자동 형 변환)

✓ 큰 자료형에 작은 자료형을 대입 하는 것

- byte a = 10;
- int b = a;

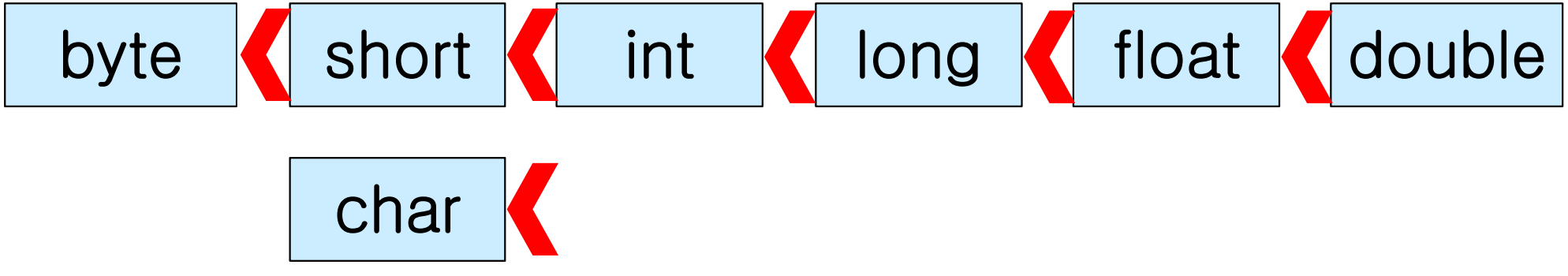
❖ Casting (강제 형 변환)

✓ 작은 자료형에 큰 자료형을 대입 되는 것

- int a = 10;
- byte b = (byte)a; (Casting)변수 or 상수

❖ 자료형 크기 구분

✓ 정수형 < 실수형



Quiz

다음 선언된 변수가 형 변환 된다면 어떻게 될 것인가?

```
public static void main(String []args){
    byte by = 10;
    short sh = 20;
    char ch = 65;
    int in = 30;
    long lo = 40L;
    float fl = 3.14F;
    double dou = 123.456;
}
```

QUIZ>>

1. int a = by ➔
2. char b = dou; ➔
3. long c = sh; ➔
4. float d = in; ➔
5. short e = ch; ➔
6. double f = by; ➔

Quiz

다음 선언된 변수가 형 변환 된다면 어떻게 될 것인가?

```
public static void main(String []args){
    byte by = 10;
    short sh = 20;
    char ch = 65;
    int in = 30;
    long lo = 40L;
    float fl = 3.14F;
    double dou = 123.456;
}
```

QUIZ>>

1. int a = by ➔ by
2. char b = dou; ➔ (char)dou
3. long c = sh; ➔ sh
4. float d = in; ➔ in
5. short e = ch; ➔ (short)ch
6. double f = by; ➔ by

Wrapper class

❖ 자료를 효율적으로 관리하면서 은닉화를 위해 만들어진 자료형 대체 클래스

✓ 자바는 OOP(Object Oriented Programing)기반 언어

✓ 클래스의 특징 중 '은닉화'개념 적용

✓ prototype의 자료형은 할당된 메모리 크기로 짐작이 가능

✓ 기본 자료형에 대한 은닉화를 위해 Wrapper class가 만들어 졌다.

✓ Wrapper class로 만든 객체는 무조건 4byte로 확인된다.

❖ Wrapper class

Boolean	Byte	Character	Short	Integer	Long	Float	Double

❖ Wrapper class의 객체는 기본 자료형보다 많은 기능을 이용 가능

변수의 초기화

- ❖ 자료형을 이용하여 메모리에 변수를 할당
- ❖ stack영역(지역 변수)
  - ▪ 자동으로 초기화 되지 않는다.(직접 초기화 필요)
- ❖ heap영역(동적할당)
  - ▪ 자동으로 각 자료형의 기본 값으로 초기화 된다.(아래 표 참조)

자료형	default 값
boolean	false
char	'₩u0000'
byte	0
short	0
int	0
long	0L
float	0.0f
double	0.0d or 0.0
참조형 변수	null

Quiz1

❖ 기본 자료형으로 변수를 생성 하고 값 대입하기 실습

- 이름을 저장할 수 있는 변수를 생성하고 자신의 이름으로 초기화하시오.
- 나이를 저장할 수 있는 변수를 생성하고 자신의 나이로 초기화하시오.
- 실수(소수)를 저장할 수 있는 변수를 생성하고 0으로 초기화하시오.
- 위에서 만든 변수에 3.14를 저장하시오.
- float형의 변수를 생성하고 위에서 만든 값을 대입하시오.

Quiz2

❖ Wrapper Class로 변수(객체) 생성

- 이름을 저장할 수 있는 객체를 생성하고 자신의 이름으로 초기화하시오.
- 나이를 저장할 수 있는 객체를 생성하고 자신의 나이로 초기화하시오.
- 실수(소수)를 저장할 수 있는 객체를 생성하고 0으로 초기화하시오.
- 위에서 만든 객체에 3.14를 저장하시오.
- Float형의 객체를 생성하고 위에서 만든 값을 대입하시오.