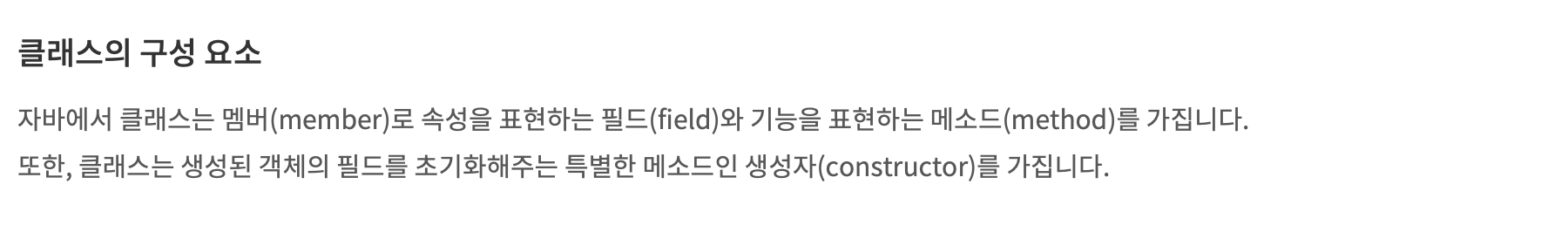
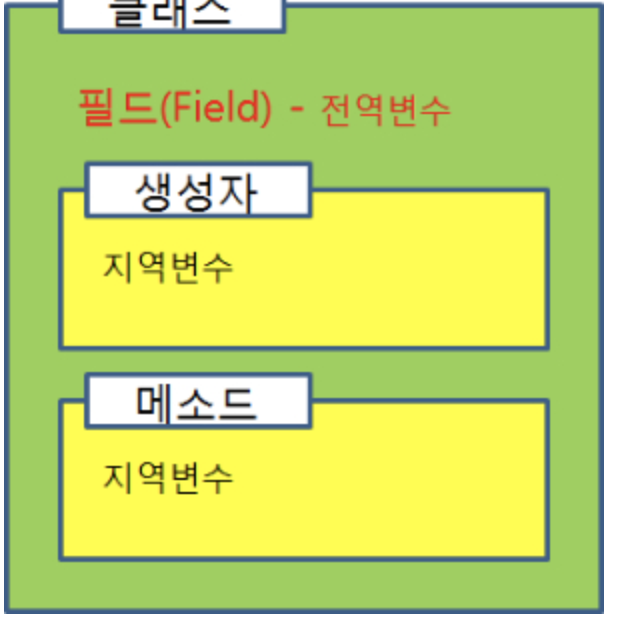
2주차 문제 풀이

ch.6 클래스

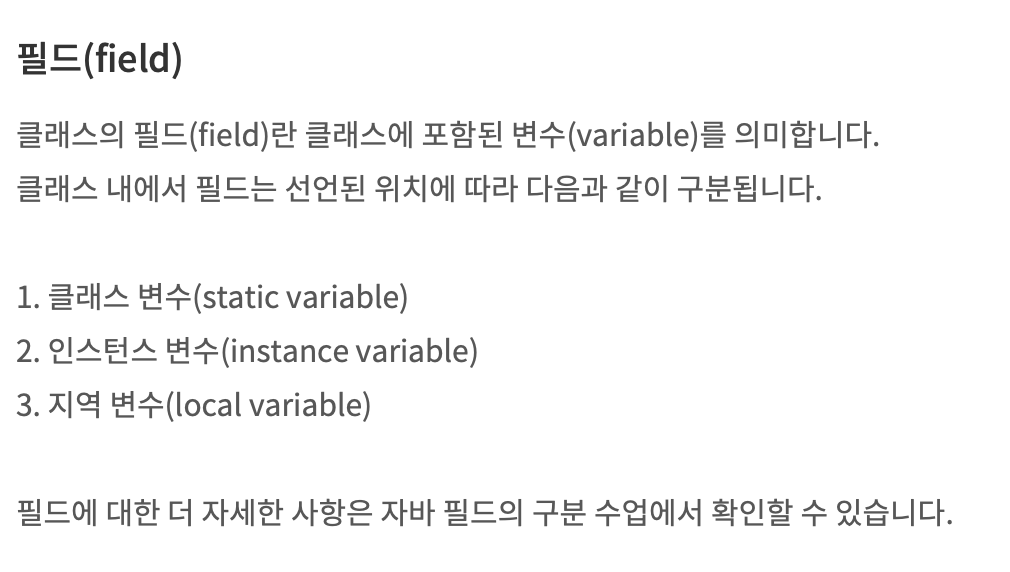
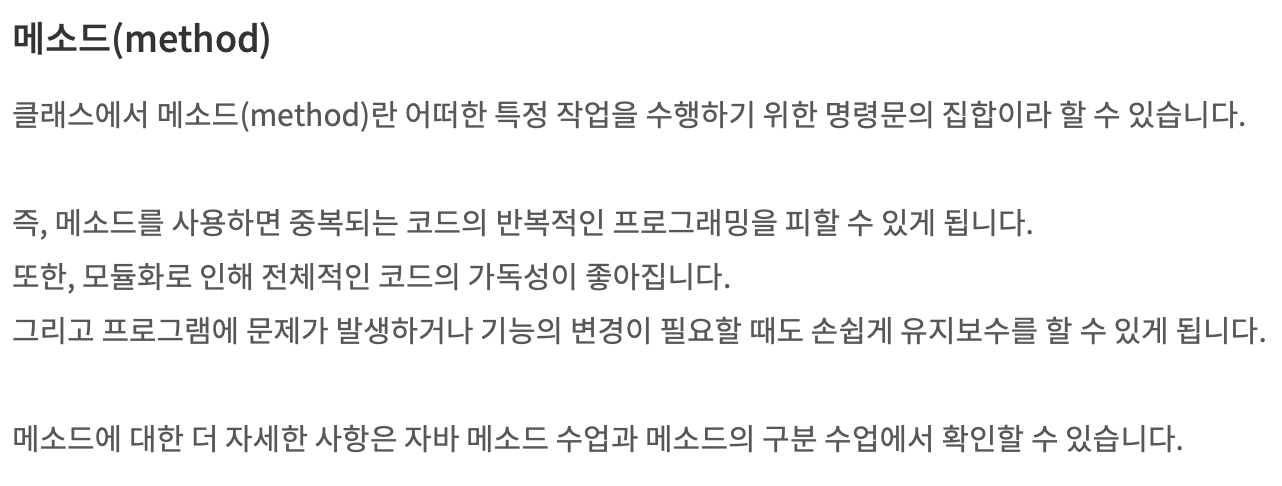
2. 클래스의 구성 멤버가 아닌 것은?

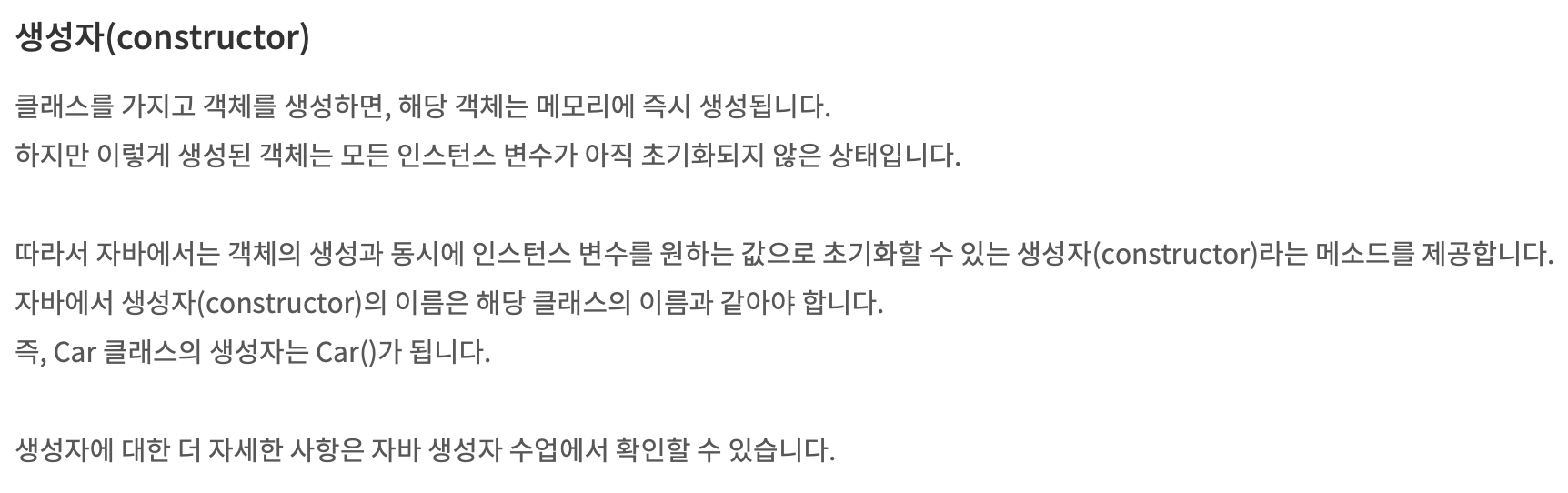
* 필드 : 객체의 상태 정보를 저장. 객체 데이터가 저장되는 곳. - 전역변수
* 생성자(c0nstruct0r)
* 메소드 : 함수와 같은 역할.
* 로컬 변수



* 

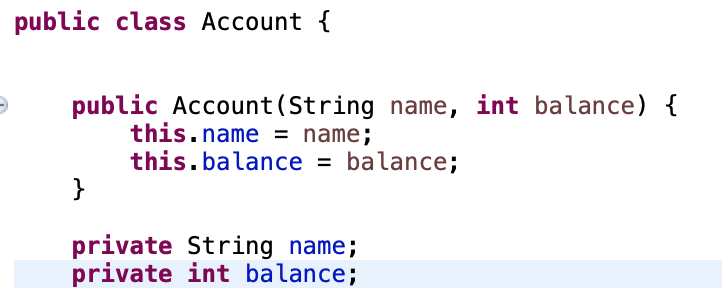
3. 필드, 생성자, 메소드에 대한 설명으로 틀린것은?

* 필드는 객체의 데이터를 저장한다.
* 생성자는 객체의 초기화를 담당한다.
* 메소드는 객체의 동작부분으로, 실행코드를 가지고 있는 블록이다.
* 클래스는 반드시 필드와 메소드를 가져야 한다. => 빈클래스 가능. 메인은 메소드인듯.(그런듯) 모든 클래스는 이라는 말이 없음.
* 
* 



4. 필드에 대한 설명으로 틀린 것은?

* 필드는 메소드에서 사용할 수 있다.
* 인스턴스 필드 초기화는 생성자에서 할 수 있다.
* 필드는 반드시 생성자 선언 전에 선언되어야한다.(객체 지향이라 상관x)
* 필드는 초기값을 주지 않더라도 기본값으로 자동 초기화된다.



[4번 반례(good)]

5. 생성자에 대한 설명으로 틀린 것은?

* 객체를 생성하려면 생성자 호출이 반드시 필요한 것은 아니다. // 객체를 생성하기 위해서는 생성자를 호출해야지 새로 만들 수 있음. 없어도 결국 컴파일러가 생성함

// 일반적으로 설계도인 클래스가 구체적인 실체인 인스턴스가 되었을 때 객체라고 부른다. 인스턴스가 되려면 생성자 호출을 한다. -이응창

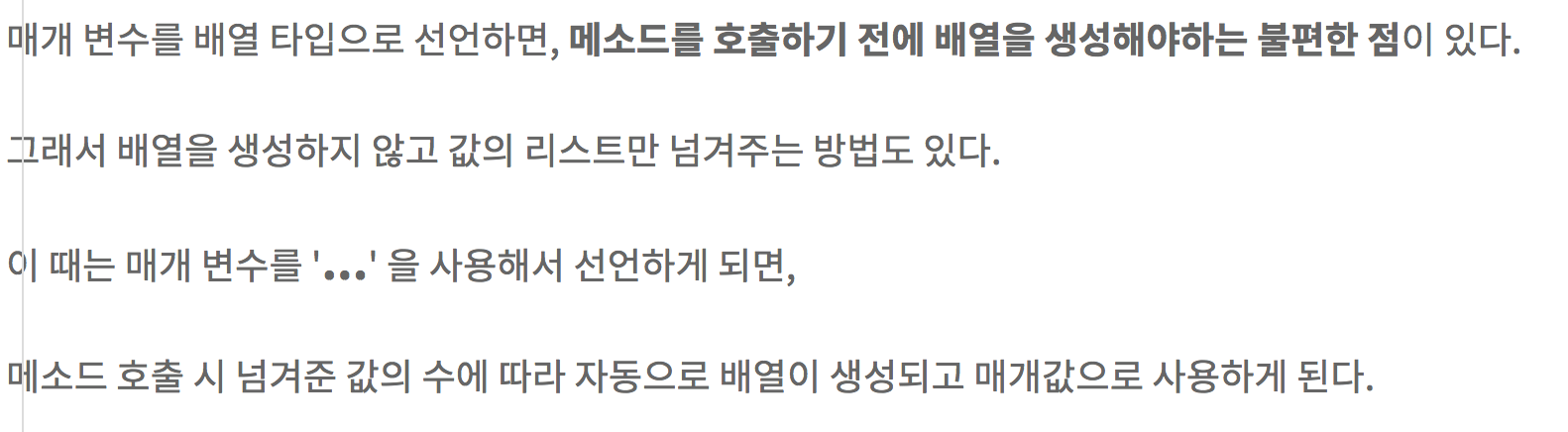
* 생성자는 다른 생성자를 호출하기 위해 this()를 사용할 수 있다.
* 생성자가 선언되지 않으면 컴파일러가 기본 생성자를 추가한다. // 1번의 반례
* 외부에서 객체를 생성할 수 없도록 생성자에 private 접근 제한자를 붙일 수 있다. //다른 객체에서 접근 못함. getter, setter는

6. 메소드에 대한 설명으로 틀린 것은?

* 리턴값이 없는 메소드는 리턴 타입을 void로 해야한다.
* 리턴 타입이 있는 메소드는 리턴 값을 지정하기 위해 반드시 return 문이 있어야한다.
* 매개값의 수를 모를 경우 “...”를 이용해서 매개변수를 선언할 수 있다.

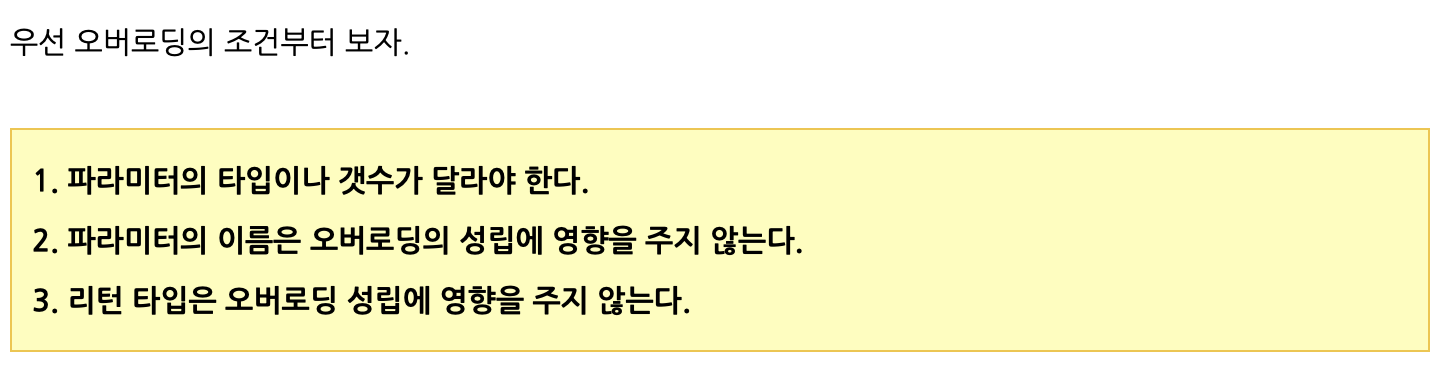
// 매개변수를 “...”으로 선언하면, 메소드 호출 시 넘겨준 값의 수에 따라 자동으로 배열이 생성되고 매개값으로 사용된다.

* 메소드의 이름은 중복해서 선언할 수 없다.(중복가능해서 오버로딩 되지만, 매개변수의 타입,개수,순서가 달라야한다.)



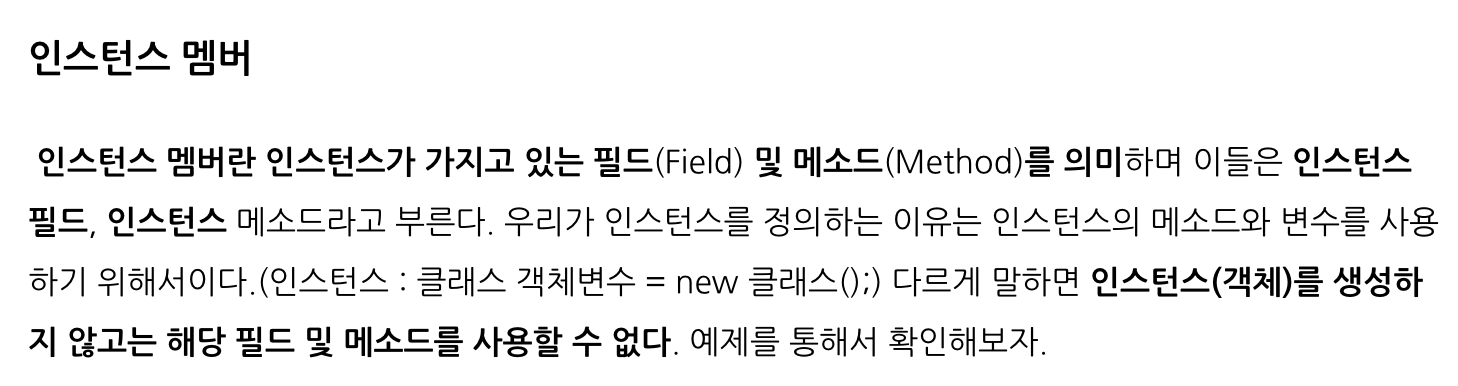
-----3번 근거

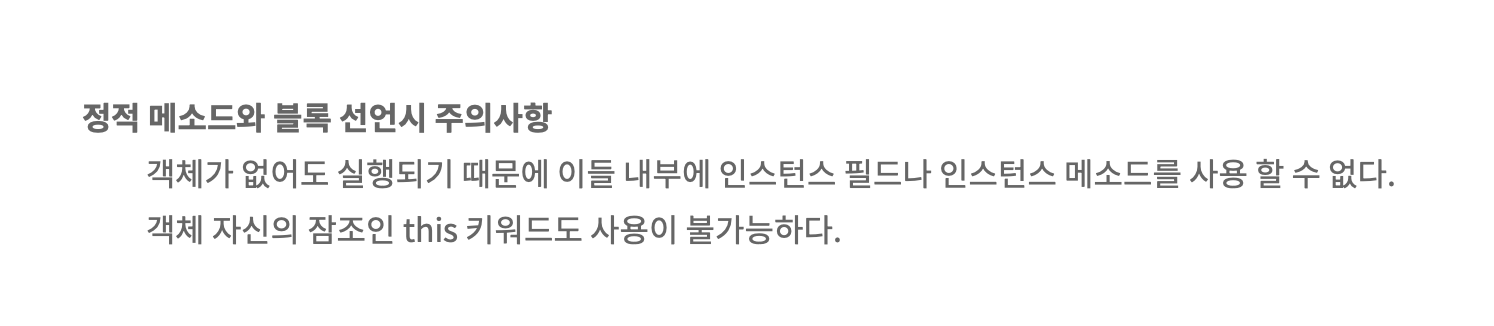
7. 메소드 오버로딩에 대한 설명으로 틀린 것은?

* 동일한 이름의 메소드를 여러 개 선언 하는 것을 말한다.
* 반드시 리턴 타입이 달라야 한다.(3번이 있기에 2번이 가능)
* 매개 변수의 타입, 수, 순서를 다르게 선언해야 한다.
* 매개 값의 타임 및 수에 따라 호출될 메소드가 선택된다.

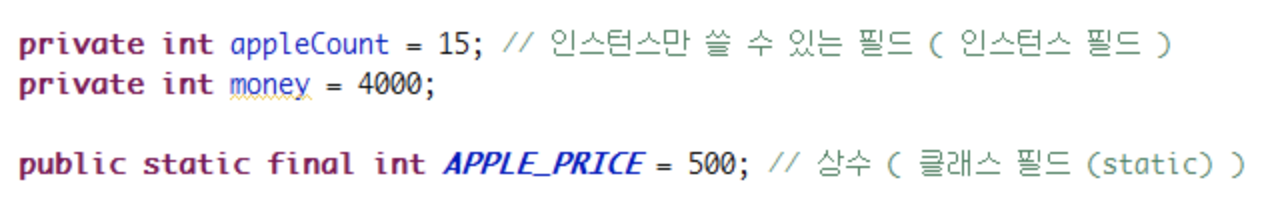
8. 인스턴스 멤버와 정적 멤버에 대한 설명으로 틀린 것은?

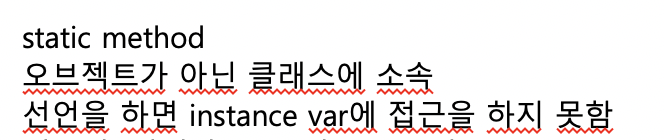
* 정적 멤버는 static 으로 선언된 필드와 메소드를 말한다.
* 인스턴스 필드는 생성자 및 정적 블록에서 초기화될 수 있다. //정적 블록에서 초기화 할 수 없음.
* 정적 필드와 정적 메소드는 객체 생성 없이 클래스를 통해 접근할 수 있다. //System.out.println
* 인스턴스 필드와 메소드는 객체를 생성하고 사용해야 한다. //메모리 할당



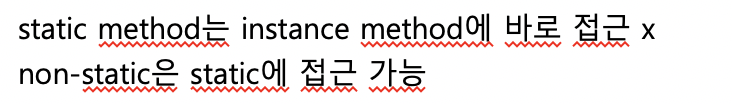


static == global

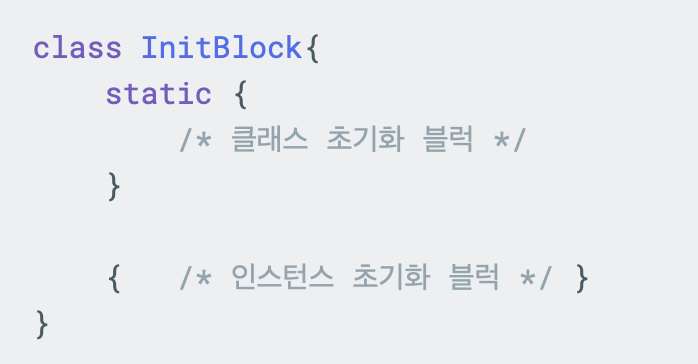




//정적 메소드와 블록 선언시 주의사항에 대한 첨부파일과 연계



//객프 필기



#include <stdio.h>

int main(void)

{

asdf(string);

}

asdf가 사용자 지정 함수일 때 선언 전에 메인 함수에서 쓸 수 없는 것 처럼?

9. final 필드와 상수(static final)에 대한 설명으로 틀린 것은?

* final 필드와 상수는 초기값이 저장되면 값을 변경할 수 없다.
* final 필드와 상수(static final)는 생성자에서 초기화될 수 있다.(final 필드는 되는데 상수는 안됨)
* 상수의 이름은 대문자로 작성하는 것이 관례이다.
* 상수는 객체 생성 없이 클래스를 통해 사용할 수 있다.

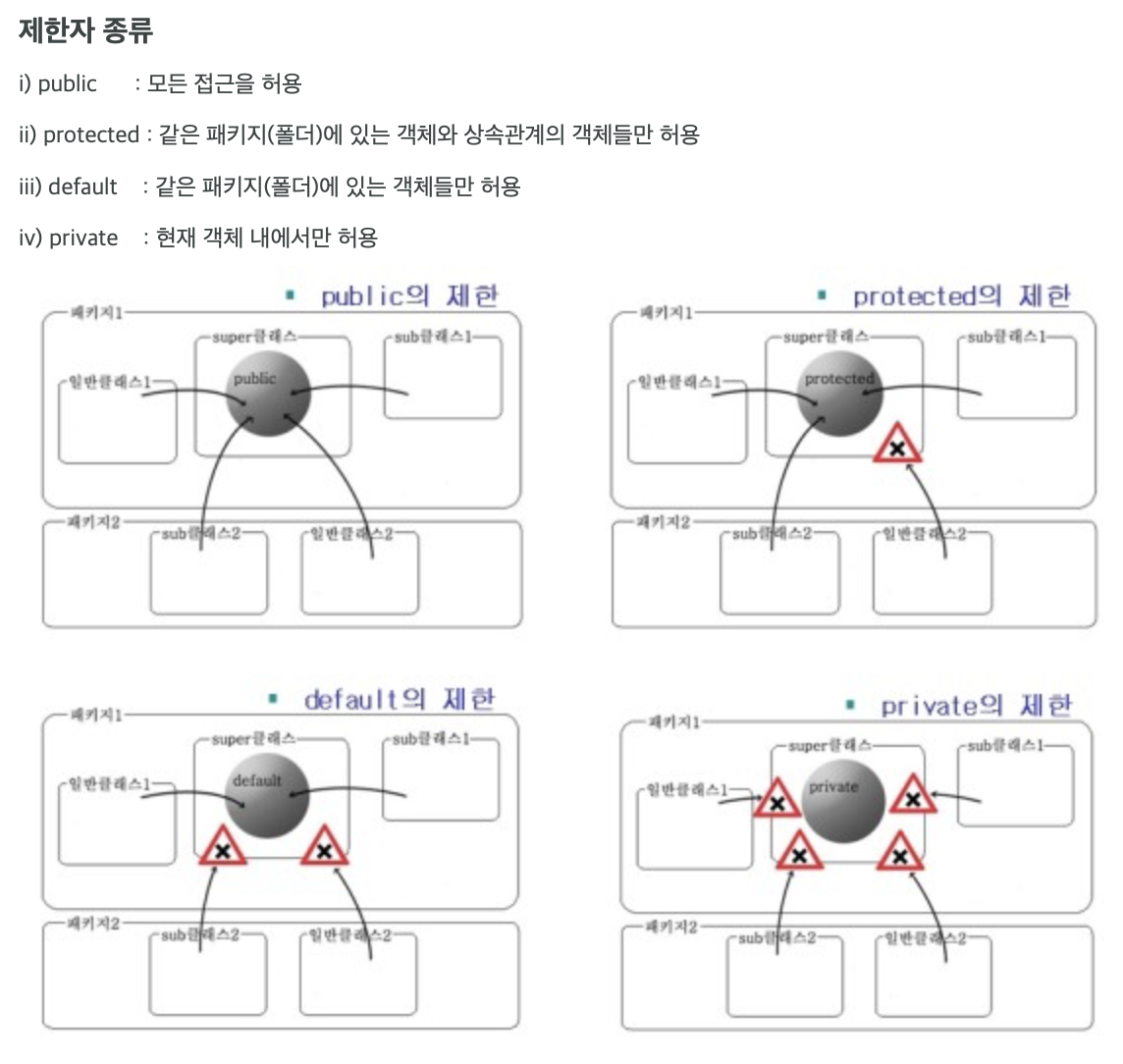
final은 한 번 초기화 되면 값이 바뀔 수 없다.

11. 접근 제한에 대한 설명으로 틀린 것은?

* 접근 제한자는 클래스, 필드, 생성자, 메소드의 사용을 제한한다.
* public 접근 제한은 아무런 제한 없이 해당 요소를 사용할 수 있게 한다.
* default 접근 제한은 해당 클래스 내부에서만 사용을 허가한다.//패키지 내부임.

//다른 패키지에서 사용못하게 하는거래~

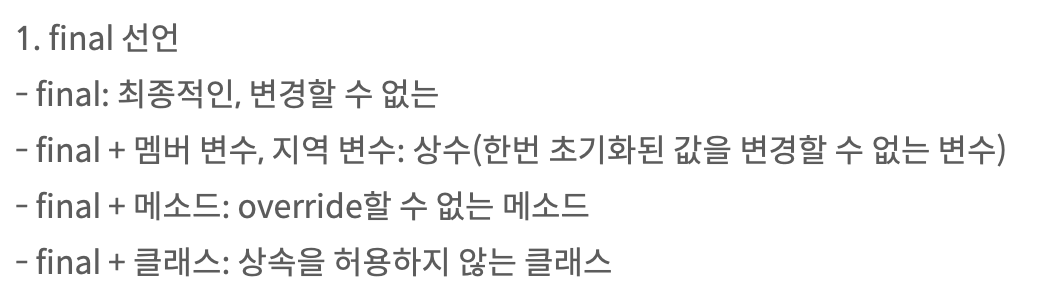
* 외부에서 접근하지 못하도록 하려면 private 접근 제한을 해야한다.



ch.7 상속

1. 자바의 상속에 대한 설명 중 틀린 것은?

* 자바는 다중 상속을 허용한다. //여러 부모가 있는 건 안된다.
* 부모의 메소드를 자식 클래스에서 재정의(오버라이딩) 할 수 있다.
* 부모의 private 접근 제한을 갖는 필드와 메소드는 상속의 대상이 아니다. //private는 상속의 대상이 아니다.
* final 클래스는 상속할 수 없고, final 메소드는 오버라이딩 할 수 없다.



2. 