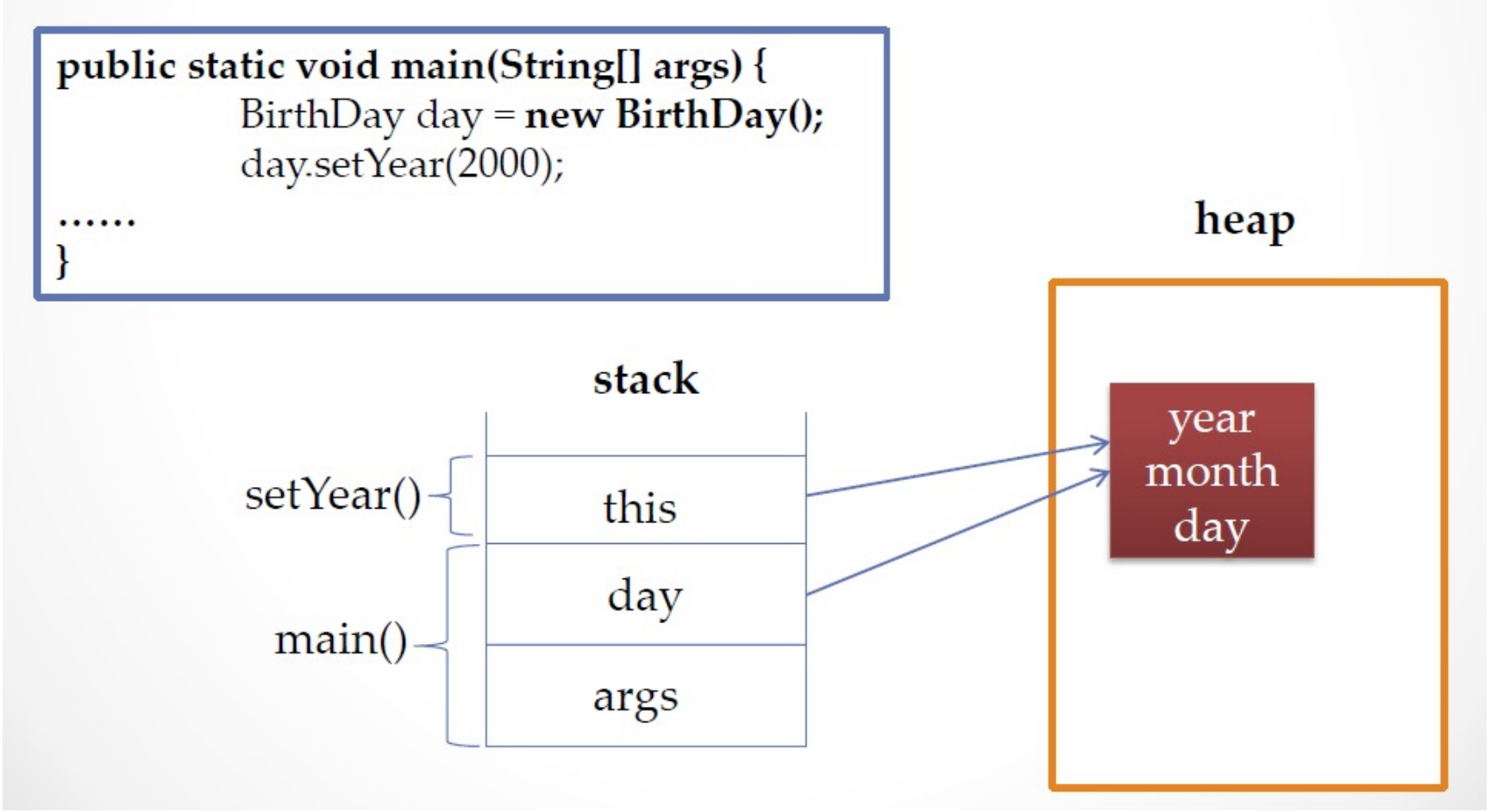
클래스와 객체 2

this 가 하는 일

- 자신의 메모리을 가리킴
- 생성자에서 다른 생성자를 호출
- 자신의 주소를 반환 함
-

자신의 메모리를 가리키는 this

- 생성된 인스턴스 스스로를 가리키는 예약어



자신의 메모리를 가리키는 this

```
public Person(String name , int age){
        this.name = name;
        this.age = age;
}
```

note : 위 코드에서 this를 생략하게 되면 name이나 age는 파라미터로 사용되는 name과 age로 인식된다.

생성자에서 다른 생성자를 호출 하는 this

```
public Person(){
        this("이름없음", 1);
}

public Person(String name , int age){
        this.name = name;
        this.age = age;
}
```

note : *this*를 이용하여 다른 생성자를 호출할 때는 그 이전에 어떠한 *statement*도 사용할 수 없다.
위와 같이 생성자가 여러 개 이고 파라미터만 다른 경우 *constructor overloading* 이라고 한다.

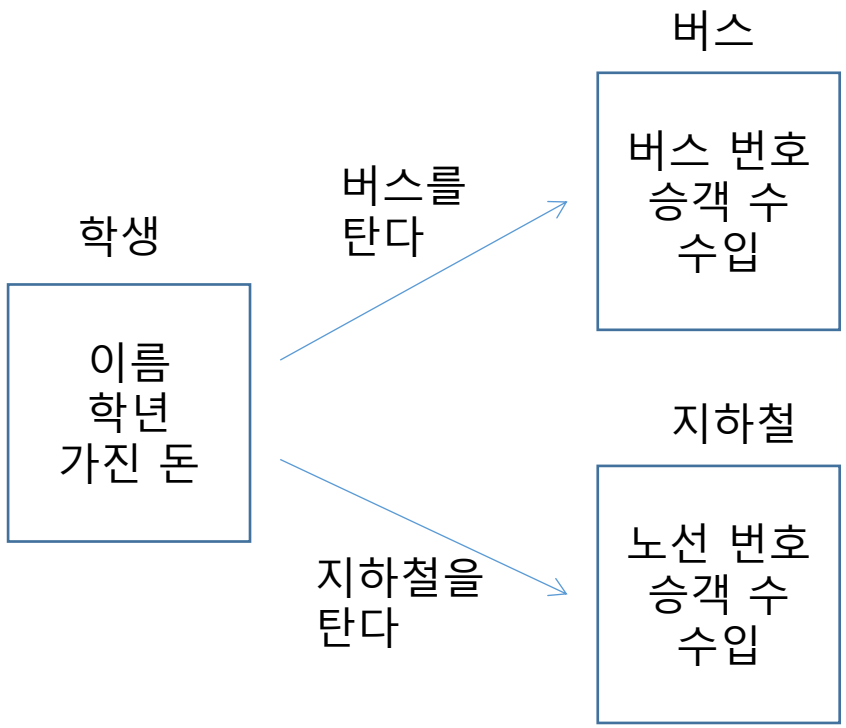
자신의 주소를 반환하는 this

```
Person getPerson(){
        return this;
}

public static void main(String[] args) {

        Person p = new Person();
        p.name = "James";
        p.married = true;
        p.numberOfChild = 3;
        p.age = 40;

        Person p2 = p.getPerson();
        System.out.println(p2.age);
}
```

note : 여기서의 this는 reference value를 return 한다.

객체 간의 협력

- 학생이 버스나 지하철을 가는 상황을 객체 지향으로 프로그래밍

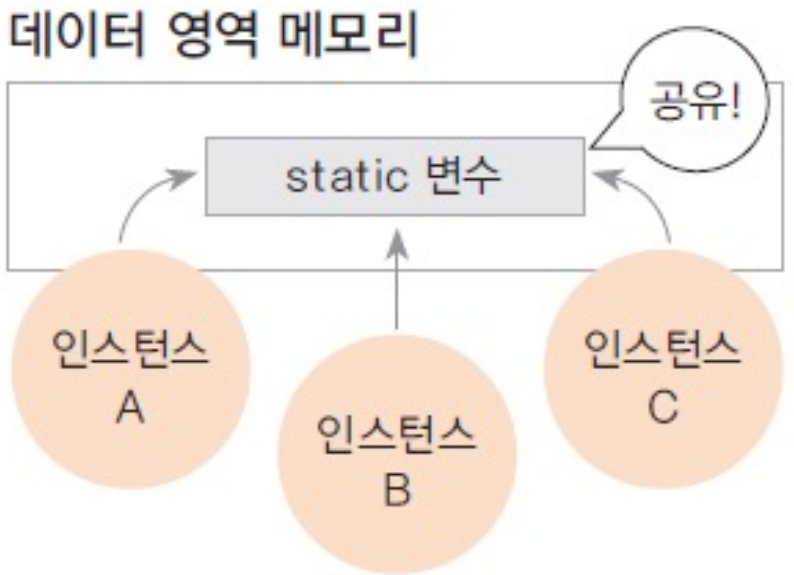


예제 코드는 책을 참고 합니다.

static 변수

- static 변수의 정의와 사용 방법

- 여러 개의 인스턴스가 같은 메모리의 값을 공유하기 위해 사용



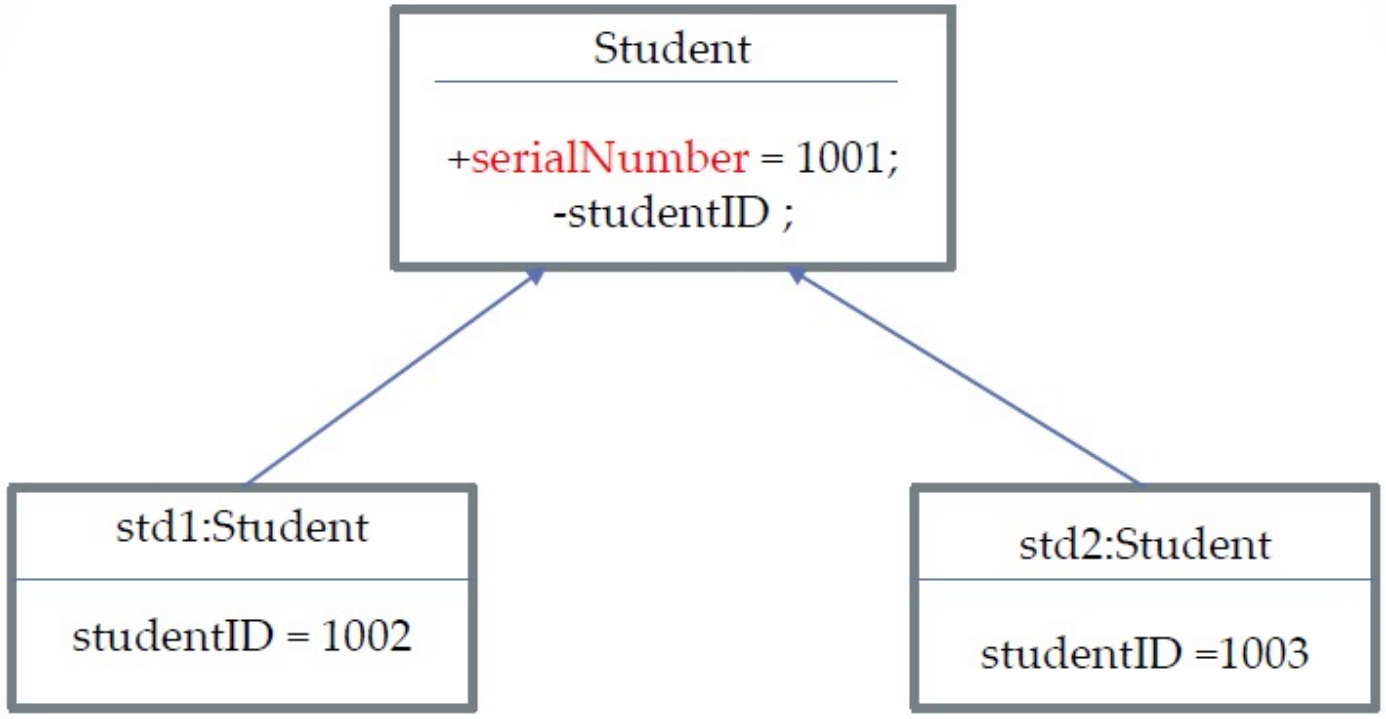
static 변수

- static 변수는 인스턴스가 생성될 때 마다 다른 메모리를 가지는 것이 아니라 프로그램이 메모리에 적재(load) 될때 데이터 영역의 메모리에 생성 됨
- 따라서 인스턴스의 생성과 관계없이 클래스 이름으로 직접 참조 함

```
Student.serialNum = 100; //serailNum이 static 변수
```

- 클래스 변수 라고도 함
- 멤버변수는 다른 말로 인스턴스 변수라고 함

static 변수 vs. 인스턴스 변수

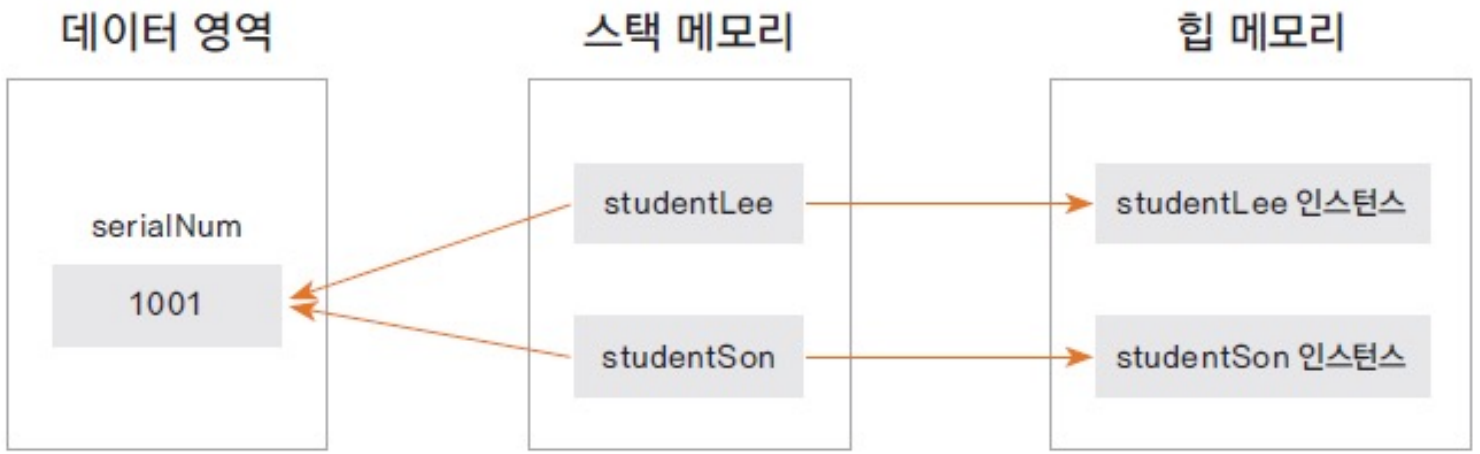


note : *serialNumber를 static 으로 선언하면 모든 student instance에 대해 하나의 변수로 유지 되고 이러한 변수를 class 변수라 한다.*

static 변수 예

- 여러 인스턴스가 하나의 메모리 값을 공유 할 때 필요
- 학생이 생성될 때마다 학번이 증가해야 하는 경우
- 기준이 되는 값은 static 변수로 생성하여 유지 함

- 각 학생이 생성될 때 마다 static 변수 값을 복사해 와서 하나 증가 시킨 값을 생성된 인스턴스의 학번 변수에 저장해 줌



static 메서드

- 클래스 메서드 라고도 함
- 메서드에 static 키워드를 사용하여 구현
- 주로 static 변수를 위한 기능 제공

- static 메서드에서 인스턴스 변수를 사용할 수 없음
- static 메서드도 인스턴스의 생성과 관계 없이 클래스 이름으로 직접 메서드 호출

```
Student.getSerialNum(); //getSerialNum() 이 static 메서드
```

- 인스턴스의 변수의 경우 꼭 인스턴스가 먼저 생성되어야 하므로 static메서드에서는 생성이 불확실한 인스턴스 변수를 사용할 수 없음

변수의 유효 범위

변수 유형	선언 위치	사용 범위	메모리	생성과 소멸
지역 변수 (로컬 변수)	함수 내부에 선언	함수 내부에서만 사용	스택	함수가 호출될 때 생성되고 함수가 끝나면 소멸함
멤버 변수 (인스턴스 변수)	클래스 멤버 변수로 선언	클래스 내부에서 사용하고 private이 아니면 참조 변수로 다른 클래스에서 사용 가능	힙	인스턴스가 생성될 때 힙에 생성되고, 가비지 컬렉터가 메모리를 수거할 때 소멸됨
static 변수 (클래스 변수)	static 예약어를 사용하여 클래스 내부에 선언	클래스 내부에서 사용하고 private이 아니면 클래스 이름으로 다른 클래스에서 사용 가능	데이터 영역	프로그램이 처음 시작할 때 상수와 함께 데이터 영역에 생성되고 프로그램이 끝나고 메모리를 해제할 때 소멸됨

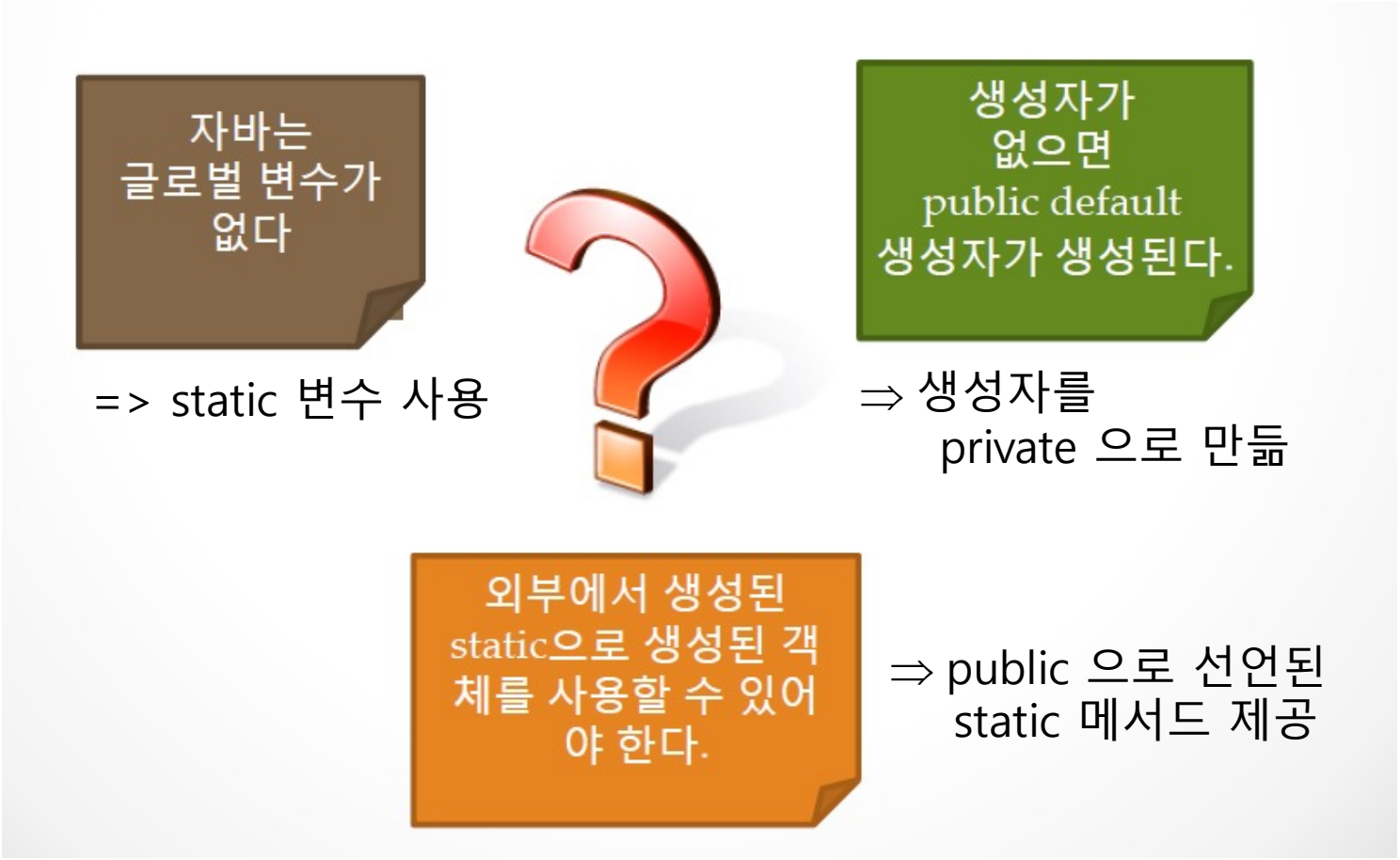
static 응용 : singleton 패턴

자동차 회사가 있고
자동차 회사에는 직원들이 있고
자동차 회사에는 여러 개의 공장들이
있고, 생산된 자동차를 운반하는
운반차들이 있다

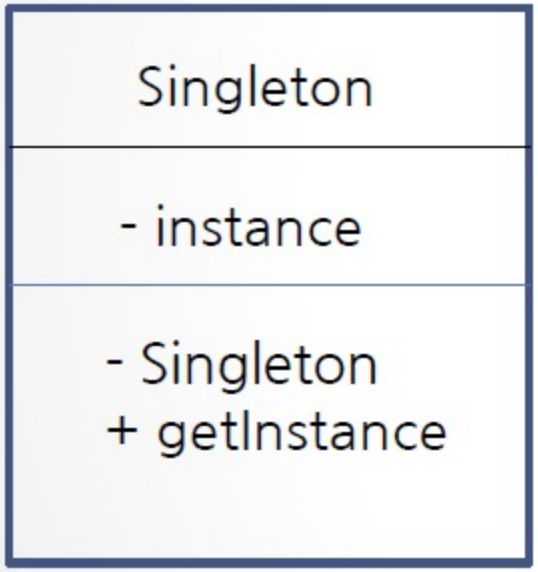
위에서 객체는 무엇이고, 이 중에
자동차 회사 시스템을 만들 때
단 한 개만이 존재하는 객체는 무엇인가?

static 응용 : singleton 패턴

- 전 시스템에 단 하나의 인스턴스만이 존재하도록 구현하는 방식



static 응용 : singleton 패턴




```
public class Singleton {
   private static Singleton singleton = new Singleton();//null;

   private Singleton() {
      System.out.println("인스턴스를 생성했습니다.");
   }

   public static Singleton getInstance() {
   if (singleton == null )
   singleton = new Singleton();

   return singleton;
   }
}
```

감사합니다.

끝