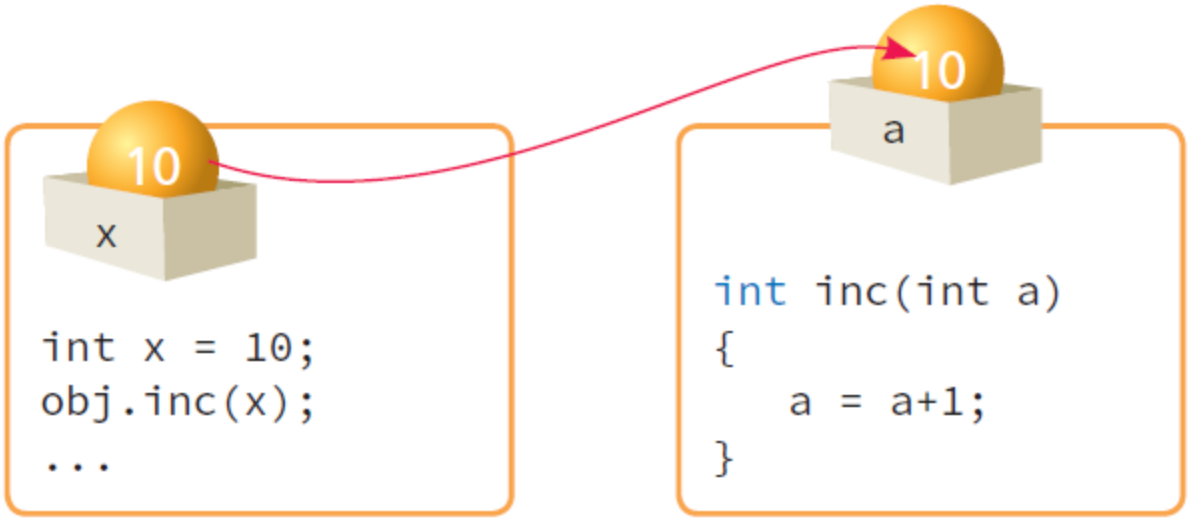
어서와 Java는 처음이지!

제6장 클래스, 메소드 심층연구

- 메소드로 기초 변수 전달
- 메소드로 객체 전달
- 메소드로 객체 반환
- 정적 멤버
- 내장 클래스

메소드로 기초형 변수가 전달되는 경우

- 기초형 변수가 전달되는 경우



메소드로 기초형 변수가 전달되는 경우

```
public class MyCounter {
        int value;
        void inc(int a) {
                a = a + 1;
        }
}
```

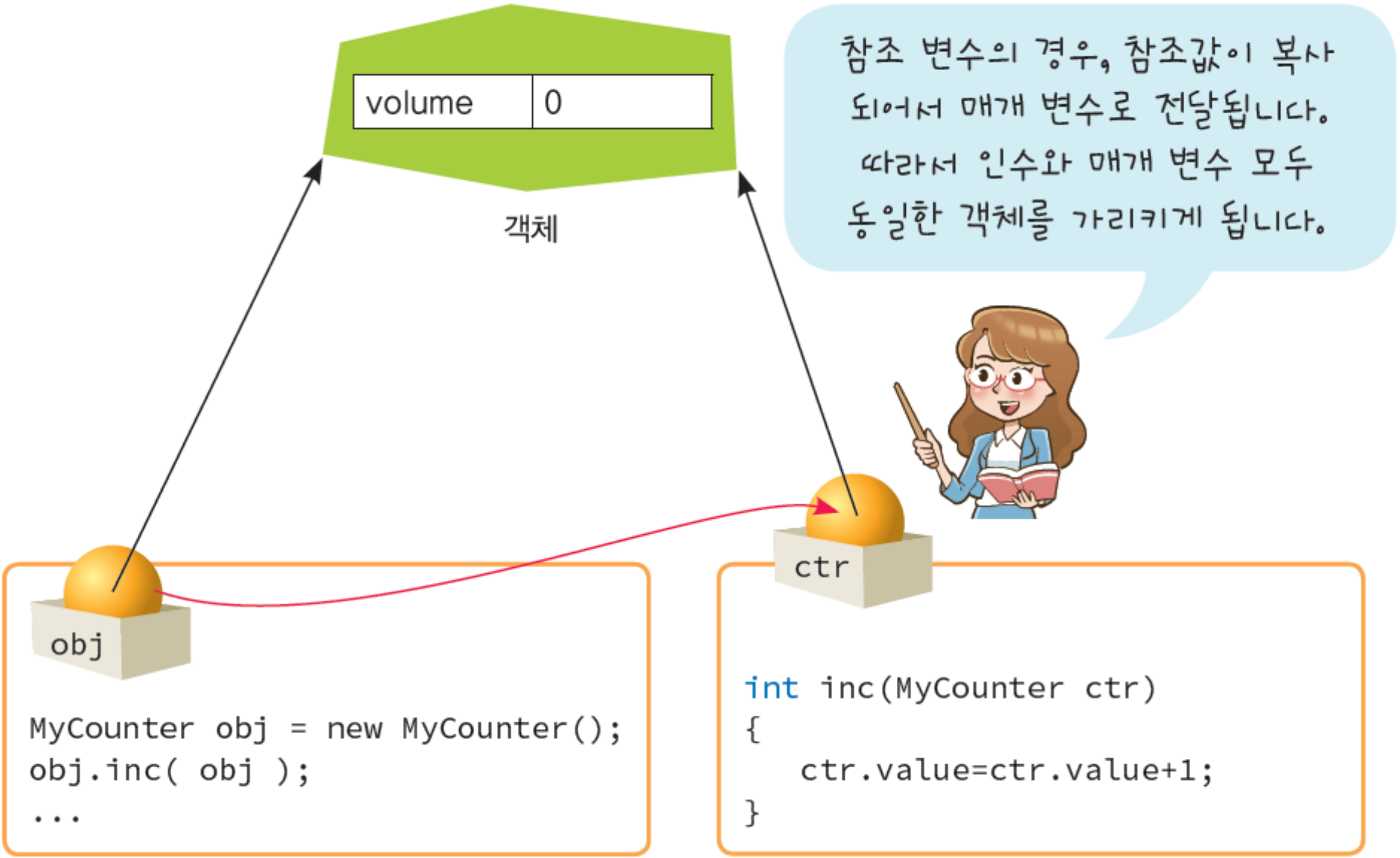
```
public class MyCounterTest1 {
        public static void main(String args[]) {
                MyCounter obj = new MyCounter();
                int x = 10;
                obj.inc(x);
                System.out.println("x = " + x);
        }
}
```


```
x = 10
```

메소드로 객체가 전달되는 경우

- 객체를 메소드로 전달하게 되면:
  - 객체가 복사되어 전달되는 것이 아니고 참조 변수의 값이 복사되어서 전달된다.

메소드로 객체가 전달되는 경우



메소드로 객체가 전달되는 경우

```
class MyCounter {
        int value = 0;
        void inc(MyCounter ctr) {
                ctr.value = ctr.value + 1;
        }
}
```

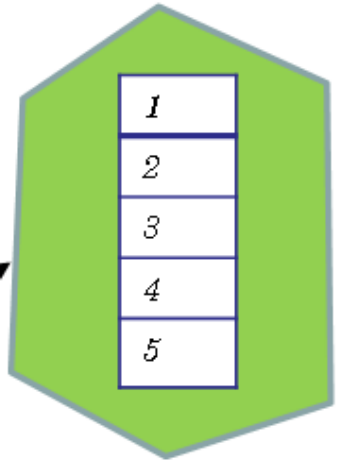
```
public class MyCounterTest2 {
        public static void main(String args[]) {
                MyCounter obj = new MyCounter();

                System.out.println("obj.value = " + obj.value);
                obj.inc(obj);
                System.out.println("obj.value = " + obj.value);
        }
}
```


```
obj.value = 0
obj.value = 1
```

메소드로 배열이 전달되는 경우

- 배열도 객체이기 때문에 배열을 전달하는 것은 배열 참조 변수를 복사하는 것이다.

메소드로 배열이 전달되는 경우



list

```
int[] list = { 1, 2, 3, 4, 5 };
ArrayProc obj = new ArrayProc();
obj.inc(list);
…
```

array

```
void inc(int[] array)
{
   …
   for (int i = 0; i < array.length; i++)

                array[i] = array[i] + 1;
}
```

- 사용자로부터 값을 받아서 배열에 채운 후에 배열에 저장된 모든 값의 평균을 구하여 출력하는 프로그램을 작성하여 보자.

array.ArrayProc
getValues(array: int[]): void getAverage(array: int[]): double

array.ArrayProcTest
STUDENTS: int
main(args: String[]): void


```
성적을 입력하시오:10
성적을 입력하시오:20
성적을 입력하시오:30
성적을 입력하시오:40
성적을 입력하시오:50
평균은 = 30.0
```

```
import java.util.Scanner;

public class ArrayProc {

	public void getValues(int[] array) {
		Scanner scan = new Scanner(System.in);
		for (int i = 0; i < array.length; i++) {
			System.out.print("성적을 입력하시오:");
			array[i] = scan.nextInt();
		}
	}
	public double getAverage(int[] array) {
		double total = 0;
		for (int i = 0; i < array.length; i++)
			total += array[i];
		return total / array.length;
	}
}
```

SOLUTION

```
public class ArrayProcTest {
        final static int STUDENTS = 5;

        public static void main(String[] args) {
                int[] scores = new int[STUDENTS];
                ArrayProc obj = new ArrayProc();
                obj.getValues(scores);
                System.out.println("평균은 = " + obj.getAverage(scores));
        }
}
```

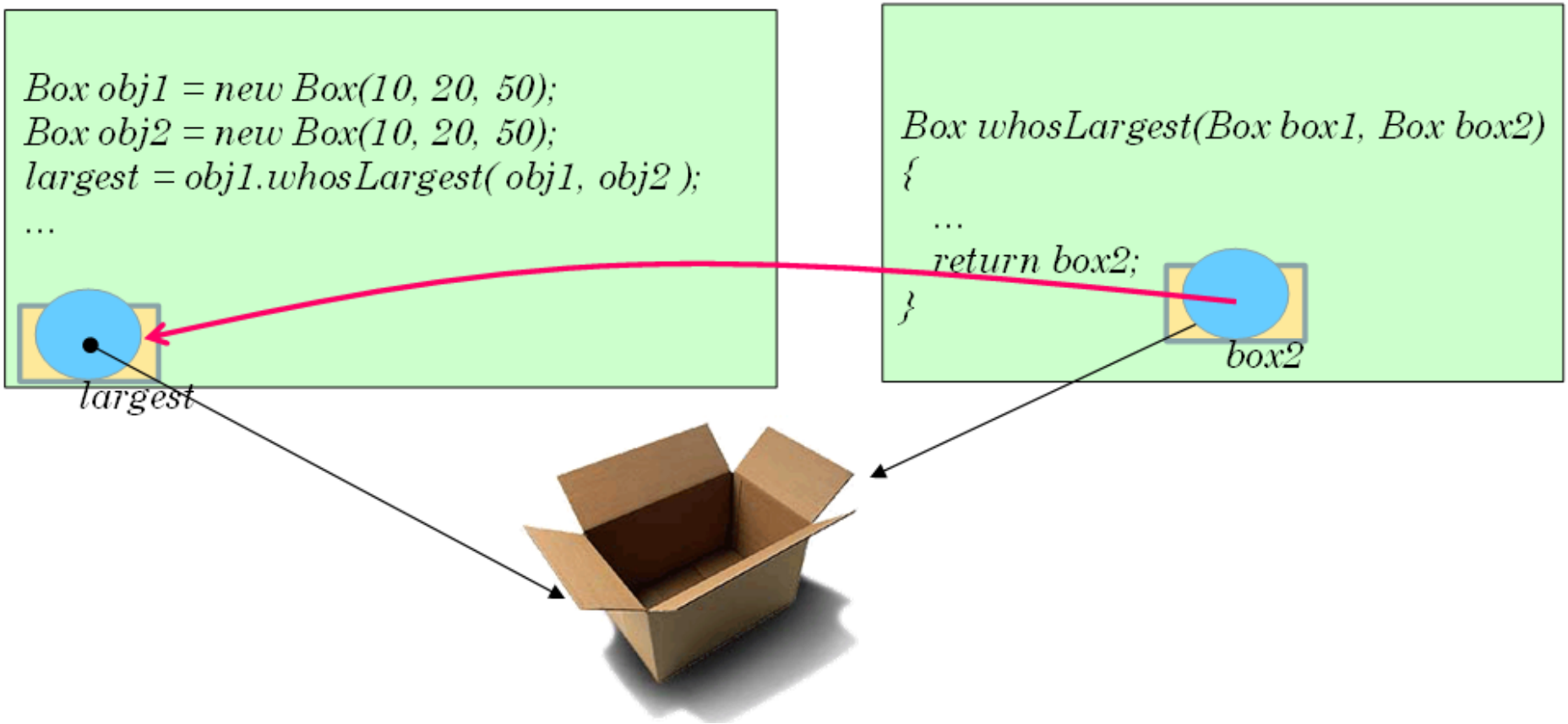
메소드에서 객체 반환하기

```
public class Box {
        int width, length, height;
        int volume;

        Box(int w, int l, int h) {
                width = w;
                length = l;
                height = h;
                volume = w * l * h;
        }
        Box whosLargest(Box box1, Box box2) {
                if (box1.volume > box2.volume)
                        return box1;
                else
                        return box2;
        }
 }
```

메소드에서 객체 반환하기

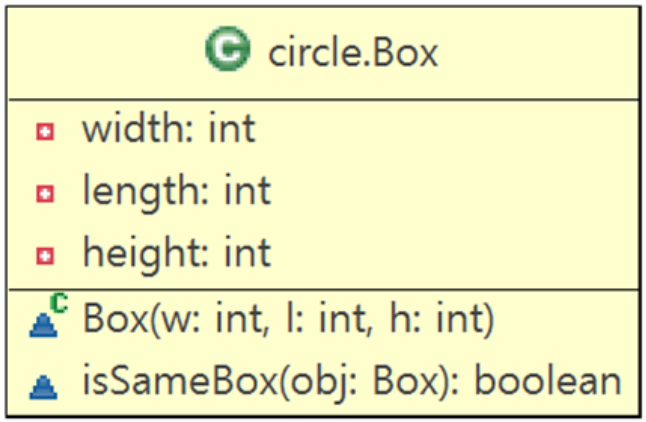
```
public class BoxTest {
        public static void main(String args[]) {
                Box obj1 = new Box(10, 20, 50);
                Box obj2 = new Box(10, 30, 30);

                Box largest = obj1.whosLargest(obj1, obj2);
                System.out.println("(" + largest.width + "," + largest.length + ","
                                + largest.height + ")");
        }
}
```

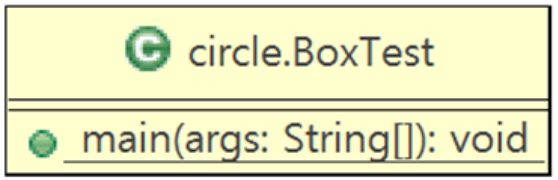
(10,20,50)


Box obj1 = new Box(10, 20, 50);
Box obj2 = new Box(10, 20, 50);
largest = obj1.whosLargest(obj1, obj2);
...
largest
Box whosLargest(Box box1, Box box2)
{
...
return box2;
}
box2

LAB: 같은 크기의 Box인지 확인하기

- 2개의 박스가 같은 치수인지를 확인하는 메소드 isSameBox()를 작성하여 보자. 만약 박스의 크기가 같으면 true를 반환하고 크기가 다르면 false를 반환한다. isSameBox()의 매개 변수는 객체 참조 변수가 된다.





SOLUTION

```
public class Box {
        int width, length, height;

        Box(int w, int l, int h) {
                width = w;
                length = l;
                height = h;
        }

        boolean isSameBox(Box obj) {
                if ((obj.width == width) & (obj.length == length)
                                & (obj.height == height))
                        return true;
                else
                        return false;
        }
}
```

SOLUTION

```
public class BoxTest {
        public static void main(String args[]) {
                Box obj1 = new Box(10, 20, 50);
                Box obj2 = new Box(10, 20, 50);

                System.out.println("obj1 == obj2 : " + obj1.isSameBox(obj2));
        }
}
```


```
obj1 == obj2 : true
```

객체

obj2

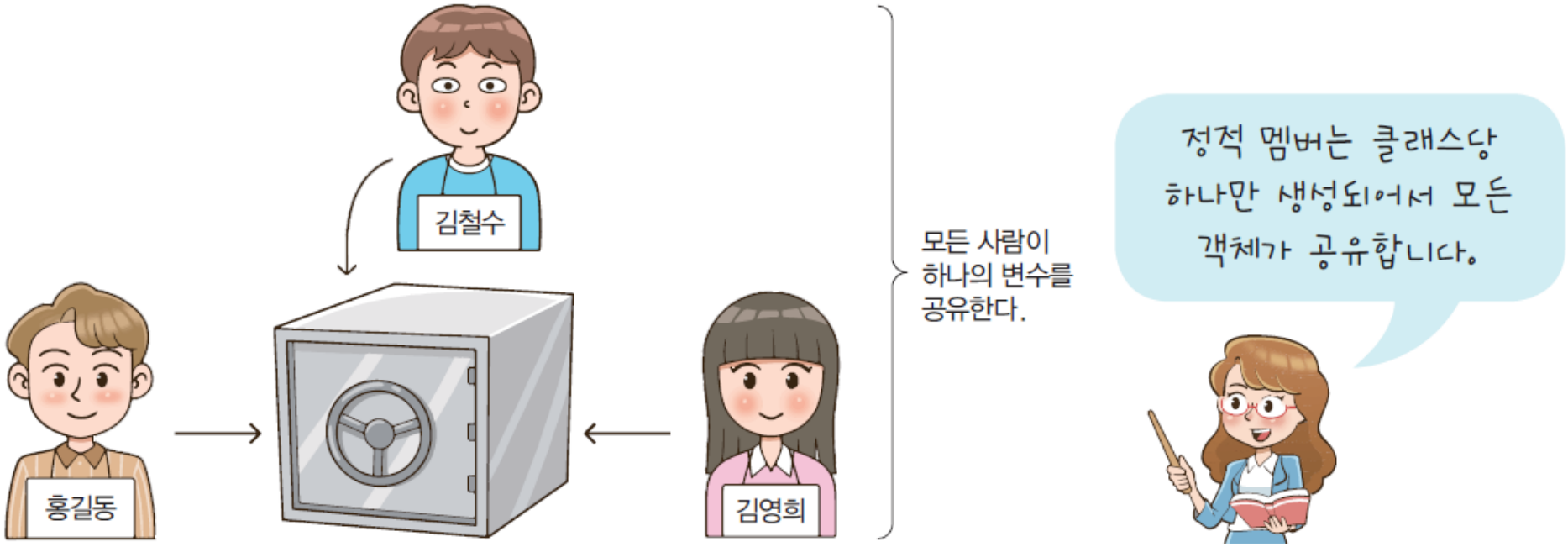
```
Box obj1 = new Box(10, 20, 50);
Box obj2 = new Box(10, 20, 50);
obj1.isSameBox( obj2 );
…
```

obj

```
boolean isSameBox(Box obj)
{
   …
}
```

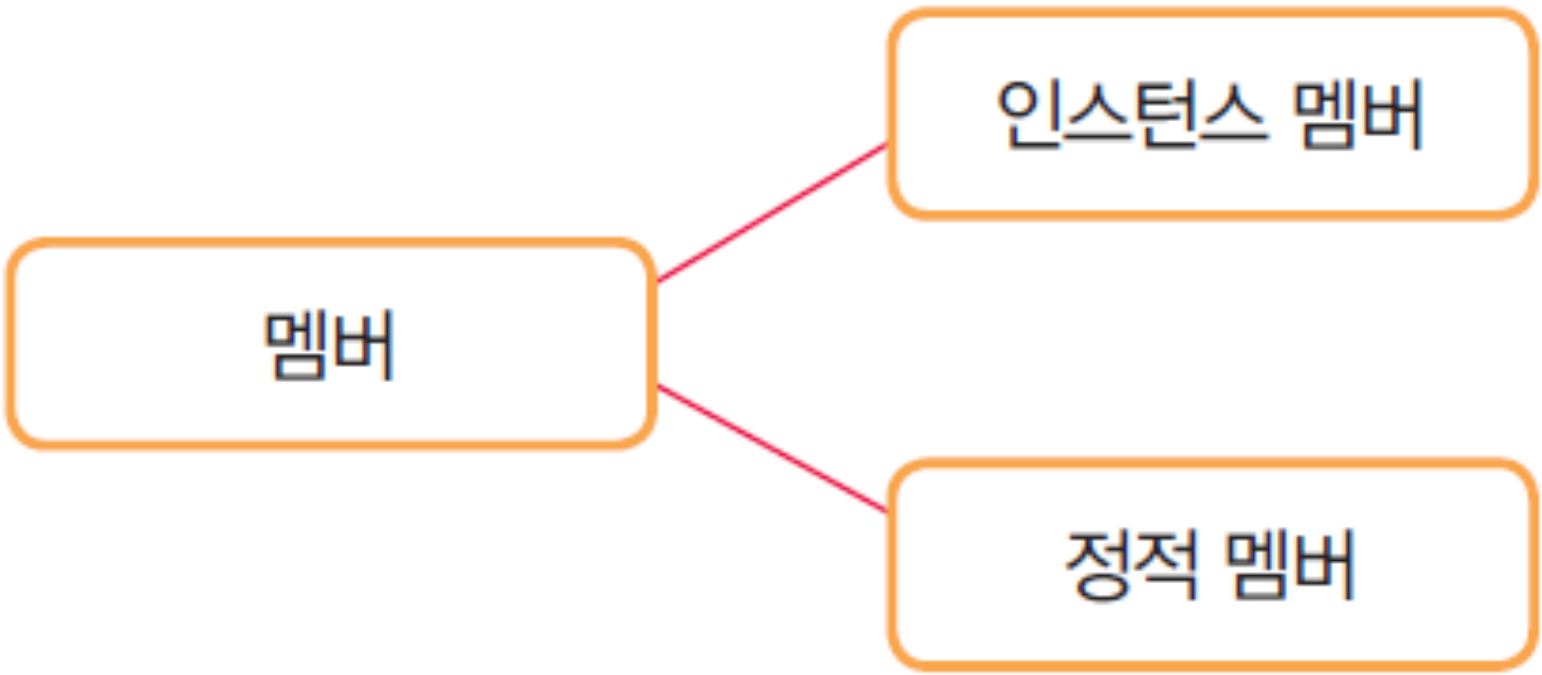
정적 멤버

- 정적 멤버(static member)는 모든 객체를 통틀어서 하나만 생성되고 모든 객체가 이것을 공유하게 된다.



인스턴스 멤버 vs 정적 멤버

- 클래스의 멤버는 인스턴스 멤버와 정적 멤버로 나누어진다.



인스턴스 변수

- 인스턴스마다 별도로 생성되기 때문에 인스턴스 변수(instance variable)라고도 한다.

```
class Television {
    int channel;
    int volume;
    boolean onOff;
}
```

channel	7
volume	9
onOff	trun

객체 A

channel	9
volume	10
onOff	trun

객체 B

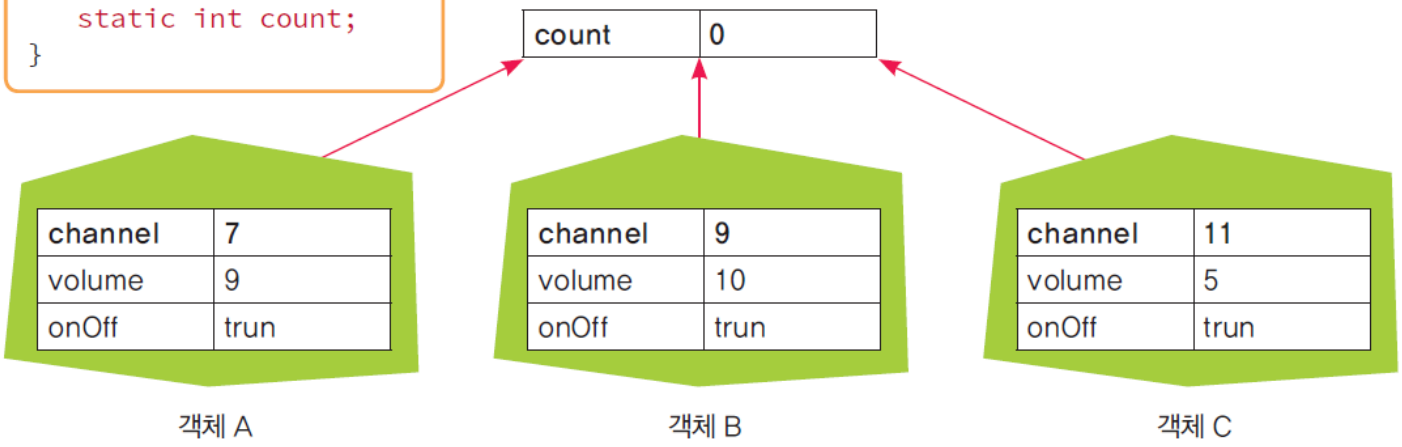
channel	11
volume	5
onOff	trun

객체 C

정적 변수

- 정적 변수는 모든 객체에 공통인 변수이다.
- 정적 변수는 하나의 클래스에 하나만 존재한다

```
class Television {
   int channel;
   int volume;
   boolean onOff;
   static int count;
}
```



정적 변수 예제

```
public class Car {
    private String model;
    private String color;
    private int speed;

     // 자동차의 시리얼 번호
     private int id;
    private static int numbers = 0;

    public Car(String m, String c, int s) {
        model = m;
        color = c;
        speed = s;
         // 자동차의 개수를 증가하고 id에 대입한다.
        id = ++numbers;
    }
}
```

SOLUTION

```
public class CarTest {
    public static void main(String args[]) {
        Car c1 = new Car("S600", "white", 80); // 첫 번째 생성자 호출
        Car c2 = new Car("E500", "blue", 20); // 첫 번째 생성자 호출
        int n = Car.numbers; // 정적 변수
        System.out.println("지금까지 생성된 자동차 수 = " + n);
    }
}
```

지금까지 생성된 자동차 수 = 2

정적 메소드

- 변수와 마찬가지로 메소드도 정적 메소드로 만들 수 있다.
- 정적 메소드는 static 수식자를 메소드 앞에 붙이며 클래스 이름을 통하여 호출되어야 한다.

- (예) double value = Math.sqrt(9.0);

```
public class Car {
    private String model;
    private String color;
    private int speed;

    // 자동차의 시리얼 번호
    private int id;
    // 실체화된 Car 객체의 개수를 위한 정적 변수
    private static int numbers = 0;

    public Car(String m, String c, int s) {
        model = m;
        color = c;
        speed = s;
        // 자동차의 개수를 증가하고 id에 대입한다.
        id = ++numbers;
    }
        // 정적 메소드
    public static int getNumberOfCars() {
        return numbers; // OK!
    }
}
```

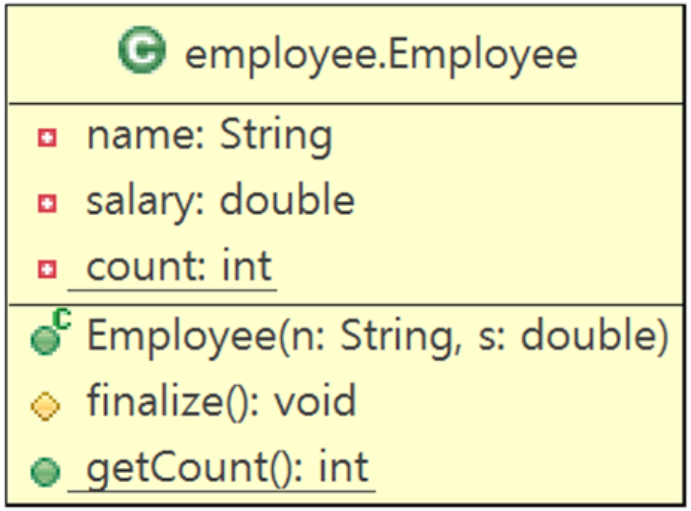
정적 변수 예제

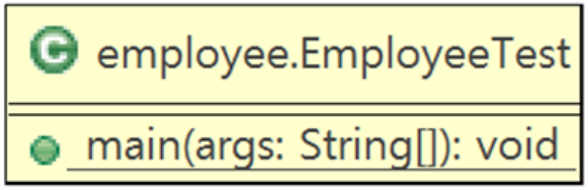
```
public class CarTest {
    public static void main(String args[]) {
        Car c1 = new Car("S600", "white", 80);          // 첫 번째 생성자 호출
        Car c2 = new Car("E500", "blue", 20);           // 첫 번째 생성자 호출
        int n = Car.getNumberOfCars();    // 정적 메소드 호출
        System.out.println("지금까지 생성된 자동차 수 = " + n);
    }
}
```


```
지금까지 생성된 자동차 수 = 2
```

LAB: 같은 크기의 Box인지 확인하기

- 직원을 나타내는 클래스에서 직원들의 수를 카운트하는 예를 살펴보자. 직원의 수를 정적 변수로 나타낸다. 객체가 하나씩 생성될 때마다 생성자에서 정적 변수 count를 증가한다.





SOLUTION

```
public class Employee {
        private String name;
        private double salary;

        private static int count = 0; // 정적 변수
        // 생성자
        public Employee(String n, double s) {
                name = n;
                salary = s;
                count++; // 정적 변수인 count를 증가
        }
        // 객체가 소멸될 때 호출된다.
        protected void finalize() {
                count--; // 직원이 하나 줄어드는 것이므로 count를 하나 감소
        }
        // 정적 메소드
        public static int getCount() {
                return count;
        }
}
```

SOLUTION

```
public class EmployeeTest {
        public static void main(String[] args) {
                Employee e1, e2, e3;
                e1 = new Employee("김철수", 35000);
                e2 = new Employee("최수철", 50000);
                e3 = new Employee("김철호", 20000);

                int n = Employee.getCount();
                System.out.println("현재의 직원수=" + n);
        }
}
```


```
현재의 직원수=3
```

Q & A