Java Wrapper class에 대해

- 기본 타입(primitive type)을 객체로 다루기 위해서 사용하는 클래스
- 생성된 인스턴스의 값만을 참조할 수 있으므로, 인스턴스에 저장된 값을 직접 변경하는 것은 불가능

(1) Wrapper 클래스의 연산

- Auto Boxing/Unboxing에 의해 자동으로 계산이 가능
- Wrapper 클래스 객체 간의 동등비교는 equals() 메소드 사용
- Wrapper 클래스와 기본자료형의 연산은 Auto Boxing/Unboxing에 의해 동등비교(==) 연산자와 equals() 메소드 모두 가능

(2) Wrapper 클래스의 구현된 코드(일부)

```
class Integer {
        private final int value;

        public Integer(int value) {
            this.value = value;
        }

        public static Integer valueOf(int i) {
                //...
        }

        public int intValue() {
                return value;
        }
}
```

(3) swap 메소드 구현방법

- Wrapper 클래스의 Auto Boxing/Unboxing에 의해 새로운 객체가 생성되므로, 다른 언어와는 다르게 구현되어야 함

(예 1)

```
public class Test {
    static class MyClass {
        int value;
        public MyClass(int value) {
            this.value = value;
        }
    }

    public static void swap(MyClass x, MyClass y) {
        int temp;
        temp = x.value;
        x.value = y.value;
        y.value = temp;
    }

    public static void main(String[] args) {
        MyClass a = new MyClass(3);
        MyClass b = new MyClass(5);
        System.out.println("prev: a = " + a.value + ", b = " + b.value);
        swap(a, b);
        System.out.println("next: a = " + a.value + ", b = " + b.value);
    }
}
```

(예 2)

```
public class Test {
    public static void swap(int[] arr) {
        int temp = arr[0];
        arr[0] = arr[1];
        arr[1] = temp;
    }

    public static void main(String[] args) {
        int a = 3, b = 5;
        int[] arr = {a, b};
        System.out.println("prev: a = " + arr[0] + ", b = " + arr[1]);
        swap(arr);
        System.out.println("next: a = " + arr[0] + ", b = " + arr[1]);
    }
}
```