Data Structure Assignment #2.1

20171620 문성찬

---

1. 교과서 915쪽 D.24에 나오는 equals는 무엇이고 왜 필요한가? == 와는 어떻게 다른가? 설명하라.

D.24에 처음에 나오는 equals() 메서드는

```
public boolean equals(Object other){
    return (this == other);
}
```

java.lang에 속한 Object 클래스의 equals(Object object) 메서드로써 other과 this(자기자신)의 레퍼런스를 == 로 단순 비교하여 결과를 리턴하도록 만들어져 있으며, 내용에 대해서는 비교하지 않습니다.

그 이후에 나오는 equals() 메서드는

```
public boolean equals(Object other){
    boolean result = false;

    if(other instanceof Name){
        Name othername = (Name)other;
        result = first.eqauls(otherName.first) && last.eqauls(otherName.last);
    }
    return result;
}
```

other가 Name 클래스의 객체인지 또는 Name 클래스의 하위 클래스의 객체인지 확인하고, 현재 객체와 인자로 넘어온 객체(other)의 first와 last 변수(String) 값을 비교합니다.

equals는 Object 클래스에 정의된 메서드입니다.
(String, Integer, Double 등의 하위 클래스에서 오버라이딩 되어있습니다.)
'==' 는 자바 언어 자체의 연산자입니다.
다시 말해서, equals 메서드는 오버라이딩 가능하지만, '==' 는 오버라이딩이 불가능합니다
또한 위 예시에서 언급한대로 equals는 기본적으로 객체 내부 값을 비교하며, '==' 는 객체 인스턴스의 주소값(참조값)을 비교합니다.

따라서 equals는 객체들의 주소값(참조값)에 상관없이 단순히 객체 내부의 값들만을 비교하고 싶을 때, 즉 동등성 비교를 하고 싶을 때 필요합니다.

---

2. 교과서 265 쪽 Java Interlude 3 에 나오는 Comparable Interface의 compareTo()와 비교해보라. 각각 어떻게 사용 하는 것인지 설명하라.

1) equals()와 compareTo() 비교

campareTo() 는 Comparable Interface에 속해 있고, equals() 는 Object 클래스 내에 속해 있습니다.
compareTo() 는 반환형태가 integer type 이고 equals() 는 반환 형태가 boolean type이며,
두 메서드는 기본 유형(Primitive Types)에 대해서 적용할 수 없습니다.

compareTo() 는 현재 객체와 인자로 넘어온 객체 내의 값에 대해 비교하며, 값이 같다면 0을,
인자로 넘어온 객체의 값이 현재 객체의 값보다 크다면 음수를, 그렇지 않다면 양수를 리턴합니다.

equals() 는 현재 객체와 인자로 넘어온 객체 내의 값에 대해 비교하는 것은 똑같지만, 같은 값이면
True를 그렇지 않다면 False를 리턴합니다.

2) 각각의 사용 방법 설명 (예시)

물건의 numberOfObject 값을 비교하는 Object_equals와 Object_compareTo 클래스를 작성

- equals()

```
public class Object_equals
{
    private int numberOfObject;

    public Object_equals() {
        this(0);
    }

    public Object_equals(int number) {
        numberOfObject = number;
    }

    public boolean equals(Object other) {
        boolean result = false;

        if(other instanceof Object_equals) {
            Object_equals obj = (Object_equals)other;
            result = numberOfObject == obj.numberOfObject;
        }
        return result;
    }
}
```

java의 최상위 클래스인 Object에 정의된 equals를 Override하여 객체 내의 numberOfObject를
비교하도록 작성

- compareTo()

```
public interface Comparable<T>
{
    public int compareTo(T other);
}
```

```
public class Object_compareTo implements Comparable<Object_compareTo>
{
    private int numberOfObject;

    public Object_compareTo() {
        this(0);
    }
```

```
    public Object_compareTo(int number) {
        numberOfObject = number;
    }

    public int compareTo(Object_compareTo other) {
        return numberOfObject - other.numberOfObject;
    }
}
```

Comparable 인터페이스를 구현하여 compareTo가 객체 내의 numberOfObject를 비교하도록 작성

3) 추가 비교

equals() 와 같이 Override 해야하는건 코드의 유연성에 영향을 끼쳐 안좋은 영향을 주게 됩니다.
따라서 Comparable 인터페이스를 구현하는 것이 s/w quality를 올리는 데에 더 좋습니다.