

능동적 사고 방식의

java

강사 박주병

Park Ju Byeong

Park Ju Byeong



Part13 기본클래스 및 제네릭

01 기본클래스

02 Util 패키지

03 제네릭

04 실습 문제



01

기본 클래스

1-1 실습문제 (normal)

Point 클래스를 만드시오

- 멤버변수 int x, int y, int z 를 가진다.
- 생성자, toString() 과 equals() 메서드를 오버라이딩 하자.

```
Point pt1 = new Point(10,20,30);  
Point pt2 = new Point(10,20,30);  
System.out.println(pt1.equals(pt2));  
|  
System.out.println(pt1.toString());
```

```
<terminated> main [Java Application] C  
true  
x: 10 y: 20 z: 30
```

1-1 문제풀이 (normal)

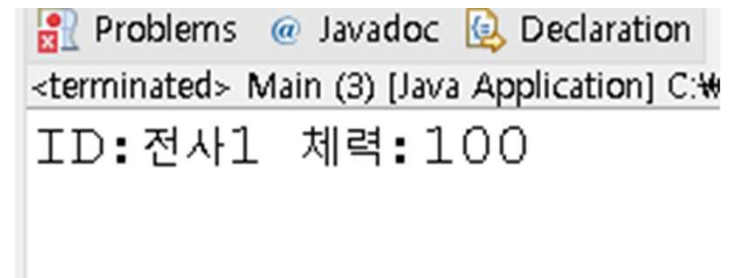
```
public class Point {  
  
    int x;  
    int y;  
    int z;  
  
    Point(int x, int y ,int z)  
    {  
        this.x= x;  
        this.y = y;  
        this.z = z;  
    }  
  
    @Override  
    public boolean equals(Object obj) {  
  
        if(obj instanceof Point)  
        {  
            Point p =(Point)obj;  
            //return x==p.x && y==p.y && z==p.z;  
            return this.toString().equals(p.toString());  
        }  
        return false;  
    }  
  
    @Override  
    public String toString() {  
        return "x: " + x + " y: "+ y+" z: "+ z;  
    }  
}
```

1-2 실습문제 (hard)

Warrior 클래스를 만드시오

- 멤버변수 String id, int hp
- 생성자, clone() 메서드를 오버라이딩 하시오
- toString()을 오버라이딩 하여 Warrior의 기본정보를 출력하시오

```
Warrior w1 = new Warrior("전사1", 100);  
  
Warrior w2 = w1.clone();  
  
System.out.println(w2);
```



객체 자체를 넘겨줘도 println() 메서드 내부에서 매개변수 객체의 toString()을 호출하여 출력해준다.

1-2 문제풀이 (hard)

```
public class Warrior implements Cloneable {  
    String id;  
    int hp;
```

Cloneable 인터페이스를
구현하지 않으면 clone()
메서드를 호출시 예외가
발생한다.

```
    public Warrior(String id, int hp)  
    {  
        this.id = id;  
        this.hp = hp;  
    }
```

```
    @Override  
    public String toString() {  
        // TODO Auto-generated method stub  
        return "ID:" + id + " 체력:" + hp;  
    }
```

```
    @Override  
    public Warrior clone() {  
        Warrior result = null;  
        try  
        {
```

공변반환타입으로 오버라이딩 가능

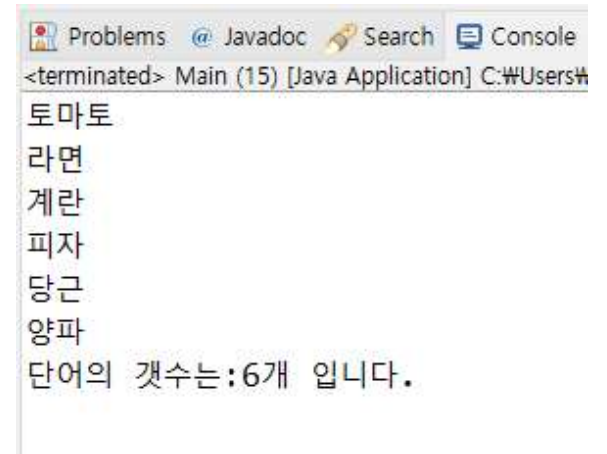
```
            result = (Warrior) super.clone();  
        } catch (CloneNotSupportedException ex)  
        {  
            return result;  
        }  
  
        return result;  
    }
```

부모의 기능을 그대로
사용한다.

2-1 실습문제 (normal)

다음과 같은 한 개의 문자열을 "," 를 기준으로 잘라내어 단어를 각각 출력 하고 단어개 몇 개인지 출력하시오

```
public static void main(String[] args) {  
    // TODO Auto-generated method stub  
  
    String foods = "토마토,라면,계란,피자,당근,양파";
```



Problems @ Javadoc Search Console
<terminated> Main (15) [Java Application] C:\Users#\n
토마토
라면
계란
피자
당근
양파
단어의 갯수는:6개 입니다.

}

'19

Park Ju Byeong

2-1 문제풀이 (normal)

```
public static void main(String[] args) {  
    // TODO Auto-generated method stub  
  
    String foods = "토마토,라면,계란,피자,당근,양파";  
  
    String[] result = foods.split(",");  
  
    for(String food : result)  
    {  
        System.out.println(food);  
    }  
    System.out.println("단어의 갯수는:"+result.length +"개 입니다.");  
}
```

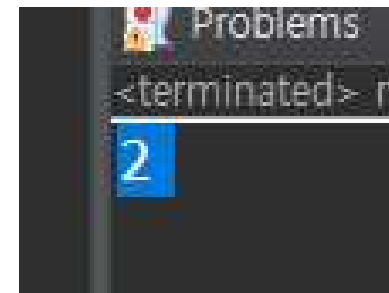
2-2 실습문제 (hard)

문자열에서 원하는 문자열이 몇 개 포함되어 있는지 찾는
`count()` 메서드를 만드시오

`count()` 메서드는 `main` 클래스에 만들어도 되고 별도의 클래스를 만들어도 된다.

`indexOf()` 메서드를 활용하시오

```
String str = "abcdefgab";  
System.out.println(count(str, "ab"));
```



2-2 문제풀이 (hard)

```
public static int count(String str, String target)
{
    int count = 0; // 찾은 횟수
    int pos = 0; // 찾기 시작할 위치

    while(true)
    {
        pos = str.indexOf(target, pos);

        // 값을 찾으면
        if(pos != -1)
        {
            count++;
            pos += target.length(); // 찾은 문자열의 첫번
        }
        else // 일치하는 문자가 없으면 종료한다.
        {
            break;
        }
    }
    return count;
}
```

2-3 실습문제 (hard)

앞서 만든 Warrior 클래스에 캐릭터의 위치정보도 추가하자.

1. 멤버변수 : Point position

2. 기존의 toString() 메서드에 위치정보도 나올수 있게 수정하자.

Point 클래스는 1-1에서 만든 3차원 좌표를 저장 할 수 있는타입이다.

멤버변수로 객체를 가지고 있다면 clone()을 사용하여 복사시 멤버변수의 객체는 주소값만을 복사하게 된다.

따라서 지금 상태로는 원본 객체의 위치정보가 변경되면 복사된 객체의 위치정보도 같이 바뀌게 된다. 이 문제를 해결하기 위해 deepCopy() 메서드를 만들어 위치정보 역시 새로운 객체를 생성하여 복사 하도록 해보자.

```
Warrior w1 = new Warrior("전사1", 100, new Point(10, 20, 30));
```

```
Warrior w2 = w1.deepCopy();
```

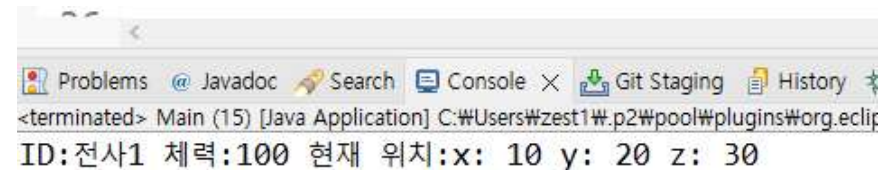
```
//w2의 위치정보를 변경하여도 w1은 영향을 받지 않는다.
```

```
w2.position.x = 100;
```

```
w2.position.y = 100;
```

```
w2.position.z = 100;
```

```
System.out.println(w1.toString());
```



2-3 문제풀이 (hard)

```
public Warrior deepCopy()
{
    Warrior result = this.clone();

    result.position = new Point(position.x, position.y, position.z);

    return result;
}

@Override
public String toString() {
    // TODO Auto-generated method stub
    return "ID:"+id+" 체력:"+hp + " 현재 위치:"+position.toString();
}
```

— 04

실습문제

3-1 실습문제 (normal)

제네릭을 활용하여 어떠한 객체든지 담아 둘 수 있는 Box 클래스를 만들어보자.

- 물건은 1개만 담아둔다.
getter, setter 를 만들자.

```
Box<String> box = new Box<>();  
Box<Integer> box2 = new Box<>();  
box.setItem("가나다라");  
  
System.out.println(box.getItem());
```

```
<terminated> mai  
가나다라
```

3-1 문제풀이 (normal)

```
public class Box<T> {  
    private T item;  
    public void setItem(T item)  
    {  
        this.item = item;  
    }  
    public T getItem()  
    {  
        return this.item;  
    }  
}
```


3-2 실습문제 (normal)

3-1 에서 만든 Box 클래스에 Book 객체만 담을수 있도록 변경해보자.

- Book 클래스가 없다면 생성을 하자(다른 패키지에 있는걸 import 해서 사용해도 됨)

```
public class Main {  
    public static void main(String[] args) {  
        // TODO Auto-generated method stub  
  
        Box<OilCar> box = new Box<OilCar>();  
        Box<String> box2 = new Box<String>();  
        Box<Book> box3 = new Box<Book>();  
    }  
}
```

3-2 문제풀이 (normal)

```
1
2
3 public class Box<T extends Book> {
4
5     private T item;
6
7     public void setItem(T item)
8     {
9         this.item = item;
10    }
11
12    public T getItem()
13    {
14        return this.item;
15    }
16 }
```

↓
해당 클래스 및 자식 클래스만 올수 있다.

3-3 실습문제 (hard)

제네릭을 이용해 나만의 ArrayList를 만들어보자.

멤버변수에 배열을 만들어 요소들을 관리하자. 배열의 타입은 T[] 가 불가능하다(제네릭타입으로 객체 생성이 불가). 따라서 Object[] 타입으로 배열을 만들고 배열 객체를 생성해야 한다.

- 클래스명 : MyArrayList
- 멤버변수 : private Object[] list
- 멤버메서드 : void add(T item)
- T get(int index)
- 배열의 사이즈는 10으로 설정한다.
- get 메서드에서 배열의 특정 요소를 반환 할때 (T) 로 형변환을 해야 한다.

```
public static void main(String[] args) {  
    // TODO Auto-generated method stub  
  
    MyArrayList<String> list = new MyArrayList<String>();  
  
    for(int i =0 ; i< 15;i++)  
        list.add("데이터"+i);  
  
    System.out.println(list.get(5));  
}
```



Park Ju Byeong

3-3 문제풀이 (hard)

```
import java.util.ArrayList;
```

```
public class MyArrayList<T>{
```

```
    private Object[] list = new Object[10];
```

→ T[]는 객체 생성이 불가능하다.
배열을 쓰기 위해선 배열 객체를 생성해야 하므로 Object[]를 사용한다.

```
    void add(T item)
```

```
    {
        for(int i = 0; i < list.length; i++)
        {
            if(list[i] == null)
            {
                list[i] = item;
                return;
            }
        }
    }
```

```
    T get(int index)
```

```
    {
        if(list.length <= index) // 배열의 사이즈를 넘어서면
            throw new ArrayIndexOutOfBoundsException("현재 요소의 범위를 넘었습니다.");
        else if(index < 0) // 음수라면
            throw new IllegalArgumentException("음수 index는 불가능합니다");
    }
```

```
        return (T) list[index];
```

→ 반환시 Object 타입을 제네릭 타입으로 변환한다.

3-4 실습문제 (expert)

3-3에서 만든 MyArrayList를 개선해보자 .

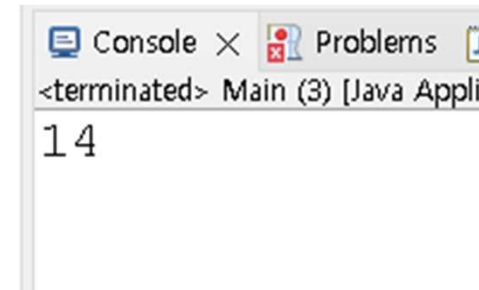
- int size() 메서드를 추가하자.
배열에 실제 들어가 있는 객체의 수를 반환한다.(배열의 size 가 아니다)
- void remove(int index) 메서드를 추가 하자.
매개변수로 넘어온 인덱스의 요소를 삭제하고 그뒤의 요소들은 앞으로 한칸씩 당긴다.
- add 메서드를 개선해보자.
현재 add 메서드는 내부의 배열이 가득 차게 되면 더 이상 요소를 넣을 수 없도록 되어 있다. 만약 add시 현재 배열이 가득찬 상태라면 배열의 사이즈를 10만큼 늘려보자.
+10 사이즈의 새로운 배열을 생성 후 기존 배열의 객체들을 모두 옮겨 줘야 한다.
remove를 하더라도 이미 늘어난 배열의 사이즈는 다시 안줄여도 된다.

```
public static void main(String[] args) {  
    // TODO Auto-generated method stub
```

```
    MyArrayList<String> list = new MyArrayList<String>();
```

```
    for(int i =0 ; i< 15;i++)  
        list.add("데이터"+i);
```

```
    list.remove(9);  
    System.out.println(list.size());
```



Park Ju Byeong

3-4 문제풀이 (expert)

```
int size()
{
    int cnt=0;

    //삭제시 요소를 당겨와서 중간에 null로 비어져 있는 케이스는 없다고 가정하고 만든다.
    for(Object item : list)
    {
        if(item == null)
            break;

        cnt++;
    }

    return cnt;
}

void remove(int index)
{
    if(list.length<= index)//배열의 사이즈를 넘어서면
        throw new ArrayIndexOutOfBoundsException("현재 요소의 범위를 넘었습니다.");
    else if(index <0) // 음수라면
        throw new IllegalArgumentException("음수 index는 불가능합니다");

    //그냥 뒤에 요소를 해당 칸으로 옮기면 되므로 null로 초기화 하는 작업은 불필요하다고 느낄수도 있다.
    //그러나 마지막 요소를 지우는 상황이라면 뒤의 요소가 없기때문에 해당 index 번호를 초기화 해줄 필요가 있다.
    list[index] = null;

    for(int i = index;i<list.length-1;i++)
    {
        list[i] = list[i+1];
    }

    //한칸씩 당겼으므로 마지막 요소는 비워둔다.
    list[list.length-1] = null;
}
```

```
void add(T item)
{
    //size 메서드가 있으니 더이상 반복문을 사용할 필요 없다.
    int index = this.size();

    //배열이 가득차 있으면
    if(index >= list.length)
    {
        Object[] temp = new Object[list.length+10];

        //요소들을 옮긴다.
        for(int i =0; i <this.size();i++)
            temp[i] = list[i];

        //배열의 주소값을 새로운 배열의 주소로 교체 한다;
        list = temp;
    }

    //새로운아이템을 추가한다.
    list[index] = item;
}
```



THANK YOU



강사 박주병