

능동적 사고 방식의

java

강사 박주병

Park Ju Byeong

Park Ju Byeong



Part07 메모리



01 지역변수, 멤버변수

02 클래스 변수(static 변수)

03 메모리 영역

04 실습 문제



01

지역변수, 멤버변수

실습문제1

1-1 Person 클래스를 만들어 사용해보자(normal)

- 멤버변수 : String name (이름)
String RRN (주민번호)

```
public class Main {  
  
    public static void main(String[] args) {  
        // TODO Auto-generated method stub  
  
        Person p1 = new Person();  
        Person p2 = new Person();  
        Person p3 = new Person();  
  
        p1.name = "홍길동";  
        p1.RRN = "750215-1234567";  
  
        p2.name = "김길동";  
        p2.RRN = "850215-1234567";  
  
        p3.name = "박길동";  
        p3.RRN = "040215-2234567";  
  
        System.out.println("이름:" + p1.name + "\t주민번호: " + p1.RRN);  
        System.out.println("이름:" + p2.name + "\t주민번호: " + p2.RRN);  
        System.out.println("이름:" + p3.name + "\t주민번호: " + p3.RRN);  
    }  
}
```

```
<terminated> Main (8) [Java Application] C:\Users\Wzest1\p2\pool\plug  
이름:홍길동      주민번호: 750215-1234567  
이름:김길동      주민번호: 850215-1234567  
이름:김길동      주민번호: 850215-1234567
```

정답

```
public class Person  
{
```

```
    String name;
```

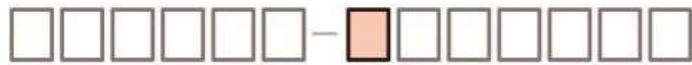
```
    String RRN;
```

외부에서 쓰기 위해선
멤버변수(인스턴스변수)로 선언해야
된다.

1-2 앞서 만든 Person 클래스에 기능을 추가해보자(hard)

- 멤버 메서드 : void showState() 이름과 주민번호를 출력한다
String getGender() 주민번호를 이용해 성별을 반환한다.
- 앞서만든 showState 메서드 내부에서 getGender() 메서드를 이용해 성별도 같이 나오도록 하자

주민등록번호 성별 표시



1800년대 태어난 남자 - 9	1900년대 태어난 외국인 남자 - 5
여자 - 0	여자 - 6
1900년대 태어난 남자 - 1	2000년대 태어난 외국인 남자 - 7
여자 - 2	여자 - 8
2000년대 태어난 남자 - 3	
여자 - 4	

```
public class Main {  
  
    public static void main(String[] args) {  
        // TODO Auto-generated method stub  
  
        Person p1 = new Person();  
        Person p2 = new Person();  
  
        p1.RRN="950101-1234567";  
        p2.RRN="950101-6789123";  
  
        String p1Gender = p1.getGender();  
        String p2Gender = p2.getGender();  
  
        System.out.println(p1Gender);  
        System.out.println(p2Gender);  
  
        p1.showState();  
        p2.showState();  
    }  
}
```

정답

```
String getGender()
{
    String gender = RRN.substring(7,8);
    switch(gender)
    {
        case "1","3","5","7","9":
            gender = "남";
            break;
        default:
            gender = "여";
            break;
    }
    return gender;
}
```

주민번호에서 성별 부분만 잘라낸다.

남자인 경우

그외는 모두 여자

정해진 성별을 메서드 밖으로 반환한다.

```
void showState()
{
    System.out.println("이름:" + name + "\t주민번호:" + RRN + "\t성별: " + getGender())
}
```

성별을 출력할때 만들어 놓은 getGender() 메서드를 활용하면 된다.

정답

```
switch(gender)
{
    case "1", "3", "5", "7", "9":
        gender = "남";
        break;

    default:
        gender = "여";
        break;
}
```

```
if(gender.equals("1")
    || gender.equals("3")
    || gender.equals("5")
    || gender.equals("7")
    || gender.equals("9"))
    gender = "남";
else
    gender = "여";
```

이런 경우 switch문이 좀더 간결하다.

1-3 앞서 만든 Person 클래스에 기능을 추가해보자(expert)

- 멤버 메서드 : int getAge() 주민번호를 이용해 나이를 반환한다.(만나이)
- 기존의 showState 메서드에 getAge를 이용하여 나이도 출력 하도록 하자.
- 성별 코드로 몇세기인지를 파악해야 정확한 나이 계산이 가능하다.

아래의 코드는 현재의 년,월,일을 int 형태로 얻는 코드이다.
나이계산에 활용하도록 하자.

```
//현재 년월일을 가져오기 위한 객체 생성
Calendar cal = Calendar.getInstance();

//현재의 년월일을 변수에 저장한다.
int nowYear = cal.get(Calendar.YEAR);
int nowMonth = cal.get(Calendar.MONTH)+1;
int nowDay = cal.get(Calendar.DAY_OF_MONTH);
```

```
Person p1 = new Person();
Person p2 = new Person();

p1.RRN="010512-3234567";
p2.RRN="950101-2789123";

p1.showState();
p2.showState();
```

```
<terminated> main (8) [Java Application] C:\Users\zest1\p2\p001\plugins\org.eclipse.justj.openjdk.m
이름:null 주민번호:010512-3234567 성별: 남 나이:21
이름:null 주민번호:950101-2789123 성별: 여 나이:28
```

정답

```
int getAge()
{
    //현재 년월일을 가져오기 위한 객체 생성
    Calendar cal = Calendar.getInstance();

    //현재의 년월일을 변수에 저장한다.
    int nowYear = cal.get(Calendar.YEAR);
    int nowMonth = cal.get(Calendar.MONTH)+1;
    int nowDay = cal.get(Calendar.DAY_OF_MONTH);

    //년도 뒷자리 2개를 가져온다.
    int year = Integer.parseInt(RRN.substring(0,2));

    //성별을 가져온다. 성별의 숫자에 따라 태어난 년도가 결정되기에 필요하다.
    String generation = RRN.substring(7,8);
    //성별의 숫자를 통해 어느세대인지 구분하여 태어난 년도를 만든다.
    switch(generation)
    {
        case "0","9":
            year +=1800;
            break;
        case "1","2","5","6":
            year +=1900;
            break;
        case "3","4","7","8":
            year +=2000;
            break;
    }
    //주민번호에서 년월일을 가져온다.
    int bornMonth = Integer.parseInt(RRN.substring(2,4));
    int bornDay = Integer.parseInt(RRN.substring(4,6));
    //만나이를 계산한다. 생일이 지났는지 여부에 따라 1살을 더할지 말지 결정된다.
    int bornYear = nowYear - year-(nowMonth>= bornMonth && nowDay>= bornDay ? 0 : 1);
    return bornYear;
}
```



03

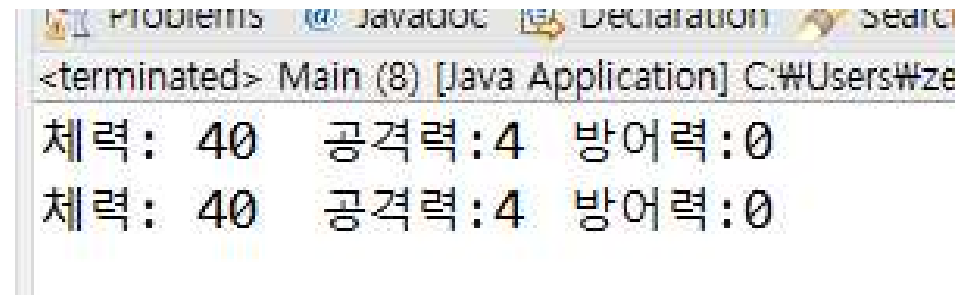
메모리 영역

실습문제2

2-1 Marine 클래스를 만들고 객체를 생성하여 사용해보자.(normal)

- 멤버변수 : int hp , int power(공격력) ,int armor(방어력)
- 멤버메서드 : showState() 객체의 상태를 표시 한다.

```
public class Main {  
  
    public static void main(String[] args) {  
        // TODO Auto-generated method stub  
        Marine m1 = new Marine();  
        Marine m2 = new Marine();  
  
        m1.showState();  
        m2.showState();  
    }  
}
```



Problems JavaDoc Declaration Search
<terminated> Main (8) [Java Application] C:\Users\#ze
체력: 40 공격력:4 방어력:0
체력: 40 공격력:4 방어력:0

정답

```
2  
3 class Marine {  
4     int hp=40;  
5     int power=4;  
6     int armor=0;  
7     . . . . .  
8 }
```

→ 멤버변수 선언과 동시에
초기화 하였기에 객체
생성시 기본값이 된다.

```
void showState()  
{  
    System.out.println("체력: "+hp+"\t 공격력:"+power + "\t 방어력:"+armor);  
}
```

2-2 Marine 클래스의 powerUp, armorUp 메서드를 만들자.(hard)

- 모든 객체가 같은 공격력과 방어력을 가지며 업그레이드시 모든 객체가 다같이 올라가야 한다.)
- power , armor 변수가 수정 되어야 한다.

```
Marine marine1 = new Marine();  
Marine marine2 = new Marine();  
  
marine1.powerUp();  
marine1.armorUp();  
  
marine1.showState();  
marine2.showState();
```

클래스명	Marine	
메서드	powerUp()	매개변수:없음 내용: 모든 유닛의 power를 1증가 시킨다. 리턴:없음
	armorUp()	매개변수:없음 내용: 모든 유닛의 armor를 1증가 시킨다. 리턴:없음

Problems Javadoc Declaration

<terminated> Main [Java Application] C:\Users\WU

```
체력: 40    공격력: 5    방어력: 1  
체력: 40    공격력: 5    방어력: 1
```

정답

```
class Marine {  
    int hp=40;  
    static int power=4;  
    static int armor=0;
```

→ 클래스변수(static변수)로 만들어야
모든 객체들이 공유해서 쓰는값이
된다.

```
void powerUp()  
{  
    power++;  
}
```

→ 인스턴스메서드는 제한없이
인스턴스변수,클래스변수 둘다
사용가능하다.
물론 powerUp과 armorUp을
클래스메서드로 만들어도 된다.

```
void armorUp()  
{  
    armor ++;  
}
```


2-3 Marine 클래스에 attack 메서드를 구현하자(hard)

- 멤버메서드 : void attack(Marine target)
- 매개변수로 받은 Marine 객체의 체력을 감소시킨다.
- 공격시 (공격력 - 상대방의 방어력) 만큼 상대 체력을 감소 시킨다.

```
Marine marine1 = new Marine();  
Marine marine2 = new Marine();  
  
marine1.powerUp();  
marine1.armorUp();  
  
marine1.attack(marine2);  
  
marine1.showState();  
marine2.showState();
```

```
<terminated> Main [Java Application] C:\Users\USER5  
체력: 40    공격력: 5    방어력: 1  
체력: 36    공격력: 5    방어력: 1
```


정답

```
void attack(Marine target)  
{
```

매개변수로 공격대상이 되는 객체를 받는다.

```
    target.hp -= (power - target.armor);
```

공격대상의 hp를 감소시킨다.

2-4 Marine 클래스에 기능을 추가하자(expert)

- 멤버변수: Point position(유닛의 위치정보를 저장)
int shootingRange(공격 사정거리)
- 멤버메서드 : int getDistance(Marine target) 매개변수로 받은 유닛과의 거리를 반환한다.
- attack 메서드에서 getDistance() 와 shootingRange 변수를 이용하여 공격 가능거리가 아니면 공격을 못한다고 출력하자.

Point 클래스를 사용하기 위해서 java.awt.Point 를 import 해야 한다.

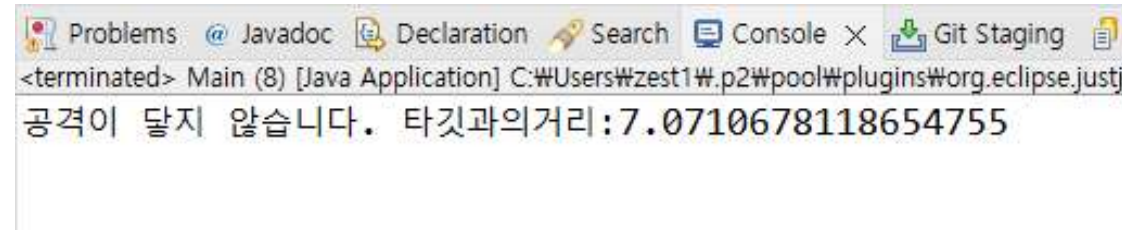
제공근 : Math.sqrt 제공 : Math.pow(2,3) -> 2의3제곱 -> 8

```
Marine m1 = new Marine();
Marine m2 = new Marine();

m1.position = new Point(2,2);
m2.position = new Point(7,7);

m1.attack(m2);
```

$$\text{거리} = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$



정답

```
double getDistance(Marine target)
{
    //타겟과의 거리계산
    double result = Math.sqrt(
        Math.pow(target.position.x - position.x, 2)
        + Math.pow(target.position.y - position.y, 2)
    );

    return result;
}

void attack(Marine target)
{
    double distance = getDistance(target);

    if(distance > shootingRange)
    {
        System.out.println("공격이 달지 않습니다. 타겟과의거리:"+distance);
        return;
    }

    target.hp -= (power - target.armor);
}
```



THANK YOU



강사 박주병