

능동적 사고 방식의

java

강사 박주병

Park Ju Byeong

Park Ju Byeong



Part06 객체지향 —

01 객체지향의 이해

02 메서드

03 메서드 활용

04 실습 문제



01

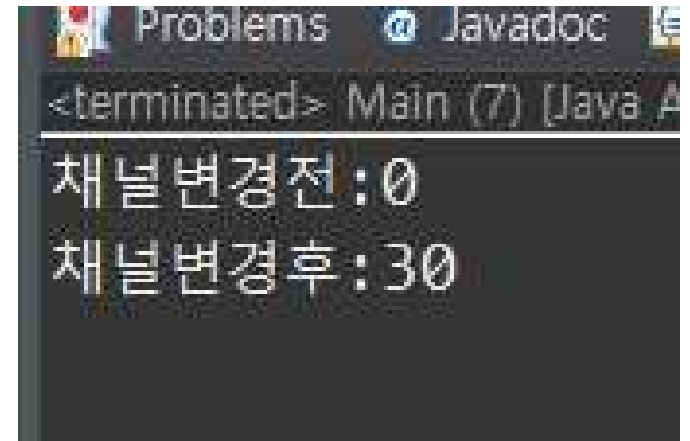
객체지향의 이해

1-1 실습문제 (normal)

아래의 정보를 참고하여 TV 클래스를 만들고 객체를 생성하여 채널 값을 변경해보자.

- 클래스명 : TV
- 멤버변수 : channel

```
public class Main {  
  
    public static void main(String[] args) {  
        // TODO Auto-generated method stub  
  
        TV t1 = new TV();  
  
        System.out.println("채널변경전:" + t1.channel);  
  
        t1.setChannel(30);  
  
        System.out.println("채널변경후:" + t1.channel);  
    }  
}
```



Problems @ Javadoc
<terminated> Main (7) [Java A
채널변경전:0
채널변경후:30

1-1 문제풀이 (normal)

```
public class Main {  
    public static void main(String[] args) {  
        // TODO Auto-generated method stub  
  
        Tv t1 = new Tv();  
  
        System.out.println("채널변경전:"+t1.channel);  
  
        t1.channel = 30;  
        System.out.println("채널변경후:"+t1.channel);  
    }  
}
```

— 02

메서드

2-1 실습문제 (normal)

MyMath 클래스를 구현 하여 아래의 식의 답을 출력하자

- 사칙연산 연산자 대신 MyMath 클래스의 메서드를 이용하여 계산하자

```
MyMath math = new MyMath();  
  
System.out.println(result);  
System.out.println(500/5+3*27-5);
```

```
<terminate  
176  
176
```

클래스명	MyMath	
멤버변수	없음	
메서드	add()	매개변수:int 2개 리턴: 더하기 결과
	subtract()	매개변수:int 2개 리턴: 빼기 결과
	multiply()	매개변수:int 2개 리턴: 곱하기 결과
	divide()	매개변수:int 2개 리턴: 나누기 결과

2-1 문제풀이 (normal)

```
MyMath math = new MyMath();

int result = math.subtract(math.add(
    math.divide(500, 5)
    , math.multiply(3, 27)), 5);

System.out.println(result);
System.out.println(500/5+3*27-5);
```


2-2 실습문제 (Hard)

Student 클래스를 구현 후 아래와 같이 출력하자(객체배열활용)

- 객체배열 활용하고 성적은 Math.random() 으로 1~100점 사이

```
Student[] students = new Student[5];
```

```
System.out.println(student.kor
    + "\t" + student.eng
    + "\t" + student.math
    + "\t" + student.getTotal()
    + "\t" + student.getAverage());
```

클래스명	Student	
멤버변수	String name	학생이름
	int kor	국어
	int eng	영어
	int math	수학
메서드	getTotal()	매개변수: 없음 리턴: 합계
	getAverage()	매개변수: 없음 리턴: 평균

```
! <terminated> Main (7) [Java Application] C:\Users\zest1\p2\pool\pl
```

국어	영어	수학	합계	평균
60	38	73	171	57
82	9	37	128	42
95	26	15	136	45
21	17	71	109	36
19	16	71	106	35

2-2 문제풀이 (Hard)

```
Student[] students = new Student[5];

students[0] = new Student();
students[1] = new Student();
students[2] = new Student();
students[3] = new Student();
students[4] = new Student();

System.out.println("국어\t영어\t수학\t합계\t평균");
for(Student student : students)
{
    student.kor = (int)(Math.random()*100)+1;
    student.eng = (int)(Math.random()*100)+1;
    student.math = (int)(Math.random()*100)+1;

    System.out.println(student.kor
        + "\t" + student.eng
        + "\t" + student.math
        + "\t" + student.getTotal()
        + "\t" + student.getAverage());
}
```



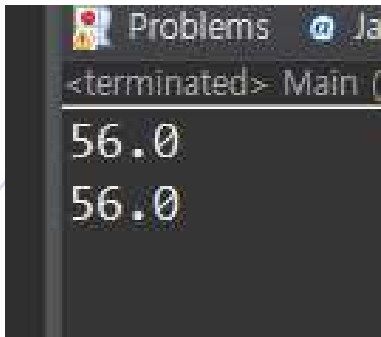
03

메서드 활용

3-1 실습문제 (normal)

실습문제1에서 구현한 MyMath 클래스의 add메서드를 float 과 double도 가능하도록 오버로딩 해보자

```
MyMath math = new MyMath();  
  
float result = math.add(30.5f, 25.5f);  
  
System.out.println(result);  
  
double result2 = math.add(30.5, 25.5);  
  
System.out.println(result2);
```



3-1 문제풀이 (normal)

```
int add(int x, int y)
{
    return x+y;
}

float add(float x, float y)
{
    return x+y;
}

double add(double x, double y)
{
    return x+y;
}
```

3-2 실습문제 (normal)

MyMath 클래스에 평균, 최대, 최소값을 구하는 메서드를 추가 하자
(매개 변수의 개수는 제한없이 늘어나야 한다.)

```
MyMath math = new MyMath();  
  
System.out.println(math.avg(32,10,320,40,10,-5,3));  
  
System.out.println(math.max(32,10,320,40,10,-5,3));  
  
System.out.println(math.min(32,10,320,40,10,-5,3));
```

클래스명	MyMath	
메서드	avg()	매개변수: int 가변인자 리턴: int형 평균
	max()	매개변수:int 가변인자 리턴: int형 최대값
	min()	매개변수:int 가변인자 리턴: int형 최소값

```
<terminated>  
58  
320  
-5
```


3-2 실습문제 (normal)

```
int avg(int... n)
{
    int sum=0;
    for(int i : n)
        sum+= i;

    return sum/n.length;
}

int max(int... n)
{
    int max = n[0];

    for(int i : n)
        if(max<i)
            max = i;

    return max;
}
```

```
int min(int... n)
{
    int min = n[0];

    for(int i : n)
        if(min>i)
            min = i;

    return min;
}
```

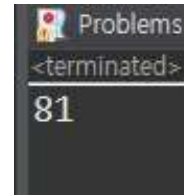
3-3 실습문제 (hard)

MyMath 클래스에 거듭제곱을 계산하는 메서드를 만들자 (재귀호출 사용할것)

power(3,4) -> 3x3x3x3 =81

```
MyMath math = new MyMath();  
  
long result = math.power(3, 4);  
  
System.out.println(result);
```

클래스명	MyMath	
메서드	power()	매개변수: int x, int n 리턴: x를 n제곱한 결과



Problems
<terminated>
81

3-3 문제풀이 (hard)

```
MyMath math = new MyMath();  
long result = math.power(3, 4);  
System.out.println(result);
```

```
int power(int x, int n)  
{  
    if(n == 1)  
        return x;  
  
    return x * power(x, n-1);  
}
```

```
int power(int x, int n)  
{  
    if(n == 1)  
        return x;  
  
    return x * power(x, n-1);  
}
```

```
int power(int x, int n)  
{  
    if(n == 1)  
        return x;  
  
    return x * power(x, n-1);  
}
```

```
int power(int x, int n)  
{  
    if(n == 1)  
        return x;  
  
    return x * power(x, n-1);  
}
```

3 5

```
int power(int x,int n)
{
    if(n==1)
        return x;

    //짝수
    if(n%2 ==0)
        return power(x*x,n/2);
    else
        return x*power(x*x,(n-1)/2);
}
```

3 3*3 2

3 2

```
int power(int x,int n)
{
    if(n==1)
        return x;

    //짝수
    if(n%2 ==0)
        return power(x*x,n/2);
    else
        return x*power(x*x,(n-1)/2);
}
```

3*3 1

3 1

```
int power(int x,int n)
{
    if(n==1)
        return x;

    //짝수
    if(n%2 ==0)
        return power(x*x,n/2);
    else
        return x*power(x*x,(n-1)/2);
}
```

3

//짝수

if(n%2 ==0)

return power(x*x,n/2);

else

return x*power(x*x,(n-1)/2);

```
int power(int x , int n)
{
    if(n ==1)
        return x;

    return x*power(x,n-1);
}
```

```
MyMath t1 = new MyMath();
|
long startTime = System.nanoTime();
System.out.println(t1.power(3, 1500));

System.out.println(System.nanoTime() - startTime + " 나노세컨드");
```

```
Problems @ Javadoc Declaration
<terminated> main [Java Application] C:\WU
6351001347334244913
318000 나노세컨드
```

시간복잡도 : $O(n)$

```
int power(int x,int n)
{
    if(n==1)
        return x;

    //짝수
    if(n%2 ==0)
        return power(x*x,n/2);
    else
        return x*power(x*x,(n-1)/2);
}
```

```
<terminated> main [Java Application] C
6351001347334244913
125500 나노세컨드
```

시간복잡도 : $O(\log n)$

3-4 실습문제 (Expert)

하노이타워 문제를 재귀호출을 이용하여 풀이과정을 출력해보자.(원판의 개수가 늘어나도 풀려야 한다.)

- 한번에 하나의 원판만 옮길수 있다.
- 작은 원판 위에 큰 원판이 올수 없다.
- A에 있는 원판을 C로 모두 옮기는것이 목표다.



클래스명	HanoiTower	
메서드	resolve()	매개변수: 원판수,출발,임시,도착 내용: 원판을 도착지로 옮긴다. 리턴: 없음

```
HanoiTower ht = new HanoiTower();  
ht.resolve(5, 'A', 'B', 'C');
```

```
terminated: main java Application C:\ros  
원판 1을 A에서 B으로 옮긴다.  
원판 2를 A에서 C으로 옮긴다.  
원판 1을 B에서 C으로 옮긴다.  
원판 3를 A에서 B으로 옮긴다.  
원판 1을 C에서 A으로 옮긴다.  
원판 2를 C에서 B으로 옮긴다.  
원판 1을 A에서 B으로 옮긴다.  
원판 4를 A에서 C으로 옮긴다.  
원판 1을 B에서 C으로 옮긴다.  
원판 2를 B에서 A으로 옮긴다.  
원판 1을 C에서 A으로 옮긴다.  
원판 3를 B에서 C으로 옮긴다.  
원판 1을 A에서 B으로 옮긴다.  
원판 2를 A에서 C으로 옮긴다.  
원판 1을 B에서 C으로 옮긴다.
```

3-4 문제풀이 (Expert)

```
class HanoiTower {  
  
    void resolve(int n, char from, char tmp, char to)  
    {  
        if(n==1)  
            System.out.println("원판 1을 "+from + "에서 " + to + "으로 옮긴다.");  
        else  
        {  
            resolve(n-1, from, to, tmp);  
            System.out.println("원판 "+n+"를 "+from+"에서 "+to+"으로 옮긴다.");  
            resolve(n-1, tmp, from, to);  
        }  
    }  
}
```



THANK YOU



강사 박주병