오류처리, 제네릭

practice 10

실습 과제 제출 시 유의 사항

- 실습 과제 제출 시 패키지 형태로 압축된 소스코드와 보고서를 함께 이클레스에 제출하셔야 합니다.
- 각 문제마다 조건이 붙어 있는 경우 적색으로 강조되어 있습니다.
- 연장 제출은 제출 기한 종료 이후 7일 이내로만 받습니다. 필요시 강의 계획서의 조교 메일로 보내시면 됩니다.

이클립스 디버깅 명령어

	설명	방법
Step Into	한 문장씩 실행, 메소드를 만나면 메소드 안으로 진입	F6 또는  아이콘 클릭
Step Over	한 문장씩 실행, 메소드를 만나면 메소드 진입 안함	F5 또는  아이콘 클릭
Run to Line	지정된 문장까지 실행	Run->Run to Line 메뉴 또는 Ctrl+R
Resume	중단된 프로그램 다시 실행	F8 또는  아이콘 클릭
Terminate	프로그램 종료	Run->Terminate

예외 처리

- Try – Catch – Finally
- Try – With – Resources
- Throws

문제 1

- 다음 문장에 의하여 발생하는 예외는 무엇인가?

(1) int[] anArray = new int[3];
System.out.println(anArray[3]);

(2) String[] strs = new String[3];
System.out.println(strs[0].length());

(3) Integer.parseInt("abc");

(4) Object o = new Object();
Integer i = (Integer)o;

문제 2

- 사용자 정의 예외
  - 다른 예외와 구별하여 사용자 정의 예외 클래스를 만들어 예외 처리를 하는 것도 가능하다. 이는 보통 Exception 클래스의 서브클래스를 생성시켜 만든다.

```
class OwnException extends Exception{
    public OwnException(){
        super("My Own Exception");
    }
}
```

```
public static void throwExceptMethod() throws OwnException{
    throw new OwnException();          //사용자가 정의한 예외를 발생시킨다.
}
```

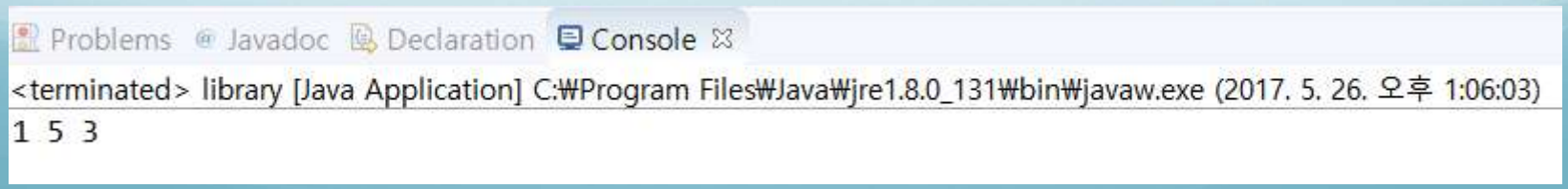
제네릭 사용 예시

```java
class any_class
{
    public static <T> void swap(T[] a, int i , int j)
    {
        T temp = a[i];
        a[i] = a[j];
        a[j] = temp;
    }
}
```

```java
public class library {
    public static void main(String args[])
    {
        base_class[] language = {new base_class(1), new base_class(3), new base_class(5)};

        any_class.swap(language,  1,  2);

        for (int i = 0; i < language.length; i++)
            language[i].print();
    }
}
```



와일드 카드 사용 예시

- 매개변수 / 반환 형태 제네릭 정의 시 사용
  - <? Extends 상위 타입> : 상위 클래스 제한
  - <? Super 하위 타입> : 하위 클래스 제한

```
class food<T>
{

}

class java_on_friday
{
    public static <eat> void choice(food<? extends Object> x, food<? extends ActionListener> y)
    {
    }
}
```

```
class java_on_wednesday
{
    public static <eat> void choice(food<? super any_class> x)
    {

    }
}
```

문제 3

- 아래의 코드가 오류 없이 컴파일되는가?
- 만약 컴파일되지 않는다면 원인은 무엇인가?

```
public final class MyAlgorithm {
    public static <T> T max(T x, T y) {
        return x > y ? x : y;
    }
}
```

문제 4

- 은행 예금을 나타내는 클래스 ClientAccount를 작성하라.
- ClientAccount 클래스의 명세는 아래와 같다.

필드

Balnace(잔액)

메소드

withdraw(출금), deposit(입금)

Withdraw()에서 인출 금액이 잔액보다 크면 **NegativeBalanceException을 발생**한다. ClientAccount클래스를 테스트하는 BankingTest 클래스를 작성하고 발생되는 예외를 try/catch를 이용하여 처리하여 보라.

문제 5

- 타입 매개 변수 T를 가지는 클래스 MyMath를 작성하여 보자. MyMath에는 평균을 구하는 getAverage() 메소드가 있다. getAverage 메소드는 숫자 형태의 자료형 만을 매개변수로 받는다. 와일드 카드를 이용하여 제네릭 자료형을 제한하여라.

- 1 ~ 6까지를 원소로 가지는 Integer 배열을 생성하고 작성한 MyMath를 이용하여 이것의 평균을 구해 보아라.