단위 테스트

- 문자열 계산기 구현

요구사항

- 쉼표(,) 또는 콜론(:)을 구분자로 가지는 문자열을 전달하는 경우 구분자를 기준으로 분리한 각 숫자의 합을 반환한다.(예: “” => 0, "1,2" => 3, "1,2,3" => 6, “1,2:3” => 6)
- 앞의 기본 구분자(쉼표, 콜론)외에 커스텀 구분자를 지정할 수 있다. 커스텀 구분자는 문자열 앞부분의 “//”와 “₩n” 사이에 위치하는 문자를 커스텀 구분자로 사용한다. 예를 들어 “//;₩n1;2;3”과 같이 값을 입력할 경우 커스텀 구분자는 세미콜론(;)이며, 결과 값은 6이 반환되어야 한다.
- 문자열 계산기에 음수를 전달하는 경우 RuntimeException으로 예외처리해야 한다.

요구사항

- Indent(들여쓰기) depth를 2단계에서 1단계로 줄여라.
- else를 사용하지 마라.
- method가 한 가지 일만 하도록 최대한 작게 만들어라.

리팩토링이란?

- 소스 코드의 가독성을 높이고 유지보수를 편하게 하기 위해 소스 코드의 구조를 변경하는 것을 의미한다.
- 리팩토링을 하더라도 기능상의 결과가 변경되는 것은 아니다. 리팩토링 작업 이전과 똑같은 기능을 해야 한다.

힌트 보기 전에

- 요구사항을 한번에 구현하려고 하지 말고 **작은 단위로 나눈다**.

  이 연습이 프로그래밍으로 구현하는 것보다 더 중요하다.
- http://docs.oracle.com/javase/8/docs/api/ java.lang package에 String 클래스를 보면 String 클래스가 지원하는 method를 참고해 구현한다.

실습

빈 문자열 또는 null 값을 입력할 경우 0을 반환해야 한다.(예 : “” => 0, null => 0)

힌트

```
if (text == null) {}
if (text.isEmpty()) {}
```

실습

숫자 하나를 문자열로 입력할 경우 해당 숫자를 반환한다.
(예 : “1”)

힌트

```
int number = Integer.parseInt(text);
```

실습

숫자 두개를 컴마(,) 구분자로 입력할 경우 두 숫자의 합을 반환한다.(예 : “1,2”)

힌트

```
String[] numbers = text.split(",");
```

실습

세 개 이상의 숫자를 컴마(,) 구분자로 입력할 경우 모든 숫자의 합을 반환한다.(예 : “1,2,3”)

실습

구분자를 컴마(,) 이외에 New Line을 사용할 수 있다.
(예 : “1,2₩n3”)

힌트

```
String[] tokens= text.split(",|\n");
```

실습

구분자를 컴마(,) 이외에 콜론(:)을 사용할 수 있다.
(예 : “1,2:3” => 6)

힌트

```
String[] tokens= text.split(",|:");
```

실습

“//”와 “₩n” 문자 사이에 커스텀 구분자를 지정할 수 있다.
(예 : “//;₩n1;2;3” => 6)

힌트

```
Matcher m = Pattern.compile("//(.)\n(.*)").matcher(text);
m.find();
String customDelimeter = m.group(1);
String[] tokens= m.group(2).split(customDelimeter);
```

실습

음수를 전달할 경우 RuntimeException 예외가 발생해야 한다. (예 : “-1,2,3”)

힌트

구글에서 “junit4 expected exception”으로 검색해 해결책을 찾는다.