

# 자료 구조 숙제 # 1

제출일 : 2020년 5월 3일 일요일 (eCampus)

Hw1.zip : LabTest.java, hw1.java, lab.in, lab.out, hw1.pdf

---

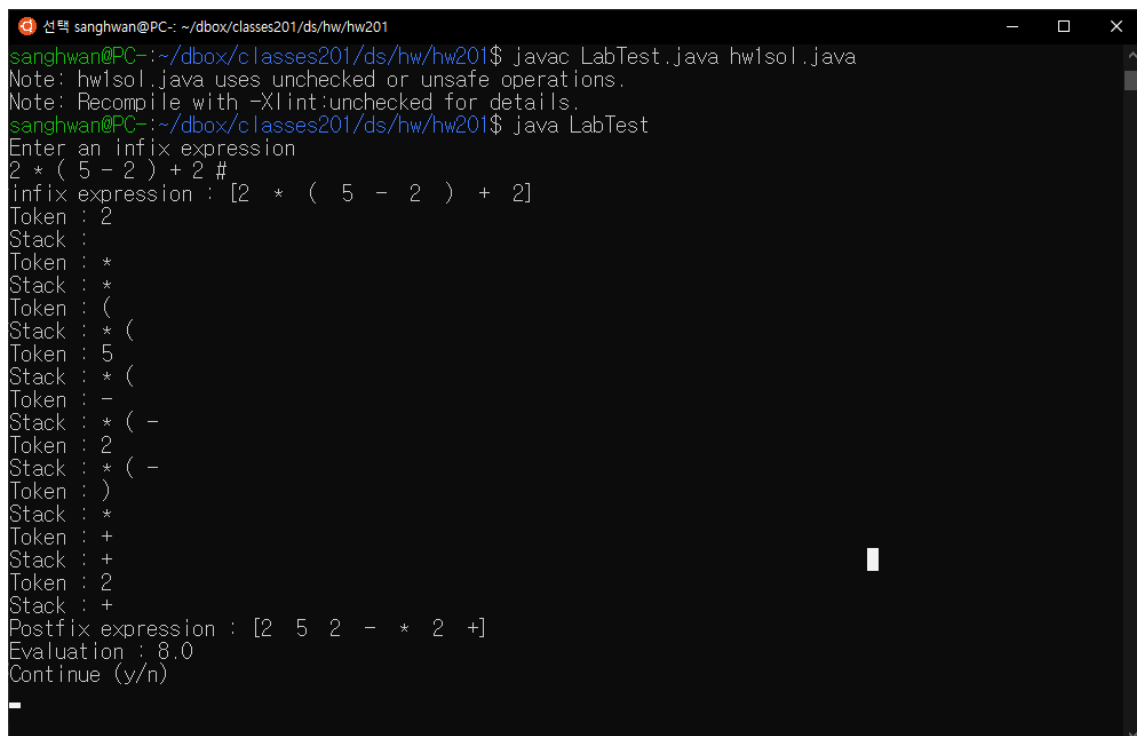
## 제출

hw1.java 를 학번.java 로 변경하여 이 파일 한 개만 제출할 것.

---

이번 숙제는 infix로 구성된 수식의 표현을 postfix로 변환한 후 그 식을 계산하는 내용이다.  
Infix 표현을 postfix 표현으로 변경하는 과정에서 **stack**의 구현이 필요하다.

수행 예는 다음과 같다.



```
선택 sanghwan@PC: ~/dbox/classes201/ds/hw/hw201
sanghwan@PC:~/dbox/classes201/ds/hw/hw201$ javac LabTest.java hw1sol.java
Note: hw1sol.java uses unchecked or unsafe operations.
Note: Recompile with -Xlint:unchecked for details.
sanghwan@PC:~/dbox/classes201/ds/hw/hw201$ java LabTest
Enter an infix expression
2 * ( 5 - 2 ) + 2 #
Infix expression : [2 * ( 5 - 2 ) + 2]
Token : 2
Stack :
Token : *
Stack : *
Token : (
Stack : * (
Token : 5
Stack : * (
Token : -
Stack : * ( -
Token : 2
Stack : * ( -
Token : )
Stack : *
Token : +
Stack : +
Token : 2
Stack : +
Postfix expression : [2 5 2 - * 2 +]
Evaluation : 8.0
Continue (y/n)
y
```

위 예에서 사용자가 infix expression을 입력하면 프로그램은 postfix expression을 출력하고, 그 식의 값도 함께 출력한다. Postfix expression을 처리하는 동안 각각의

Token (operator나 operand)을 처리할 때마다, Token의 내용을 보여주고, 그 처리가 끝난 후의 Stack의 내용을 보여준다. 예를 들면, 위 예제에서 중간쯤에 다음과 같은 2 라인이 나온다.

**Token 2**

**Stack : \* ( -**

첫번째 라인이 Token의 내용을 출력하는 라인이고, 두번째 라인이 Stack의 내용을 출력하는 라인이다.

그리고 계산 결과가 Evaluation: 라인에 출력된다.

또한 사용자의 y/n 대답에 따라 새로운 수식을 입력받거나 중단한다.

중위 수식 입력 시 각 operand와 operator는 공백으로 분리되어 있고, operand는 0에서 9까지의 단 단위 숫자를 사용하고, operator는 +, -, \*, /, (, ) 의 6개만을 사용한다.

사용자로부터의 입력은 끝에 항상 # 이 나오도록 하여 입력의 끝임을 표시한다.

Evaluation은 integer 연산을 사용하지 않고, double 타입의 연산을 사용한다. 즉 1/2는 integer 연산을 사용할 경우 0이 되지만 double 타입의 연산을 사용할 경우 0.5가 된다.

LabTest.java의 main() 함수에서는 다음 함수를 호출하게 된다.

- Expression.Eval(infix);

- infix는 infix 형식으로 입력받은 내용이 Vector<String> 타입으로 들어가 있는데, Vector<String>의 한 원소가 하나의 Token에 해당된다.

연산자의 우선 순위는 Java의 우선 순위를 따른다.

필요시 hw1.java 파일에 Expression 클래스뿐만 아니라 Stack 클래스도 구현한다.

주의:

실제 채점에서는 주어진 lab.in 이외의 다른 수식이 많이 사용될 것이기 때문에 lab.in의 내용 이외에 다른 수식도 임의로 입력하여 작성한 프로그램을 테스트 하기 바람.

## 프로그램 테스트

### 컴파일

```
$ javac hw1.java LabTest.java
```

## 실행

```
$ java LabTest
```

## 주어진 **input**으로 실행

```
$ java LabTest < lab.in
```

## 주어진 **output**과 비교

```
$ java LabTest < lab.in > abc
```

```
$ diff abc lab.out
```