REPORT

과목명 :	자료구조
담당 :	장형수 교수님
제출마감일 :	2021년 09월 20일
학과 :	생명과학과
학번 :	20171483
이름 :	조주현

SOGANG UNIVERSITY SOGANG UNIVERSITY

1. Problem description

1.. 문자열을 입력받고 그 문자열이 정해진 규칙을 따르는지 확인한다. 만약 따르지 않는다면 최소한의 문자를 삽입해 규칙에 맞게 수정하여 수정된 문자열과 추가한 문자의 수를 출력한다. 정해진 규칙을 따르면 well-formed라 정의하며 그 규칙은 다음과 같다.

1) 빈 문자열은 well-formed이다.
2) (,), [,], {, } 외의 다른 문자로 이루어져 있으면 well-formed이다.
3) 만약 특정 문자열 s가 well-formed라면 (s), {s}, [s]는 well-formed이다.
4) 두개의 well-formed 문자열을 이어 써도 well-formed이다.

2.. 분수 2개의 사칙연산을 계산하는 과제. 분수 2개는 (분자, 분모) 순으로 입력받고, (덧셈, 뺄셈, 곱셈, 나눗셈) 순서의 계산을 수행한다. 0으로 나눌 경우 오류를 출력하고, 계산 결과가 정수일 경우 정수로 출력, 음수일경우 '-'는 분자 앞에 붙힌다. 결과는 기약분수로 출력되어야 한다.

3.. 정수 한개를 입력받고 소수인지 여부를 판별, 소수가 아니라면 약수를 큰수부터 ' ,'로 구분하여 출력

2. Algorithm description

※ 1

1) 괄호('(', ')', '{', '}', '[', ']')가 아닌 문자들은 well-formed이기 때문에 well-formed를 판별할 때는 고려하지 않는다.
2) 먼저 맞는 쌍이란 '( )', '{ }', '[ ]' 로 정의한다.
3) 문자열을 읽어가며 열린 괄호들을 stack에 저장한다. 그 다음 문자를 읽을 때 stack에 있는 가장 위의 원소와 비교하여 맞는 쌍일 때 stack에 있는 해당 원소를 제거한다.
4) 위 방법을 반복하여 만약 맞지 않는 쌍이 나올 경우, 혹은 닫는 괄호가 나왔는데 stack이 비어있는 경우는 well-formed가 아니다. 문자열을 다 읽어도 위와 같은 경우가 나오지 않는다면 그 문자열은 well-formed이다.
5) 문자열을 수정할 때에는 1~3번을 반복하면서, 문자열의 문자와 stack에 있는 문자의 쌍이 맞지 않는 경우 stack의 문자와 맞는 쌍을 새로운 문자열(new_string)에 추가해준다. 이 때 반복문의 반복 횟수를 증가시키지 않는다.
6) stack이 비어있는 경우 새로 읽은 문자열과 맞는 쌍을 new_string에 추가해준다.
7) 입력받은 문자열의 문자와 stack의 문자가 쌍이 맞는 경우 stack의 문자를 제거하고 문자열의 문자만 new_string에 추가해준다.
8) 문자열을 모두 읽고 stack이 비어있으면 new_string을 출력, 만약 stack이 비어있지 않다면 stack에 있는 문자의 쌍에 맞게 new_string에 넣어주고, 동시에 stack의 원소를 제거한다.

※ 2

1) 입력받을 정수 4개와 연산기호를 구조체로 구성한다.
2) 4개의 정수를 입력받고 분모에 들어갈 수가 0인 경우 오류 출력
3) 4개의 정수를 이용해 분수의 계산을 수행한다. 이 때 기초적인 통분과정을 통해 분자와 분모를 계산한다.
4) 계산은 for문과 switch문을 이용해 사칙연산에 맞는 분자와 분모를 구해주고 기약분수를 만드는 함수에 분자와 분모를 넘겨준다.
5) 분자와 분모를 곱해 양, 음을 판단하고, 최대공약수를 이용해 기약분수로 바꿔준 뒤 출력한다.

※3

1) 입력받을 정수의 약수를 센다. 약수가 2개이면 소수, 그 외에는 소수가 아니다.
2) for문을 이용해 입력받은 정수와 그 정수부터 2까지 나눈 나머지가 0인 경우 출력, 마지막에 1을 출력한다.

3. Time complexity analysis of the algorithm (생략)

4. Program output

※Assignment 1

well-formed 입력

```
(()[xy]){}abc
Well formed string
Program ended with exit code: 0
```

not well-formed 입력

```
(x)))
Not well formed string
추가되는 괄호수 : 2
(x)()()
Program ended with exit code: 0
```

```
([(}))]
Not well formed string
추가되는 괄호수 : 3
([({})()])
Program ended with exit code: 0
```

※ Assignment 2

정상적인 입력

```
3 6 2 -5
(3 / 6) + (2 / -5) = 1 / 10
(3 / 6) - (2 / -5) = 9 / 10
(3 / 6) * (2 / -5) = -1 / 5
(3 / 6) / (2 / -5) = -5 / 4
Program ended with exit code: 0
```

0으로 나눈 경우 오류 출력

```
3 0 1 2
CAN'T DEVIDE WITH 0!!
Program ended with exit code: 0
```

```
3 4 1 0
CAN'T DEVIDE WITH 0!!
Program ended with exit code: 0
```

```
3 5 0 3
(3 / 5) + (0 / 3) = 3 / 5
(3 / 5) - (0 / 3) = 3 / 5
(3 / 5) * (0 / 3) = 0
CAN'T DEVIDE WITH 0!!
Program ended with exit code: 0
```

※ Assignment 3

소수인 경우

```
5
Prime number입니다.
Program ended with exit code: 0
```

소수가 아닌 경우

```
10
Prime number가 아닙니다.
약수는 : 10, 5, 2, 1
Program ended with exit code: 0
```

```
1
Prime number가 아닙니다.
약수는 : 1
Program ended with exit code: 0
```