Programming Project #1: Lexical Analyzer 구현

1. 문제

이 과제에서는 수(integer, real number)와 스트링으로 이루어진 수식(assignment 포함)의 어휘를 인식하는 lexical analyzer를 작성한다. lex를 사용하지 않고 state transition diagram을 그려 이를 기반으로 직접 구현한다 (구현언어: C 언어).

2. 수식의 예제

```
10 + 5 * 3
10.5 / 5
a = 10
a - 5
"hello"+"world"        // helloworld
"hello"+ 5             // hello5
"hello" * 2            // hellohello
b = "hello" + a        // hello10
b:2-3                  // "llo" - b string에서 2번째 문자에서 3글자
```

힌트] 이 예시는 어휘의 모습을 보여주기 위한 것으로 실제로 lexical analysis 단계에서는 수식의 문법구조나 타입 일치 등은 고려하지 않는다.

3. 언어의 어휘

Identifiers (ID)

- 영문자 대소문자 및 숫자로만 이루어짐.
- 첫 글자는 반드시 대소문자이고 숫자는 끝부분에서만 나타날 수 있고 중간에 나타나날 수 없음.
  바른 예) abc, Abc123, sum5
  잘못된 예) 123abc, a123bc
- 길이는 제한이 없으나 실제 구분은 첫 10자로 함.
  가령, a1234567890은 a123456789와 동일한 id임.

Integer

- 부호와 숫자로만 이루어진 정수. 단, 0을 제외하고는 0으로 시작하면 안되고 0은 부호가 없어야 함.
  예시) 0, 10, +10, -10

Real

- 기본적으로 C 언어의 double constant 표기법을 따름 (지수형은 제외)
  예시) 0.5, -123.123, +.123

String

- 기본적으로 C 언어의 표기법을 따름
- string 내에 "처리, multiple line string 등

Operators

+ - * / = :

기타 기호

;

[주의사항]

- 어휘 사이에 공백문자(blank, tab, new line)은 자유롭게 나타날 수 있다.
- operator 양쪽에는 공백이 없어도 된다.
- identifier, number들 사이는 operator가 없다면 하나 이상의 공백문자로 구분되어야 한다.

4. 입력 예시

```
start = 5;
len = 4;
str = "this is a string":start-len;
result = str:0-2 + "-" + str:3-1;
result = result + "relationship"
```

[힌트] 문장구조는 체크하지 않으니 실제 문법과 달라도 상관없다. 가령, 아래와 같이 입력해도 된다.

```
start len = 1 = 4
```

5. Symbol Table 및 String Table

- Identifier는 Symbol Table에 저장하고 string은 String Table에 저장한다.
- 동일한 identifier 또는 string은 각 table에 하나씩만 저장한다. 즉, 중복해서 저장하지 않는다.

<Symbol Table>

index	symbols
1	“start”
2	“len”
3	“str”
4	“result”

<String Table>

index	strings
1	“this is a string”
2	“_”
3	“relationship”

6. 출력 형식

- Lexical analyzer는 input text를 입력받아 아래와 같은 형식으로 token list와 Symbol Table 및 String Table을 출력한다.
- 하나의 token은 <token name, token value>로 표기한다.
- Identifier의 token value는 Symbol Table의 index이고, string의 token value는 String Table의 index이다.
- Token을 출력할 때 옆에 lexeme을 함께 출력한다.

[Token List 예시]

```
<ID, 1>            start
<ASSIGN, >         =
<INT, 5>           5
<SEMICOLON, >      ;
...
<ID, 3>            str
<ASSIGN, >         =
<STRING, 1>        "this is a string"
<COLON, >          :
...
```

[Table 출력 형식]

- 각자 설계해서 사용한다.

7. Lexical Error의 처리

- Lexical error가 있다면 error 정보를 출력한다.
- Error 정보는 error가 발생한 라인의 라인 번호 및 오류 문자 등을 포함해야 한다.

- Error 문자가 연속적으로 발생할 경우 묶어서 하나로 처리해도 되고 한글자씩 여러 번 error를 처리해도 된다.
- Error를 처리한 후 계속해서 lexical analysis를 진행한다.

[예시]

start = %;

<ID, 1> start
<ASSIGN, > =
Error: line 1 %
<SEMICOLON, > ;

7. 제출물

- 보고서 (소스코드 포함)

8. 보고서 형식

1) 서론
   - 과제 개요
   - **구현된 부분과 구현되지 않은 부분을 명확하게 명시할 것** (표를 사용)
2) 문제 분석
   - token의 종류
   - transition diagram (강의노트에서 제시된 것을 수정하여 사용)
3 설계
   - 중요 자료구조 및 프로그램 모듈 구조 및 모듈 설명
   - 강의노트의 설계 내용을 수정하여 사용 가능
4) 입력 데이터 2개 이상**(파일명 : ex1.txt, ex2.txt)**
   - 한 개는 정상적인 입력 텍스트(문법은 틀려도 됨)
   - 다른 하나는 lexical error가 포함된 입력 텍스트
   - 가능하면 모든 토큰의 종류가 포함되도록 함
5) 결과데이터 (화면 캡쳐)
   - Token List
   - Symbol Table
   - String Table
6) 첨부자료: source code

9. 제출방법 및 제출일

- "hw1_학번" 디렉토리를 만들어 보고서와 코드와 Makefile(*.c, *.h, makefile 등)를 넣는다.
- Makefile에서 생성되는 실행을 위한 파일의 이름은 "hw1_학번.exe"로 한다.
- 아주Bb 과제게시판에 "hw1_학번" 디렉토리 전체를 압축하여 올릴 것. (파일명: hw1_학번.zip)
- 제출일: 2022년 10월 7일(금) 23:59PM

  *** 보고서 출력본: 10월 6일 수업시간에 제출. 출력본에는 소스코드 및 실행화면은 포함하지 않음. Bb 제출본에는 '8. 보고서 형식'에 해당하는 내용을 모두 작성하여 제출.**

주의사항] 제출기한을 하루 초과시 5%감점. 제출기한 2일을 초과하는 경우 0점 처리