Programming Languages, 2022 Fall

Assignment: Lexer (Scanner)

목표:

본 과제에서는 컴파일러의 첫 단계인 Lexical Analyzer (Lexer 혹은 Scanner)를 구현하는 것을 목표로 합니다. Lexer 는 소스 파일을 연속된 bits/bytes 의 형태로 입력 받아 의미있는 토큰으로 구성하는 역할을 수행합니다. Lexer 에 의해 분석된 토큰은 향후 Syntax Analyzer 의 입력으로 사용되어 주어진 소스 파일이 문법에 맞게 작성되었는지를 판단하게 됩니다.

구현할 Lexer 는 다음 연산자와 기호(공백 포함)를 사용하여 토큰을 구분합니다. 즉, 입력으로 받은 연속된 비트 스트림을 다음 기호를 사용하여 토큰으로 나누게 됩니다.

+ - * / % < <= > >= = == != && || ! ; , . [ ] ( ) { } []

* '[', ']' 와 '[]'는 다른 기호입니다.

본 과제에서 구현해야 할 기능은 다음과 같습니다.

- 텍스트 형식으로 된 (확장자는 .npl) 소스 파일을 비트스트림으로 입력받습니다. (공백을 무시하여 비트 스트림으로 만들 것)
- 위에 제시된 기호를 기준으로 토큰으로 구분하여 저장을 합니다.
  - 찾은 토큰을 통계적으로 출력합니다.
  - 토큰명 : 등장 회수
- 키워드와 키워드가 아닌 토큰 (e.g., 변수, 상수, 함수 이름 등)을 구분합니다.
  - 키워드는 다음 장에 있습니다.
  - 키워드 및 기호가 아닌 것은 identifier 혹은 constant 입니다.
- Identifier 와 Constant 로 분류된 토큰의 유효성을 검사하고 오류가 있는 경우 해당 토큰의 위치 (줄 번호)를 출력합니다.
  - 이름 규칙

- ◆ 변수의 이름은 영문자(대소문자), 숫자, 언더스코어(_)로만 구성됩니다.
- ◆ 변수의 이름은 숫자로 시작될 수 없습니다.
- ◆ 변수의 이름 사이에는 공백을 포함할 수 없습니다.
- ◆ 변수의 이름으로 미리 정의된 키워드는 사용할 수 없습니다.

■ 실수 및 정수 규칙

- ◆ 숫자 사이에는 공백을 포함할 수 없습니다.
- ◆ 양수 기호는 있을 수도 있습니다. (예: +1)
- ◆ 소수점 이하만 표기할 수도 있습니다 (예: .123)
- ◆ 지수표기법에 따른 입력을 허용합니다. (예: 0.45 → 4.500000e-1)

키워드:

"int", "double", "bool", "string", "void", "break", "class", "else", "extends", "for", "if", "new", "null", "return", "this", "while", "static", "Print", "instanceof", "NewArray", "implements", "interface", "switch", "case", "default"

사용언어: C 언어

제출물: 소스코드, 실행화면을 포함한 간략한 보고서 (최대 3 장)

제출기한: 10 월 5 일 자정