프로그래밍 언어

(나)

20181259

조수민

목표

다음의 EBNF로 문법이 정의되는 언어를 위한 Parser를 구현하시오.

[C, C++, Java, Python 중에서 하나의 프로그래밍 언어를 선택하여 구현하시오.]주어진 bnf는 다음과 같다

<program> → {<declaration>} {<statement>}

<declaration> → <type> <var> ;

<statement> → <var> = <aexpr> ; | print <bexpr> ; | print <aexpr> ; |

do ‘ { ’ {<statement>} ‘ } ’ while ( <bexpr> ) ;

<bexpr> → <relop> <aexpr> <aexpr>

<relop> → == | != | < | > | <= | >=

<aexpr> → <term> {( + | - | \* | / ) <term>}

<term> → <number> | <var> | ( <aexpr> )

<type> → int

<number> → <dec>{<dec>}

<dec> → 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9

<var> → <alphabet>{<alphabet>}

<alphabet> → a | b | c | d | e | f | g | h | i | j | k | l | m | n | o | p | q | r |

s | t | u | v | w | x | y | z

설계

이전 구현과제 1에서 설계한 방식을 기반으로 설계했다.

구현과제 1과 다른 특징이 몇개 있는데,

1. 연산자의 우선순위가 존재하지 않음
2. Bexpr의 형식이 다름
3. Do while 문의 존재

까다로웠던 건 statement에 do while문이었다.

Current token을 확인하는 인덱스 pos를 do while문에 활용했다.

처음 do while문에 진입했을 때 pos를 temp\_pos에 저장해놓고, while문의 boolean값을 확인 후 true면 pos값을 temp\_pos - 1로 바꾸고 탈출했다.

때문에 다시 재귀로 돌아가 get\_token을 호출하면 이전의 do 토큰부터 호출이 되는 구조이다.

False일때는 pos가 이동한 채로 함수를 탈출하기 때문에 while문에서 탈출하는 것을 볼 수 있다.

구현

C 언어로 구현

* 타입텍스트, 스크린샷, 폰트이(가) 표시된 사진

  자동 생성된 설명

TokenType은 terminal-symbol과 Error로 나눔

VariableType 은 초기, Number, Boolean, Error

변수 구조체

텍스트, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명

Token – input을 token 단위로 구분하여 저장하는 구조체

Ex) input = “int x ; x = 2 ;”

Type = token\_INT value = “int”

Type = token\_VAR value = “x”

TYPE = token\_SEMI value = 2

TYPE = token\_VAR value = x

TYPE = token\_ASSIGN value = “=”

TYPE = token\_NUMBER value = 2

.

Variable – statement에서 “=” 기호로 유추되는 변수를 저장하는 구조체, ERROR\_MESSAGE 또한 변수로 가지고 있지만 해당 과제에서는 사용하지 않음

Ex) input = “int x ; x = 3 ;”

Variable[0]

Name = “x”

Value = “3”

isTrue = 0 (하지만 타입을 보고 사용하지 않음)

Type = type\_NUMBER

로직 설명

Program, declaration, statement 처럼 변수를 초기화 하거나 값을 세팅하는 문법은 void

이 외에 반환값이 있는 ex) a\_expr, term, number 함수는 int로 구현했다.

텍스트, 스크린샷, 폰트, 디자인이(가) 표시된 사진

자동 생성된 설명

Token및 input과 관련된 변수는 모두 전역변수이고, pos를 통해 token을 확인하고 var\_pos를 통해 변수를 초기화 한다.

텍스트, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명

각 단계에서 에러 메세지를 출력하는 기능이 있지만, 정확도가 떨어지고 오히려 헷갈리기에 해당 과제에서는 제외했다.

출력(print)를 처리할 때는 buffer에 먼저 기록 한다. 재귀가 종료되고Syntax\_error\_flag가 0이라면 buffer의 내용을 출력하고 1이면 syntax error를 출력하고 static 변수들을 다시 초기화 한다.

텍스트, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명

텍스트, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명

실행 결과

텍스트, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명

텍스트, 폰트, 스크린샷이(가) 표시된 사진

자동 생성된 설명