[Compiler] 2. Parser

2019067101 CSE 황인용

Overview

C-Minus Parser Implementation

Implementation Method : Yacc (Bison) - "cminus_parser"

[Compilation method and environment]

VirtualBox Linux: Ubuntu 18.0.4

Implementation

1. globals.h

Abstract Syntax Tree를 구성하는 Node의 유형을 다음과 같이 정의하였다. 또한 아래에 맞게 struct TreeNode 또한 수정하였다.

```
typedef enum {StmtK, ExpK, DecK, TypeK, ParamK, OptK} NodeKind;
typedef enum {CompoundK, IfK, IfelseK, WhileK, ReturnK} StmtKind;
typedef enum {AssignK, VariableK, OpK, ConstK, CallK, IdK} ExpKind;
typedef enum {VarK, FunK} DecKind;
typedef enum {IntK, VoidK} TypeKind;
typedef enum {SingleK, ArrayK} ParamKind;
typedef enum {RelopK, AddopK, MulopK} OptKind;
```

2. util.c

새로 설정한 Node type에 알맞는 생성 함수를 추가하고, printTree에서는 Node type에 알맞은 출력형식을 따랐다. 자세한 내용은 코드 참조.

3. cminus.y

Token과 그에 따른 Priority와 Associativity는 다음과 같이 정의했다. 기준은 C-Minus를 C의 속한다고 판단해 C의 Priority와 Associativity를 따랐다.

```
%token IF ELSE WHILE RETURN INT VOID
%token ID NUM
%token LPAREN RPAREN LBRACE RBRACE LCURLY RCURLY SEMI
%left TIMES OVER
%left PLUS MINUS
%left LT LE GT GE
%left EQ NE
%right ASSIGN
%left COMMA
%token ERROR
```

Type의 경우, INT/VOID는 tree→type을 통해 구분하고 단일 변수 / 배열은 Param Node에서는 tree→nodekind로 구분하고 나머지는 tree→child[0]에 따라 구분하였다.

ID, Type, Operator에 대한 정보를 저장하기 위해 하위 Node를 새로 설정하고 Reduce 시, 정보만 상위 Node에 저장하고 Free 함수를 통해 메모리를 비우게 하였다.

자세한 내용은 코드 참조.

Example

1. test.cm

Parser_Testcase_Makefile.zip의 test1.txt와 동일하다.

```
user@user-VirtualBox:~/2_Parser$ ./cminus_parser test.cm

C-MINUS COMPILATION: ./test.cm

Syntax tree:
  Function Declaration: name = gcd, return type = int
    Parameter: name = u, type = int
    Parameter: name = v, type = int
    Compound Statement:
      If-Else Statement:
        Op: ==
          Variable: name = v
          Const: 0
        Return Statement:
          Variable: name = u
        Return Statement:
          Call: function name = gcd
            Variable: name = v
            Op: -
              Variable: name = u
              Op: *
                Op: /
                  Variable: name = u
                  Variable: name = v
                Variable: name = v
  Function Declaration: name = main, return type = void
    Void Parameter
    Compound Statement:
      Variable Declaration: name = x, type = int
```

```
      Variable Declaration: name = y, type = int
      Assign:
        Variable: name = x
        Call: function name = input
      Assign:
        Variable: name = y
        Call: function name = input
      Call: function name = output
        Call: function name = gcd
          Variable: name = x
          Variable: name = y
user@user-VirtualBox:~/2_Parser$
```

2. test2.cm

Parser_Testcase_Makefile.zip의 test2.txt와 동일

```
user@user-VirtualBox:~/2_Parser$ ./cminus_parser test2.cm

C-MINUS COMPILATION: ./test2.cm

Syntax tree:
  Function Declaration: name = main, return type = void
    Void Parameter
    Compound Statement:
      Variable Declaration: name = i, type = int
      Variable Declaration: name = x, type = int[]
        Const: 5
      Assign:
        Variable: name = i
        Const: 0
      While Statement:
        Op: <
          Variable: name = i
          Const: 5
        Compound Statement:
          Assign:
            Variable: name = x
              Variable: name = i
            Call: function name = input
          Assign:
            Variable: name = i
            Op: +
              Variable: name = i
              Const: 1
```

```
      Assign:
        Variable: name = i
        Const: 0
      While Statement:
        Op: <=
          Variable: name = i
          Const: 4
      Compound Statement:
        If Statement:
          Op: !=
            Variable: name = x
              Variable: name = i
            Const: 0
          Compound Statement:
            Call: function name = output
              Variable: name = x
                Variable: name = i
user@user-VirtualBox:~/2_Parser$
```