Compiler Construction

Project #3: Semantic Analysis

제출일: 2021.12.22

컴퓨터소프트웨어학부

2019000982 이지원

Assignment

C-Minus Semantic Analyzer를 Yacc(bison)을 이용하여 구현한다.

Environment

기본 환경: Mac OS

Yacc(bison): Homebrew 3.2.17을 이용해 설치한 bison(GNU Bison) 2.3

Implementation & Code

1. **Symtab.h**

   Symtab.c에서 사용할 함수들의 선언을 하였다.

   Symtab.c에서 사용할 구조체인 ScopList, BucketList를 추가로 구현하였다.

2. **Symtab.c**

   ScopeList create_scope(char * name)
   ScopList find_scope(char * name)
   void st_insert(ScopList scope, char * name, ExpType type, int lineno, int location, int isFunc)
   void add_line(BucketList bucket, int lineno)
   BucketList st_lookup(ScopeList scope, char * name)
   BucketList st_lookat(ScopeList scope, char * name)
   Void insertFuncParam(char * func, ExpType type)
   Void printSymTab(FILE * listing)

   등의 함수를 symbol table을 만들기 위해 작성하였다.

3. **Analyze.c**

   Typecheck에 필요한 변수들을 상단에 선언하였다.

   static void insertNode(TreeNode * t)
   static void checkNode(TreeNode * t)

   를 우리가 과제에서 구현하고자 하는 c-minus에 맞게 변경하였다.

   Static void afterInsertNode(TreeNode * t)
   static void insertIOFunc(void)
   static void beforeCheckNode(TreeNode* t)

   의 함수들을 새롭게 추가하였다.

Execution Method

1. 3_Semantic 폴더 안의 파일들을 같은 위치에 저장한다.
2. cd 명령어를 이용해 저장한 위치로 이동한다.
3. 터미널에서 make 명령어를 통해 make clean 후 make를 진행한다.
4. cminus_semantic 실행 파일을 원하는 test file과 함께 실행시킨다.
   - 출력 결과를 새로운 파일에 작성하여 확인하고 싶으면 > [file name]을 붙인다.
   - Ex. ./cminus_semantic test.1.txt
   - Ex. ./cminus_semantic test.1.txt > result1.txt

Result

Make를 통한 build는 성공하였지만 symbol table과 typechecker를 구현하는 과정에서 문제가 생겨 test 파일들이 제대로 실행되지 않았다.