**컴파일러설계 Project3 Report**



컴퓨터소프트웨어학부

2018007974 김서연

1. 컴파일 방법과 환경

- 주어진 Makefile을 이용해 make함

텍스트이(가) 표시된 사진

자동 생성된 설명

- 환경: Ubuntu 16.04.7 / gcc 5.4.0

2. 구현 방법과 동작 방식

main.c

- NO\_CODE를 TRUE로 설정함

globals.h

- TreeNode 구조체에 char \* 형식의 scope를 추가함

symtab.h

- BucketListRec 구조체에 ExpType 형식의 type 변수와 DeclType 형식의 dtype, TreeNode\* 형식의 t를 추가함

- 각 scope를 관리하기 위해 ScopeListRec 구조체를 추가함

symtab.c

- scope의 수를 나타내는 변수 n\_scope와, 모든 scope를 저장하는 배열인 scopeArr을 만듦

- 각 scope마다 location을 관리하기 위해 location이라는 이름의 배열을 만듦

- scope 배열에 global, output, input에 해당하는 scope를 만드는 scope\_init()을 만듦

- 인자로 주어지는 스코프의 이름과 타입(funcS/localS/globS), 데이터 타입 등을 이용해서 새로운 scope를 만드는 scope\_create()을 만듦

- scope 배열 내에서 인자로 주어지는 이름에 해당하는 scope를 검색하고, 만약 없다면 -1을, 있다면 해당 인덱스를 반환하는 scope\_get\_index()를 만듦

- scope 배열에서 주어진 인자에 해당하는 인덱스에 있는 scope를 반환하는 scope\_get\_scope()를 만듦

- st\_insert()

① 인자 scope로 주어진 이름을 가지는 scope가 scopeArr에 존재하는지 확인하고 없다면 새로 만듦

② 찾은/만들어진 scope 내에 있는 BucketList에서 인자로 주어진 name에 해당하는 bucket이 있는지 확인하고, 없다면 새로 만들어서 추가하고 있다면 line number를 리스트에 추가함

- st\_lookup()에서는 현재 scope부터 global scope까지 올라가면서 인자로 주어진 name에 해당하는 bucket이 있는지 확인하고 없으면 -1을, 있으면 해당 bucket을 반환함

- printSymTab()은 디버깅을 위해 과제 명세에 맞게 적절한 내용을 출력하도록 함

analyze.c

- 과제 명세에 알맞은 에러들을 전부 ‘error\_’를 접두사로 가지는 함수로 만듦

: error\_typeError(), error\_symbolError(), error\_undeclError(), error\_redefError(), error\_funDefError(), error\_voidVarError(), error\_voidindexing()

- 현재 scope의 이름을 저장하는 current\_scope, scope의 계층 구조를 나타내는 scope\_stack, 이 스택의 크기를 나타내는 stack\_size를 만듦

- scope\_pop()과 scope\_push()는 scope\_stack 내의 scope를 관리함

- traverse()는 기존 코드를 건드리지 않았으므로 그대로 AST를 traverse함

- insertNode(): AST를 traverse하면서 symbol table을 만드는 함수

① 현재 node의 type과 subtype에 해당하는 적절한 동작을 함

② 각 노드에 대해 scope를 검색하거나 만들거나 insert하는 등의 symbol table build에 필요한 동작을 함

- postInsertNode()는 CompK node이거나 FuncK node인 경우, 각 경우에 대해 알맞은 scope 내에서 동작할 수 있도록 scope를 관리함

- buildSymtab()은 preorder로 insertNode()를 수행하고 postorder로 postInsertNode()를 수행한 뒤, TraceAnalyze 플래그가 설정돼있다면 printSymTab()을 이용해 만들어진 symbol table을 출력함

- preCheckNode()는 checkNode()에서 node가 DeclK나 StmtK인 경우 scope를 알맞게 쓸 수 있도록 관리함

- checkNode(): type checking을 수행하는 함수

① insertNode()와 마찬가지로 AST를 traverse하면서 semantic error가 있는지 확인함

② 각 node type에서 발생할 수 있는 error 조건을 확인하고 해당 조건에서 알맞은 오류 메시지가 출력되도록 함

3. 예제 적용 결과

- 주어진 예제들에 대해서 모두 정상적으로 작동함

- 아래는 그 중 하나의 실행 코드와 결과를 나타냄

![텍스트이(가) 표시된 사진

자동 생성된 설명]()

![텍스트이(가) 표시된 사진

자동 생성된 설명]()