컴파일러과제2(Parser)

201301800 구장회

(스크린샷이 깨져서 파일로도 첨부했습니다.)

Compliation method and environment

- 환경 : 가상머신환경(Ubuntu 14.04 LTS)
- 컴파일 : default Makefile을 수정하여 컴파일
  - gcc - Wall -g 옵션 사용
  - bison -d 옵션 사용

[Makefile]

KJH — koo@koo-VirtualBox: ~/Desktop/complier/project — ssh koo@192.168.219.181 —...

```
#
# Makefile for TINY
# Gnu C Version
# K. Louden 2/3/98
#

CC = gcc

CFLAGS = -Wall -g

OBJS_S = y.tab.o lex.yy.o main.o util.o symtab.o analyze.o code.o cgen.o

cminus: $(OBJS_S)
	$(CC) $(CFLAGS) $(OBJS_S) -o cminus -lfl

main.o: main.c globals.h util.h scan.h parse.h analyze.h cgen.h
	$(CC) $(CFLAGS) -c main.c

util.o: util.c util.h globals.h
	$(CC) $(CFLAGS) -c util.c

symtab.o: symtab.c symtab.h
	$(CC) $(CFLAGS) -c symtab.c

analyze.o: analyze.c globals.h symtab.h analyze.h
	$(CC) $(CFLAGS) -c analyze.c

code.o: code.c code.h globals.h
	$(CC) $(CFLAGS) -c code.c

cgen.o: cgen.c globals.h symtab.h code.h cgen.h
	$(CC) $(CFLAGS) -c cgen.c

lex.yy.o: cminus.l scan.h util.h globals.h
	flex cminus.l
	$(CC) $(CFLAGS) -c lex.yy.c -lfl

y.tab.o: cminus.y globals.h
	bison -d cminus.y --yacc
	$(CC) $(CFLAGS) -c y.tab.c

clean:
	-rm *.o *.output
	-rm lex.yy.c
	-rm y.tab.c y.tab.h
	-rm cminus


~
```

Explanation about how to implement and how to operate

[구현]

[globals.h]

1. enum type의 수정
기존 NodeKind에 declaration , parameter , type을 위한 변수를 추가
기존 StmtKind에 compound , while , return , call 을 위한 변수를 추가
기존 ExpKind에 Array을 위한 변수를 추가
추가적으로 singleparameter , arrayparameter를 위한 변수를 enum type에 추가

2. TreeNode 수정
1번에서 수정한 enum type을 기존 TreeNode 구조체에 반영함
추가적으로 배열의 이름,크기,type을 저장하는 구조체 선언

[util.c]

1. PrintTree 함수
과제 명세의 출력에 최대한 맞도록 수정함.

2. PrintToken
if then else 구절에 새로운 토큰 'THEN' 추가

3. TreeNode를 생성하는 함수들
globals.h에 추가한 enum type의 TreeNode를 생성하고 반환하는 함수를
기존 newStmtNode , newExpNode와 동일하게 구성

[cminus.l]
과제1번에서의 오류를 발견하고 수정함
(CURLY와 BRACE를 혼동해서 사용했던 문제점)

[cminus.y] - yacc/tiny.y 파일을 덮어쓰고 수정함

1. %token의 추가
WHILE , INT , VOID , RETURN 등

2. /* Grammer for Tiny */ 부분 수정
과제 명세의 Appendix A.2대로 작성하였습니다.

[실행]

project디렉터리에서
$ make clean
$ make
$ ./cminus [testfile]

Example and Result Screenshot

[테스트파일]

1.test.cm(tiny c compiler의 기본 테스트파일)

2.test_while.cm(while문 테스트)

KJH — koo@koo-VirtualBox: ~/Desktop/complier/project — ssh koo@192.168.219.181 —...

```
1  int call(int u , int v) {
2      int t;
3      t=5;
4      while(t) {
5          u=u+v;
6          t=t-1;
7      }
8      return u;
9  }
10
11
12 void main(void) {
13     int x; int y;
14     x=input(); y=input();
15     output(call(x,y));
16 }
```

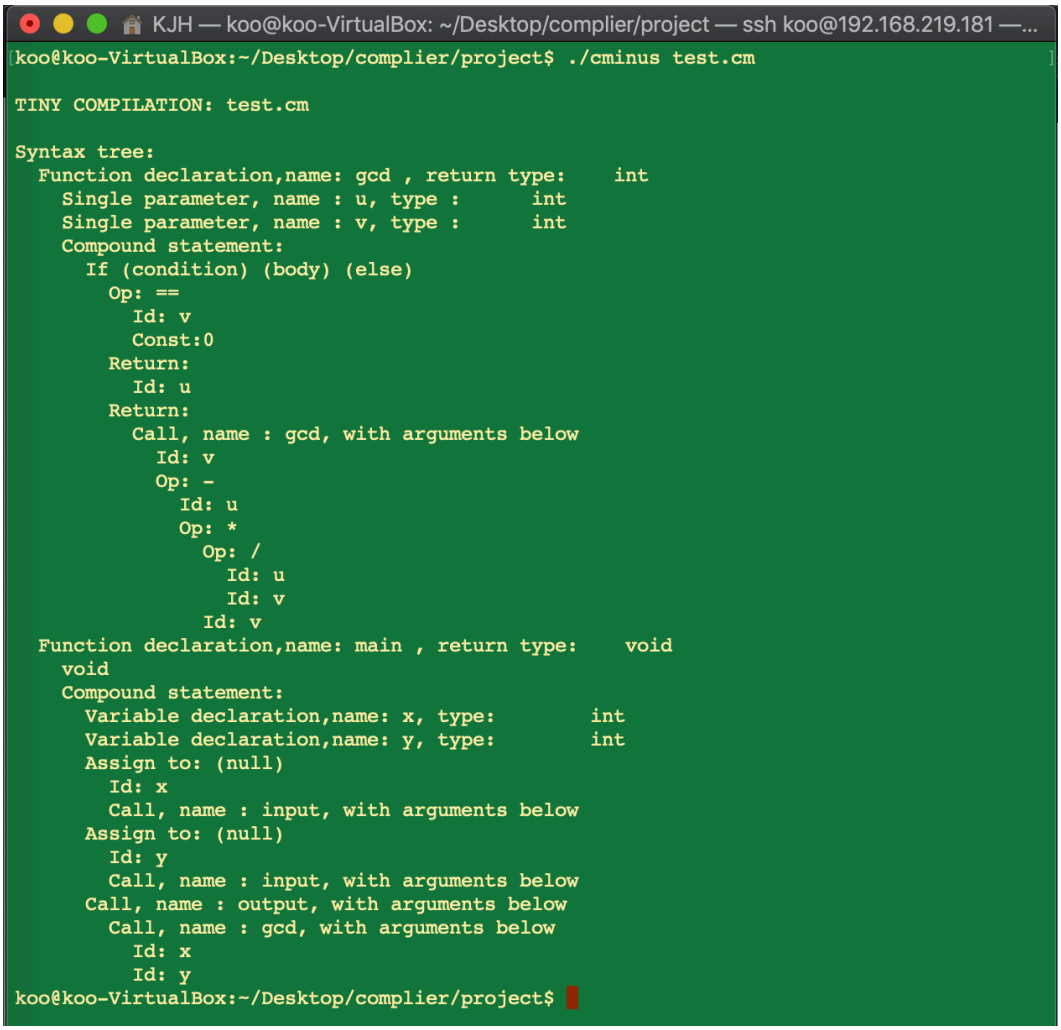
3.test_array.cm(array 테스트)

KJH — koo@koo-VirtualBox: ~/Desktop/complier/project — ssh koo@192.168.219.181 —...

```
1  int call(int u , int v , int arr[]) {
2      arr[0]=u+v;
3      arr[1]=u-v;
4      arr[2]=u/v;
5      arr[3]=u*v;
6      arr[4]=arr[0]+arr[1];
7      return 0;
8  }
9
10
11 void main(void) {
12     int x; int y;
13     int arr[5];
14     x=input(); y=input();
15     output(call(x,y,arr));
16 }
```

[결과]

1.test.cm(tiny c compiler의 기본 테스트파일)



2.test_while.cm(while문 테스트)
2.test_array.cm(array 테스트)

KJH — koo@koo-VirtualBox: ~/Desktop/complier/project — ssh koo@192.168.219.181 —...

```
koo@koo-VirtualBox:~/Desktop/complier/project$ ./cminus test_while.cm

TINY COMPILATION: test_while.cm

Syntax tree:
  Function declaration,name: call , return type:    int
    Single parameter, name : u, type :      int
    Single parameter, name : v, type :      int
    Compound statement:
      Variable declaration,name: t, type:        int
      Assign to: (null)
        Id: t
        Const:5
      While
        Id: t
        Compound statement:
          Assign to: (null)
            Id: u
            Op: +
              Id: u
              Id: v
          Assign to: (null)
            Id: t
            Op: -
              Id: t
              Const:1
      Return:
        Id: u
  Function declaration,name
    void
    Compound statement:
      Variable declaration,
      Variable declaration,
      Assign to: (null)
        Id: x
        Call, name : input,
      Assign to: (null)
        Id: y
        Call, name : input,
      Call, name : output,
        Call, name : call,
          Id: x
          Id: y
koo@koo-VirtualBox:~/Deskto
```

KJH — koo@koo-VirtualBox: ~/Desktop/complier/project — ssh koo@192.168.219.181...

```
koo@koo-VirtualBox:~/Desktop/complier/project$ ./cminus test_array.cm

TINY COMPILATION: test_array.cm

Syntax tree:
  Function declaration,name: call , return type:    int
    Single parameter, name : u, type :      int
    Single parameter, name : v, type :      int
    Array parameter, name : arr, size : 0 , type :      int
    Compound statement:
      Assign to: (null)
        Id: arr
          Const:0
        Op: +
          Id: u
          Id: v
      Assign to: (null)
        Id: arr
          Const:1
        Op: -
          Id: u
          Id: v
      Assign to: (null)
        Id: arr
          Const:2
        Op: /
          Id: u
          Id: v
      Assign to: (null)
        Id: arr
          Const:3
        Op: *
          Id: u
          Id: v
      Assign to: (null)
        Id: arr
          Const:4
        Op: +
          Id: arr
            Const:0
          Id: arr
            Const:1
      Return:
        Const:0
  Function declaration,name: main , return type:    void
    void
    Compound statement:
      Variable declaration,name: x, type:        int
      Variable declaration,name: y, type:        int
      Array declaration, name: arr, size:5, type :        int
      Assign to: (null)
        Id: x
        Call, name : input, with arguments below
      Assign to: (null)
        Id: y
        Call, name : input, with arguments below
      Call, name : output, with arguments below
        Call, name : call, with arguments below
          Id: x
          Id: y
          Id: arr
koo@koo-VirtualBox:~/Desktop/complier/project$
```