**C-Minus Project2 Report**

컴퓨터소프트웨어학부 박제우

1. 컴파일 환경/컴파일 방법

\* 컴파일환경: Ubuntu 16.04.7 LTS (Oracle VM VirtualBox에서 구동)

\* 프로젝트 1에서 구현한 Scanner + 프로젝트 설명에서 제공한 Makefile을 기초로 작성함

\* yacc -d cminus.y -> make all -> ./cminus\_parser (대상파일명) 으로 실행

2. Parser구현/작동방식 설명

- 우선 Syntax Tree만 출력하도록 main.c를 수정하고, ./yacc/tiny.y를 ./cminus.y로, ./yacc/globals.h를 ./globals.h로 덮어 쓴 상태에서 구현을 하였다.

- 기존 Tiny compiler에 해당하는 cminus.y를 우리가 구현하고자 하는 C-Minus compiler에 맞춰서 파일을 수정하고, 바꾼 명세에 따라 globals.h와 util.c파일을 수정하는 식으로 진행된다.

(1) globals.h

typedef enum {IfK,IfelseK,WhileK,AssignK,VariableK,ArrayK

,FunctionK,ParameterK,VoidParameterK,CompoundK,ReturnK

,CallK} StmtKind;

typedef struct treeNode

{ struct treeNode \* child[MAXCHILDREN];

struct treeNode \* sibling;

int lineno;

NodeKind nodekind;

union { StmtKind stmt; ExpKind exp;} kind;

union { TokenType op;

int val;

char \* name; } attr;

TokenType typename; // INT, VOID 구분 위해 추가, 상황에 따라 name과 함께 추가로 필요함

ExpType type; /\* for type checking of exps \*/

} TreeNode;

- 총 두 곳을 수정하였는데, 우선 Array, while문 등을 위한 StmtKind상태들을 추가하였고 프로젝트 명세에서 name과 type을 다 가지는 노드를 구현하기 위하여 TreeNode자료형에서 추가적으로 TokenType typename을 추가하였다.

(2) util.c

- printTree함수에서 프로젝트 명세에 맞도록 출력하는 문구를 수정하고 추가된 상태들에 대해서도 명시해준다. 추가한 내용 중 핵심적인 부분은 Variable Declaration 부분이다.

case VariableK:

case ArrayK:

fprintf(listing,"Variable Declaration: name = %s, ",tree->attr.name);

if (tree->typename == INT) fprintf(listing,"type = int");

else if (tree->typename == VOID) fprintf(listing,"type = void");

if (tree->kind.stmt == ArrayK) fprintf(listing, "[]");

fprintf(listing,"\n");

break;

Variable Declaration: name = x, type = int[]와 같은 방식으로 출력하는 내용인데, 우선 type = int등을 표시하는 부분 까지는 VariableK(변수선언)와 ArrayK(배열선언) 모두 동일하게 작성한다. 이후 ArrayK인 경우에만 맨 뒤에 []을 붙임으로써 type = int인 경우와 구분해서 출력한다.

(3) cminus.y

- 기존 Tiny compiler에 대해 작성된 코드를 프로젝트 명세에 맞춰 수정하였다. 이 중 주목할 부분으로 2가지가 있다.

static int savedNumber;

static int savedType; // Type 저장 위해 추가

…

%}

…

%token IF ELSE UNTIL RETURN WHILE ENDIF

%token ID NUM

%token ASSIGN EQ LT GE GT LE NE PLUS MINUS TIMES OVER

%token LPAREN RPAREN LBRACE RBRACE LCURLY RCURLY SEMI COMMA

%token ERROR

- 먼저 num 부분을 처리하기 위해 savedNumber를 추가하였고, type\_specifier(INT, VOID)를 처리하기 위해 savedType을 추가하였다. savedName에서처럼 이들 값이 상위 노드에서 쓰이는 것을 염두에 두어 추가한 것이다.

%nonassoc ENDIF

%nonassoc ELSE

…

selection\_stmt : IF LPAREN exp RPAREN sub\_stmt %prec ENDIF

{ $$ = newStmtNode(IfK);

$$->child[0] = $3;

$$->child[1] = $5;

}

| IF LPAREN exp RPAREN sub\_stmt ELSE sub\_stmt

{ $$ = newStmtNode(IfelseK);

$$->child[0] = $3;

$$->child[1] = $5;

$$->child[2] = $7;

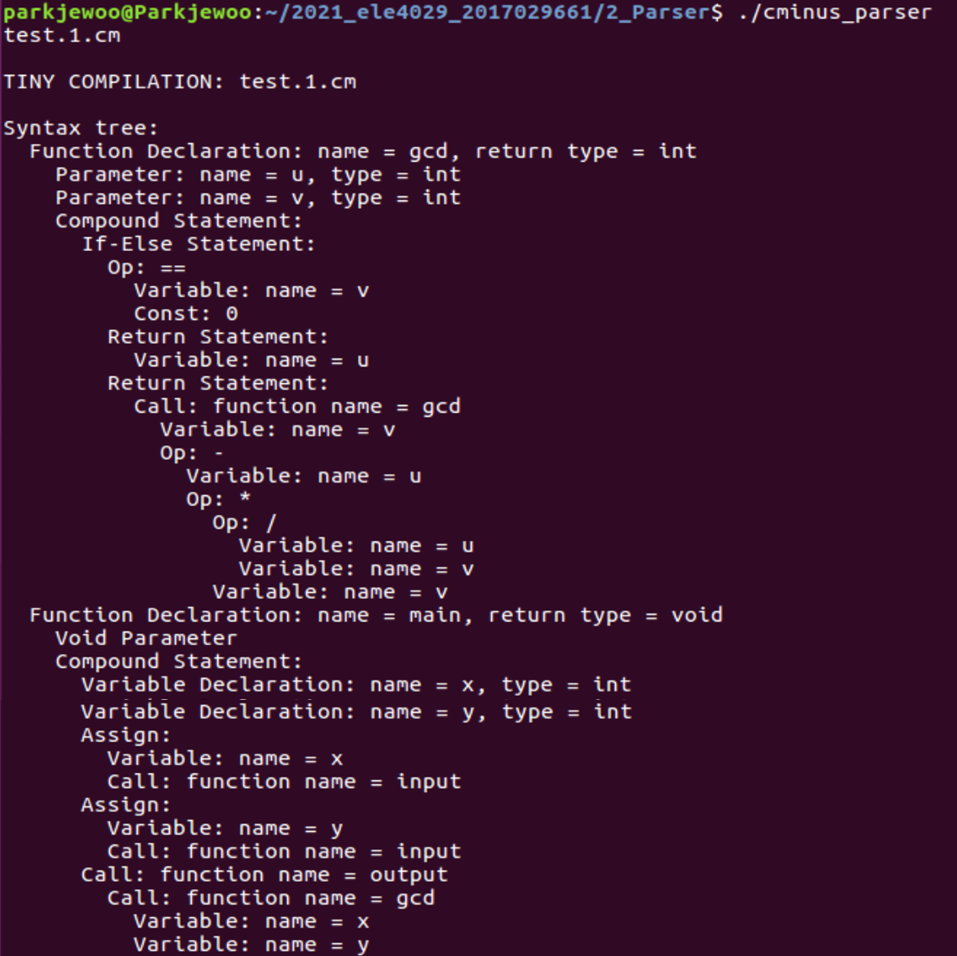
}

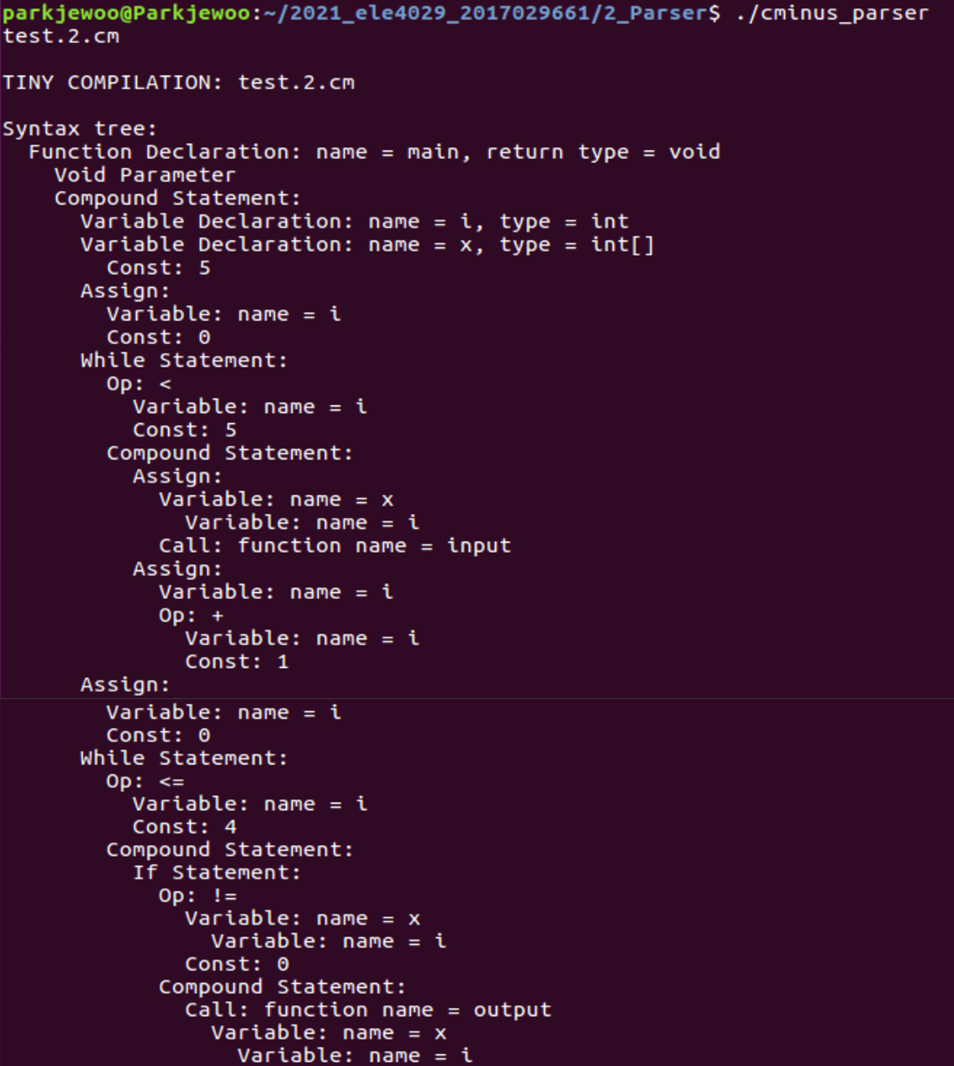
;

- 또한 If문과 If Else문으로 인한 Shift-Reduce conflict를 해결하기 위해 위와 같이 구현하였다. if if else 와 같은 코드가 있을 때 가까운 것부터 처리하여 (if (if else))처럼 처리 하기 위하여 기존 IF문에 ENDIF토큰을 끝에 추가하고 여기에 %prec을 주었다. 이와 같이 우선순위를 주는 것을 통하여 Shift-Reduce conflict를 해결할 수 있었다.

3. 실행 결과

- 제공된 test.1.txt와 test.2.txt파일명을 test.1.cm, test.2.cm으로 변경하고 (반드시 변경할 필요는 없다) yacc -d cminus.y -> make all -> ./cminus\_parser test.1.cm과 같이 실행해보니 결과가 잘 나온 것을 확인할 수 있다.





- 또한, 중첩된 if문을 가까운 순서대로 잘 Parsing하는지 체크하기 위해 프로젝트 설명에서 제시한 파일을 testif.cm으로 만들어 테스트 해본 결과 이 역시 결과가 제대로 출력되는 것을 확인할 수 있었다.

