

컴파일러 과제3

1976393 최윤지

2076278 윤지윤

2076285 이강희

1) 전체 소스코드

main.c

```
/*
 * main.c - 각 token에 대한 출력
 * programmer - 최윤지, 윤지윤, 이강희
 * date - 06 / 01 / 2023
 */
#include <stdio.h>
#include <stdlib.h>
#include "glob.h"
#include "tn.h"

extern int yyparse();
extern void PrintHStable();

void main() {

    printf("\n[Error Report : Error Information]\n");
    printf("=====
=====\n");

    // 각 변수 초기화
    cLine = 1;
    nextid = 0;
    nextfree = 0;

    // 파싱 및 에러 타입 출력
    yyparse();

    printf("=====
=====\n");
    printf("Parsing ends.***");

    printf("\n\t%d error(s) detected\n\n\n", cErrors);

    // H스테이블 출력
```

```
PrintHStable();

printf("WnWn1976393 최윤지, 2076278 윤지윤, 2076285 이강희Wn");
}
```

glob.h

```
/*
 * glob.h - global variable for the project
 * programmer – 최윤지, 윤지윤, 이강희
 * date – 06 / 01 / 2023
 */

#define STsize 1000 // ST 크기
#define HTsize 100 // HT 크기
#define FALSE 0
#define TRUE 1
#define MaxIdentLen 10 // identifier의 최대 길이

typedef struct HTentry* HTpointer; // HTentry를 가리키는 HTpointer

typedef struct HTentry {
    int index; // ST 읽기 위해 사용
    int type; // case 구분을 위해 사용
    int cLine; // 코드 line
    HTpointer next;
};

HTpointer HT[HTsize];
HTpointer look_id;
```

```

HTpointer look_tmp;

char ST[STsize];

int nextid; // 현재 identifier의 시작 index
int nextfree; // ST에서 다음 character가 삽입될 곳의 index
int hashcode;
int sameid;
int yyleng;
char* yytext;

/*
 * errorTypes
 * - wrong_st: 잘못된 statement인 경우
 * - wrong_funcdef: 잘못된 function 정의
 * - nosemi: semicolon 생략된 경우
 * - nobrace : 중괄호 생략된 경우
 * - nobracket: 대괄호 생략된 경우
 * - tlong: 11자 이상의 identifier인 경우
 * - toverflow: ST overflow가 발생한 경우
 */
enum errorTypes { wrong_st, wrong_funcdef, nosemi, nobrace, nobracket, tlong, toverflow };
typedef enum errorTypes ERRORtypes;
ERRORtypes err;

/*
 * cErrors: 에러 개수 세기
 * cLine: 코드 줄 수 세기
 * found: symtable.c 에서 사용하는 검색용 flag
 */
int cErrors; // 에러 개수
int cLine; // 코드 line
int found; // symtable.c에서 검색하기 위해 사용

```

scanner.l

```
%{
/* scanner.l - lexical analyzer for the MiniC
 * programmer – 최윤지, 윤지윤, 이강희
 * date – 06/01/2023
 */

#include <stdio.h>
#include <stdlib.h>
#include "tn.h" /* token name definition */
#include "glob.h" /* global variation */

extern ComputeHS(int nid, int nfree);
extern LookupHS(int nid, int nfree);
extern SymbolTable();
extern ADDHT(int hscore);
extern PrintHStable();
extern yyerror(s);
int i;

}%

%%

/*
 * Keyword(예약어)
 */
"const" {return(TCONST);}
"else" return(TELSE);
```

```

"if" return(TIF);
"int" return(TINT);
"return" return(TRETURN);
"void" {return(TVOID);}
"while" return(TWHILE);


/*
 * ILL Identifier : 숫자로 시작하는 경우
 */
[0-9]+[a-zA-Z_][a-zA-Z0-9_]* {
    yyerror("scanner error : start with digit identifier");
    return (TILLIDENT);
}


/*
 * 주석문
 */
W/W/.* ;


/*
 * Operator (연산자)
 */
"+" return(TPLUS);
"-" return(TMINUS);
"*" return(TMULTIPLY);
"/" return(TDIVIDE);
%" return(TMOD);
"=" {return(TASSIGN);}
"+=" return(TADDASSIGN);
"-=" return(TSUBASSIGN);
"*=" return(TMULASSIGN);
"/=" return(TDIVASSIGN);

```

```
"%=" return(TMODASSIGN);
"!" return(TNOT);
"&&" return(TAND);
"||" return(TOR);
"==" return(TEQUAL);
"!=" return(TNOTEQU);
"<" return(TLESS);
">" return(TGREAT);
"<=" return(TLESSE);
">=" return(TGREATE);
"++" return(TINC);
"--" return(TDEC);
```

```
/*
```

```
 * Special Symbols
```

```
*/
```

```
"(" {return(TLPAREN);}
")" { return(TRPAREN);}
"," {return(TCOMMA);}
"{" return(TLBRACE);
"}" return(TRBRACE);
"[" return(TLBRACKET);
"]" return(TRBRACKET);
";" {return(TSEMICOLON);}
```

```
/*
```

```
 * White spaces
```

```
*/
```

```
[ Wt] ;
[Wn] {
    cLine++;
```

```
}
```

```
/*
```

```
Identifier (명칭)
```

```
* 11자 이상일 경우, 에러 호출
```

```
* 아닌 경우, SymbolTable 호출 (ST삽입, hashCode계산, HT삽입 수행)
```

```
*/
```

```
[a-zA-Z_][a-zA-Z0-9_]* {
```

```
if(yylengtMaxIdentLen){
```

```
yyerror("scanner error :identifier is too long");
```

```
return (TLONG);
```

```
}
```

```
else{
```

```
SymbolTable();
```

```
return (TIDEN);
```

```
}
```

```
}
```

```
/*
```

```
* Constant (상수)
```

```
*/
```

```
[0-9]+ {return(TNUMBER);}
```

```
/*
```

```
* illegal Symbol
```

```
*/
```

```
. {yyerror("scanner error : illegal separator");}
```

```
%%
```

```
int yywrap(){  
    return 1;  
}
```


reporterror.c

```
/*
 * reporterror.c – error에 대한 report를 따로 처리
 * programmer – 최윤지, 윤지윤, 이강희
 * date – 06 / 01 / 2023
 */

#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "tn.h"
#include "glob.h"

extern char* yytext;
extern yylex();

/*
 * yyerror() - parse에러 외의 에러 출력
 */
yyerror(s)
char* s;
{
    if (s != "parse error") { //parser_tab.c에서 발생한 parse에러가 아닌경우
        printf("Wt%-5d %-10s %-50s\n", cLine, yytext, s); //에러 메세지 출력
        cErrors++; //에러수 1 증가
    }
}

/* reporterror
```

*에러문 출력함수

*cErrors = 에러 갯수 세는 변수, 에러가 있을 때마다 +1

*/

```
void reporterror(ERRORtypes err) {
```

```
    switch (err) {
```

```
    case 0: //잘못된 statement
```

```
        printf("%-5d %-15s %-50s\n", cLine, "< Error > ==>", " Wrong Statement");
```

```
        cErrors++;
```

```
        break;
```

```
    case 1: //올바르지 않은 함수 선언
```

```
        printf("%-5d %-15s %-50s\n", cLine, "< Error > ==>", " Wrong function definition");
```

```
        cErrors++;
```

```
        break;
```

```
    case 2://semicolon 생략됨
```

```
        printf("%-5d %-15s %-50s\n", cLine, "< Error > ==>", " Missing semicolon");
```

```
        cErrors++;
```

```
        break;
```

```
    case 3://중괄호 오류
```

```
        printf("%-5d %-15s %-50s\n", cLine, "< Error > ==>", " Missing brace");
```

```
        cErrors++;
```

```
        break;
```

```
    case 4: //대괄호 오류
```

```
        printf("%-5d %-15s %-50s\n", cLine, "< Error > ==>", " Missing bracket");
```

```
        cErrors++;
```

```
        break;
```

```
    case 5: //identifier 10자 초과
```

```
        printf("%-5d %-15s %-50s\n", cLine, "< Error > ==>", " Long identifier");
```

```
        cErrors++;
```

```
        break;
```

```
case 6: //table 사이즈 초과로 overflow
```

```
    printf("%-5d %-15s %-50s\n", cLine, "< Error > =>", " Overflow");
```

```
    cErrors++;
```

```
    break;
```

```
}
```

```
}
```

symtable.c

```
/*
```

```
* symtable.c – identifier들의 hashcode를 계산하고 st에 삽입
```

```
* programmer – 최윤지, 윤지윤, 이강희
```

```
* date – 06 / 01 / 2023
```

```
*/
```

```
#include <stdio.h>
```

```
#include <string.h>
```

```
#include <stdlib.h>
```

```
#include "glob.h"
```

```
#include "tn.h"
```

```
/* PrintHStable -
```

```
* hash table을 형식에 맞추어 출력
```

```
*/
```

```
void PrintHStable()
```

```
{
```

```
    HTpointer here;
```

```
    int i, j;
```

```
printf("\n[[HASH TABLE]]\n");
```

```
printf("=====
=====
=====");
```

```
for (i = 0; i < HTsize; i++) {
    if (HT[i] != NULL) {
        here = HT[i];
        do {
            printf("Hash Code%4d : (", i);
            for (j = here->index; ST[j] != '\0'; j++) printf("%c", ST[j]);
            printf(" : ");

            switch (here->type) {
                //int variable
                case 1: printf("integer scalar variable, line%d)\n", here->cLine); break;
                //void variable
                case 2: printf("void scalar variable, line%d)\n", here->cLine); break;
                //array(int) variable
                case 3: printf("integer array variable, line%d)\n", here->cLine); break;
                //void function
                case 4: printf("function, return type=void, line%d)\n", here->cLine); break;
                //not define
                case 0:
                case 5: printf("not defined identifier/function, line%d)\n", here->cLine); break;
                //int function
                case 6: printf("function, return type=int, line%d)\n", here->cLine); break;
                //int scalar parameter
                case 7: printf("integer scalar parameter, line%d)\n", here->cLine); break;

            }
            here = here->next;
        } while (here != NULL);
    }
}
```

```

    }

printf("=====
=====Wn");

}

//identifier의 hash code들을 모두 더하여 modulo연산을 수행
void ComputeHS(int nid, int nfree) {
    int code, i;
    code = 0;
    for (i = nid; i < nfree - 1; i++) {
        code += (int)ST[i]; //hash code더하기
    }
    hashcode = code % HTsize; //modulo
}

//identifier가 이전에 삽입된 적 있는지 살펴봄 이미 존재한다면 ST에 이미 존재하는 시작 index를
불러옴

void LookupHS(int nid, int hscore) {
    HTpointer here;
    int i, j;
    found = FALSE;
    if (HT[hscore] != NULL) {
        here = HT[hscore];
        while (here != NULL && found == FALSE) {
            found = TRUE;
            i = here->index;

```

```

        j = nid;
        sameid = i;
        while (ST[i] != 'W0' && found == TRUE) {
            if (ST[i] != ST[j]) {
                found = FALSE;
            }
            else {
                i++;
                j++;
            }
        }
        here = here->next;
    }
}
}

```

```

void ADDHT(int hscore){
//hash table에 새로운 identifier를 추가
//null이었으면 바로 추가하고, 이미 존재한다면 linked list로 연결
    HTpointer tmp;

    if (HT[hscore] == NULL) {
        tmp = (HTpointer)malloc(sizeof(struct HTentry));
        tmp->type = 0;
        tmp->next = NULL;
        tmp->cLine = cLine;
        HT[hscore] = tmp;
        tmp->index = nextid;
    }
    else {
        tmp = (HTpointer)malloc(sizeof(struct HTentry));

```

```

        tmp->type = 0;
        tmp->index = nextid;
        tmp->next = HT[hrcode];
        tmp->cLine = cLine;
        HT[hrcode] = tmp;
    }
    look_id = tmp;
}

```

//메인 함수

```
int SymbolTable()
```

```
{
```

```
    nextid = nextfree;
```

```
    // yytext 한글자씩 ST에 넣음
```

```
    for (int i = 0; i < yyleng; i++) {
```

```
        ST[nextfree++] = yytext[i];
```

```
    }
```

```
    // identifier를 다 넣은 후
```

```
    ST[nextfree++] = '\0';
```

```
    // HS계산, 각 identifier에 대해, hashtable에 이미 있는지 확인
```

```
    ComputeHS(nextid, nextfree);
```

```
    LookupHS(nextid, hashcode);
```

```
    //처음 등장한 identifier일 경우
```

```
    if (!found) {
```

```

        ADDHT(hashcode); // HT 삽입
    }
    //identifier가 이미 존재하는 경우

    else {
        nextfree = nexttid; //이번에 ST에 읽은 identifier를 다음에 덮어씀
    }

    //ST overflow 발생 시
    if (nextfree == STsize) {
        reporterror(tooverflow);
        PrintHStable();
        exit(1);
    }

    /*ST출력
    printf("===ST===\n");
    for (int i = 0; i < nextfree; i++) {
        if (ST[i] == 'W0') printf("\n");
        else printf("%c", ST[i]);
    }
    printf("\n\n\n");
    */
}

```


tn.h

```
#ifndef YYSTYPE
#define YYSTYPE int
#endif

#define TIDEN 258
#define TNUMBER 259
#define TCONST 260
#define TELSE 261
#define TIF 262
#define TINT 263
#define TRETURN 264
#define TVOID 265
#define TWHILE 266
#define TADDASSIGN 267
#define TSUBASSIGN 268
#define TMULASSIGN 269
#define TDIVASSIGN 270
#define TMODASSIGN 271
#define TOR 272
#define TAND 273
#define TEQUAL 274
#define TNOTEQU 275
#define TGREAT 276
#define TLESS 277
#define TGREATE 278
#define TLESSE 279
#define TINC 280
#define TDEC 281
#define TPLUS 282
#define TMINUS 283
```

```

#define TMULTIPLY    284
#define TDIVIDE      285
#define TMOD  286
#define TNOT  287
#define TASSIGN    288
#define TLPAREN    289
#define TRPAREN    290
#define TCOMMA     291
#define TSEMICOLON 292
#define TLBRACKET  293
#define TRBRACKET  294
#define TLBRACE     295
#define TRBRACE     296
#define TILLIDENT   297
#define TLONG 298
#define TOVER 299
#define LOWER_THEN_ELSE    300

```

```
extern YYSTYPE yylval;
```

parser.y

```

%{
/*
 * parser.y - yacc source for the MiniC
 * Programmer - 최윤지 윤지윤 이강희
 * date - 06 / 01 / 2023
 */

#include <stdio.h>
#include <stdlib.h>

```

```

#include <ctype.h>
#include <malloc.h>

#include "glob.h"

int type_int=0;
int type_void=0;
int param_int=0;


extern reporterror();
extern yylex();
extern yyerror(s);

%}

%token TIDEN TNUMBER TCONST TELSE TIF TINT TRETURN TVOID TWHILE
%token TADDASSIGN TSUBASSIGN TMULASSIGN TDIVASSIGN TMODASSIGN
%token TOR TAND TEQUAL TNOTEQU TGREAT TLESS TGREATE TLESSE TINC TDEC
%token TPLUS TMINUS TMULTIPLY TDIVIDE TMOD TNOT TASSIGN TLPAREN TRPAREN
TCOMMA TSEMICOLON
%token TLBRACKET TRBRACKET TLBRACE TRBRACE
%token TILLIDENT TLONG TOVER

%nonassoc LOWER_THEN_ELSE
%nonassoc TELSE

%%

mini_c : translation_unit
        ;

translation_unit : external_dcl
                 | translation_unit external_dcl

```

```

;
external_dcl      : function_def
    | declaration
    | TIDEN TSEMICOLON
    | TIDEN error
{
    yyerrok;
    reporterror(wrong_st); /* error - wrong statement */
}
| TRBRACE /* 왼쪽 중괄호 없음 */
{
    yyerrok;
    reporterror(nobrace);
}
;

```

```

function_def      : function_header compound_st
    |function_header TSEMICOLON
    |function_header error /* 비정상적인 함수 정의 */
{
    yyerrok;
    /* error - wrong function definition */
    reporterror(wrong_funcdef);
}
;

```

```

function_header   : dcl_spec function_name formal_param
;

```

```

dcl_spec          : dcl_specifiers
;

```

```

dcl_specifiers    : dcl_specifier
    | dcl_specifiers dcl_specifier
;

```

```

dcl_specifier     : type_qualifier

```

```

        | type_specifier
        ;

type_qualifier    : TCONST
        ;

/* 해당 하는 type 변수를 1로 설정*/
type_specifier    : TINT {type_int=1;} /* type : integer */
        | TVOID {type_void=1;} /* type : void */
        ;

function_name     : TIDEN

        {

/* identifier about parse error or not defined identifier/function*/
        if(look_id->type==0||look_id->type==5){

                if(type_void==1){
                        look_id->type=4; /* function, return type=void */
                }else if(type_int==1){
                        look_id->type=6; /* function, return type=int */
                }
                type_int=0;
                type_void=0;
                look_tmp=look_id;
        }
}

/* ill identifier, too long identifier, overflow 인 경우, scanner에서 에러 발생시킴*/
|TILLIDENT
|TLONG
|TOVER

```

```
;
```

```
formal_param      : TLPAREN opt_formal_param TRPAREN
```

```
;
```

```
opt_formal_param  : formal_param_list
```

```
|
```

```
;
```

```
formal_param_list : param_dcl
```

```
| formal_param_list TCOMMA param_dcl
```

```
;
```

```
/* 파라미터인 경우 */
```

```
param_dcl        : dcl_spec declarator{
```

```
    if(type_int==1){ /* int의 경우 */
```

```
        param_int=1;
```

```
        look_id->type=7; /* integer scalar parameter */
```

```
    }
```

```
    type_int=0;
```

```
    type_void=0;
```

```
    param_int=0;
```

```
}
```

```
;
```

```
compound_st : TLBRACE opt_dcl_list opt_stat_list TRBRACE
```

```
| TLBRACE opt_dcl_list opt_stat_list error /* 오른쪽 중괄호 없음 */
```

```
{
```

```
    yyerrok;
```

```
    reporterror(nobrace);
```

```
}
```

```
;
```

```

opt_dcl_list      : declaration_list
                  |
                  ;

declaration_list  : declaration
                  | declaration_list declaration
                  ;

declaration       : dcl_spec init_dcl_list TSEMICOLON{
                    type_int=0;
                    type_void=0;

                    }
                  | dcl_spec init_dcl_list error{
                    look_tmp->type=0;
                    yyerrok;
                    type_int=0;
                    type_void=0;
                    reporterror(nosemi); /* no semicolon error */
                    }
                  ;

init_dcl_list     : init_declarator
                  | init_dcl_list TCOMMA init_declarator
                  ;

init_declarator   : declarator
                  | declarator TASSIGN TNUMBER
                  ;

declarator        : TIDEN{
                    if(look_id->type==0){

                        if(type_int==1){
                            look_id->type=1; /* integer scalar variable */
                        }
                        else if(type_void==1)
                            look_id->type=2; /* void scalar variable */
                    }
                }

```

```

    }
    look_tmp=look_id;

}
| TIDEN TLBRACKET opt_number TRBRACKET
{

    if(look_id->type==0){
        if(type_int==1)look_id->type=3; /* integer array variable */

    }

    look_tmp=look_id;

}
|TIDEN TLBRACKET opt_number error{
    yyerrok;
    reporterror(nobacket); /* no bracket error */
}
|TILLIDENT
|TLONG;

```

opt_number : TNUMBER

;

opt_stat_list : statement_list

|

;

statement_list : statement

| statement_list statement

| statement_list declaration

;

statement : compound_st


```

        | expression_st
        | if_st
        | while_st
        | return_st
    ;

expression_st : opt_expression TSEMICOLON
              | expression error
              {
                  yyerrork;
                  reporterror(nosemi); /* no semicolon error */
              }
    ;

opt_expression : expression
              |
    ;

if_st : TIF TLPAREN expression TRPAREN statement %prec LOWER_THEN_ELSE
      | TIF TLPAREN expression TRPAREN statement TELSE statement
    ;

while_st : TWHILE TLPAREN expression TRPAREN statement
        ;

return_st : TRETURN opt_expression TSEMICOLON
        ;

expression : assignment_exp
        ;

assignment_exp : logical_or_exp
              | unary_exp TASSIGN assignment_exp
              | unary_exp TADDASSIGN assignment_exp
              | unary_exp TSUBASSIGN assignment_exp
              | unary_exp TMULASSIGN assignment_exp
              | unary_exp TDIVASSIGN assignment_exp
              | unary_exp TMODASSIGN assignment_exp
    ;

logical_or_exp : logical_and_exp
              | logical_or_exp TOR logical_and_exp
    ;

```

```

;
logical_and_exp  : equality_exp
                  | logical_and_exp TAND equality_exp
;
equality_exp     : relational_exp
                  | equality_exp TEQUAL relational_exp
                  | equality_exp TNOTEQU relational_exp
;
relational_exp   : additive_exp
                  | relational_exp TGREAT additive_exp
                  | relational_exp TLESS additive_exp
                  | relational_exp TGREATE additive_exp
                  | relational_exp TLESSE additive_exp
;
additive_exp     : multiplicative_exp
                  | additive_exp TPLUS multiplicative_exp
                  | additive_exp TMINUS multiplicative_exp
;
multiplicative_exp : unary_exp
                  | multiplicative_exp TMULTIPLY unary_exp
                  | multiplicative_exp TDIVIDE unary_exp
                  | multiplicative_exp TMOD unary_exp
;
unary_exp       : postfix_exp
                  | TMINUS unary_exp
                  | TNOT unary_exp
                  | TINC unary_exp
                  | TDEC unary_exp
;
postfix_exp     : primary_exp
                  | postfix_exp TLBRACKET expression TRBRACKET
                  | postfix_exp TLPAREN opt_actual_param TRPAREN
                  | postfix_exp TINC
                  | postfix_exp TDEC

```

```

        ;
opt_actual_param    : actual_param
        |
        ;
actual_param        : actual_param_list
        ;
actual_param_list   : assignment_exp
        | actual_param_list TCOMMA assignment_exp
        ;
primary_exp         : TIDEN
        {
            if(look_id->type==0)
                look_id->type=5; /* not defined identifier/function */

        }
        | TNUMBER
        | TLPAREN expression TRPAREN
        ;
%%

```

3) 직접 만든 에러 있는 테스트데이터 4개에 대한 input, output 캡처본

직접 만든 테스트데이터 1 (Error)

The screenshot shows a C compiler window with an error report on the left and the source code for 'error1' on the right. The error report lists two errors: 'Wrong function definition' and 'Wrong Statement'. The source code defines a 'main' function and a 'printB0' function, but the 'main' function has a return type of 'int' and a 'printf' statement, which is not defined.

```
[Error Report : Error Information]
=====
5 < Error >=> Wrong function definition
5 < Error >=> Wrong Statement
=====
Parsing ends.***      2 error(s) detected
=====

[[HASH TABLE]]
=====
Hash Code 6 : (YounJi : not defined identifier/function, line4)
Hash Code 6 : (j : integer scalar variable, line2)
Hash Code 7 : (k : integer scalar variable, line5)
Hash Code 21 : (main : function, return type=int, line1)
Hash Code 24 : (printBe : not defined identifier/function, line5)
Hash Code 59 : (printf : not defined identifier/function, line4)
=====

1976393 최윤지, 2076278 윤지윤, 2076285 이강희
계속하려면 아무 키나 누르십시오 . . .
```

```
error1 - Windows 메모장
파일(F) 편집(E) 서식(O) 보기(V) 도움말(H)
int main (){
    int j =0;
    if(j<19){
        printf(YounJi);
        int k = printBe();
    }
    return 0;
}
int printB0(){
    printf(Hi);
}
```

- Wrong function definition: 잘못된 function 정의
- Wrong Statement: 잘못된 statement
- not defined identifier/function: 정의되지 않은 identifier/function 사용
- integer scalar variable : int형 variable
- function, return type=int : 반환값이 int인 function

직접 만든 테스트데이터 2 (Error)

The screenshot shows a C compiler window with an error report on the left and the source code for 'error2' on the right. The error report lists three errors: 'scanner error : identifier is too long', 'Missing semicolon', and 'Missing brace'. The source code defines a 'main' function with a variable 'i' that is assigned a value of 100, but the 'if' statement is not properly closed.

```
[Error Report : Error Information]
=====
2 abcdefghijklmn scanner error : identifier is too long
6 < Error >=> Missing semicolon
13 < Error >=> Missing brace
=====
Parsing ends.***      3 error(s) detected
=====

[[HASH TABLE]]
=====
Hash Code 5 : (i : integer scalar variable, line3)
Hash Code 6 : (j : not defined identifier/function, line5)
Hash Code 21 : (main : function, return type=void, line1)
=====

1976393 최윤지, 2076278 윤지윤, 2076285 이강희
계속하려면 아무 키나 누르십시오 . . .
```

```
error2 - Windows 메모장
파일(F) 편집(E) 서식(O) 보기(V) 도움말(H)
void main() {
    int abcdefghijklmn = 1;
    int i =1;

    int j= 100
    if(i<100){
        i = 100;
    }else{
        i=10;
    }

    return 0;
}
```

- identifier is too long : identifier가 11자 이상인 경우
- Missing semicolon : 세미콜론 생략된 경우
- Missing brace : 중괄호 생략된 경우
- integer scalar variable : int형 variable
- not defined identifier/function: 정의되지 않은 identifier/function 사용
- function, return type=void : 반환값이 void인 function

직접 만든 테스트데이터 3 (Error)

```

[Error Report : Error Information]
=====
19 < Error > => Wrong function definition
19 < Error > => Missing brace
=====
Parsing ends.***      2 error(s) detected

[[HASH TABLE]]
=====
Hash Code  6 : (hee : integer scalar parameter, line3)
Hash Code  7 : (gh : function, return type=void, line3)
Hash Code 10 : (lee : integer scalar parameter, line7)
Hash Code 13 : (gang : integer scalar parameter, line3)
Hash Code 14 : (LEE : not defined identifier/function, line4)
Hash Code 19 : (ganghee : integer scalar variable, line1)
Hash Code 57 : (print : not defined identifier/function, line4)
=====

1976393 최윤지, 2076278 윤지윤, 2076285 이강희
계속하려면 아무 키나 누르십시오 . . .
  
```

```

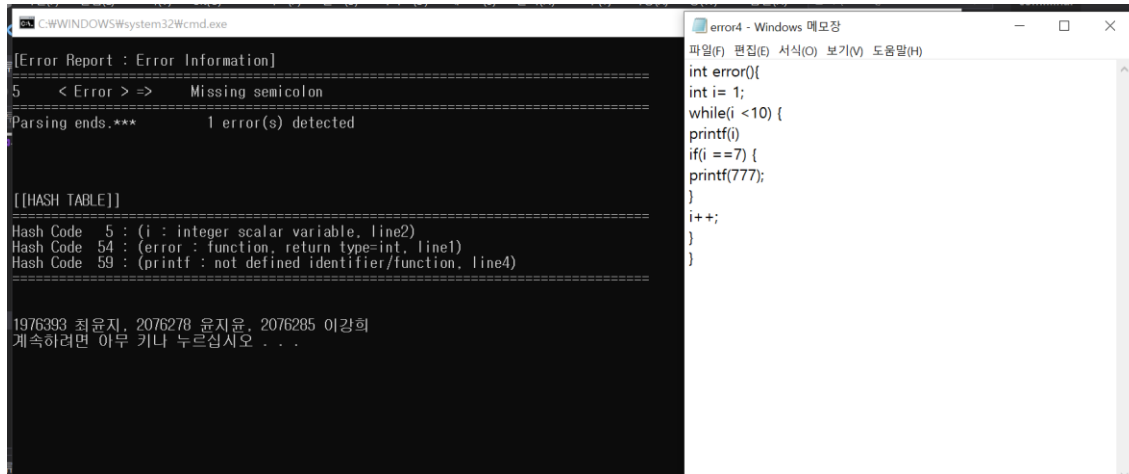
error3 - Windows 메모장
파일(F) 편집(E) 서식(O) 보기(V) 도움말(H)
int ganghee=123;

void gh(int gang, int hee) {
    print(LEE);
}

int lee(int gh) {
    return gh
}
  
```

- Wrong function definition: 잘못된 function 정의
- Missing brace : 중괄호 생략된 경우
- integer scalar parameter : int형 parameter
- function, return type=void : 반환값이 void인 function
- not defined identifier/function: 정의되지 않은 identifier/function 사용
- integer scalar variable : int형 variable

직접 만든 테스트데이터 4 (Error)



The screenshot shows a C compiler error report in a terminal window and the corresponding source code in a text editor. The error report indicates a 'Missing semicolon' on line 5. The source code is a function named 'error' that declares an integer variable 'i', enters a while loop, and uses 'printf' to print values. The error occurs because the 'printf' function is used before it is declared.

```
[Error Report : Error Information]
=====
5  <Error > =>    Missing semicolon
=====
Parsing ends.***      1 error(s) detected

[[HASH TABLE]]
=====
Hash Code  5 : (i : integer scalar variable, line2)
Hash Code 54 : (error : function, return type=int, line1)
Hash Code 59 : (printf : not defined identifier/function, line4)
=====

1976393 최윤지, 2076278 윤지윤, 2076285 이강희
계속하려면 아무 키나 누르십시오 . . .
```

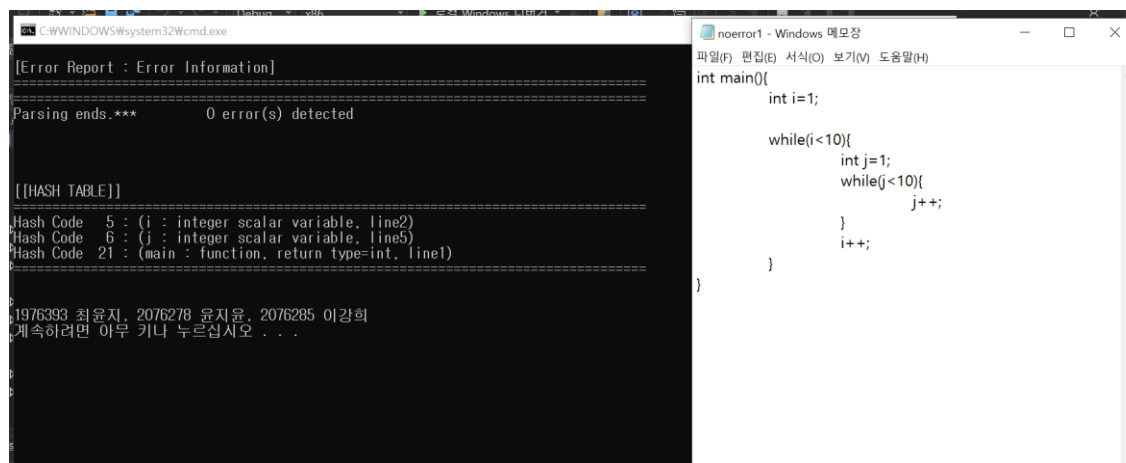
```
error4 - Windows 메모장
파일(F) 편집(E) 서식(O) 보기(V) 도움말(H)

int error(){
int i= 1;
while(i <10) {
printf(i)
if(i ==7) {
printf(777);
}
i++;
}
}
```

- Missing semicolon : 세미콜론 생략된 경우
- integer scalar variable : int형 variable
- function, return type=int : 반환값이 int인 function
- - not defined identifier/function: 정의되지 않은 identifier/function 사용

4) 직접 만든 에러 없는 테스트데이터 4개에 대한 input, output 캡처본

직접 만든 테스트데이터 1 (noError)



The screenshot shows two windows. The left window is a Windows command prompt titled 'C:\WINDOWS\system32\cmd.exe' with the following output:

```
[Error Report : Error Information]
=====
Parsing ends.***      0 error(s) detected

[[HASH TABLE]]
=====
Hash Code   5 : (i : integer scalar variable, line2)
Hash Code   6 : (j : integer scalar variable, line5)
Hash Code  21 : (main : function, return type=int, line1)
=====
1976393 최윤지, 2076278 윤지윤, 2076285 이강희
계속하려면 아무 키나 누르십시오 . . .
```

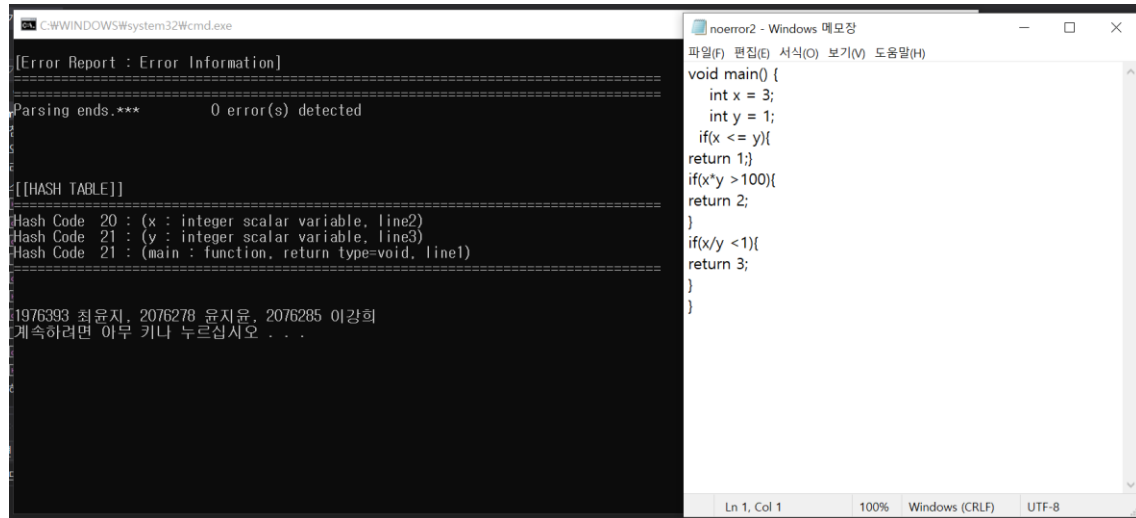
The right window is a text editor titled 'noerror1 - Windows 메모장' containing the following C code:

```
int main()
{
    int i=1;

    while(i<10){
        int j=1;
        while(j<10){
            j++;
        }
        i++;
    }
}
```

- integer scalar variable : int형 variable
- function, return type=int : 반환값이 int인 function

직접 만든 테스트데이터 2 (noError)



The screenshot shows two windows. The left window is a command prompt titled 'C:\WINDOWS\system32\cmd.exe' with the following output:

```
[Error Report : Error Information]
=====
Parsing ends.***      0 error(s) detected

[[HASH TABLE]]
=====
Hash Code 20 : (x : integer scalar variable, line2)
Hash Code 21 : (y : integer scalar variable, line3)
Hash Code 21 : (main : function, return type=void, line1)
=====
1976393 최윤지, 2076278 윤지윤, 2076285 이강희
계속하려면 아무 키나 누르십시오 . . .
```

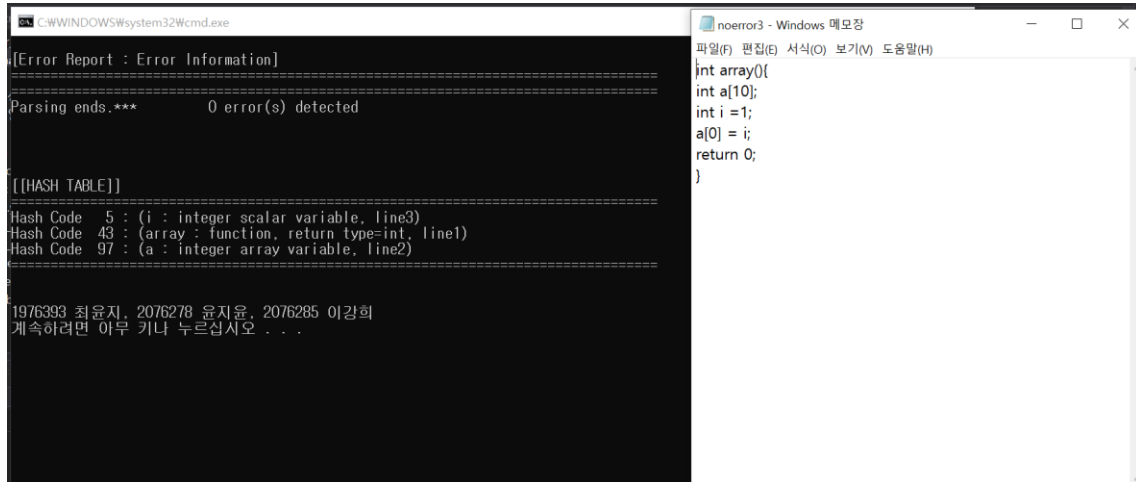
The right window is a text editor titled 'noerror2 - Windows 메모장' with the following C code:

```
파일(F) 편집(E) 서식(O) 보기(V) 도움말(H)
void main() {
    int x = 3;
    int y = 1;
    if(x <= y){
        return 1;
    }
    if(x*y > 100){
        return 2;
    }
    if(x/y < 1){
        return 3;
    }
}
```

The status bar at the bottom of the text editor shows 'Ln 1, Col 1', '100%', 'Windows (CRLF)', and 'UTF-8'.

- integer scalar variable : int형 variable
- function, return type=void : 반환값이 void인 function

직접 만든 테스트데이터 3(noError)



The screenshot shows two windows. The left window is a command prompt titled 'C:\WINDOWS\system32\cmd.exe' with the following output:

```
[Error Report : Error Information]
=====
Parsing ends.***      0 error(s) detected

[[HASH TABLE]]
=====
Hash Code   5 : (i : integer scalar variable, line3)
Hash Code  43 : (array : function, return type=int, line1)
Hash Code  97 : (a : integer array variable, line2)
=====

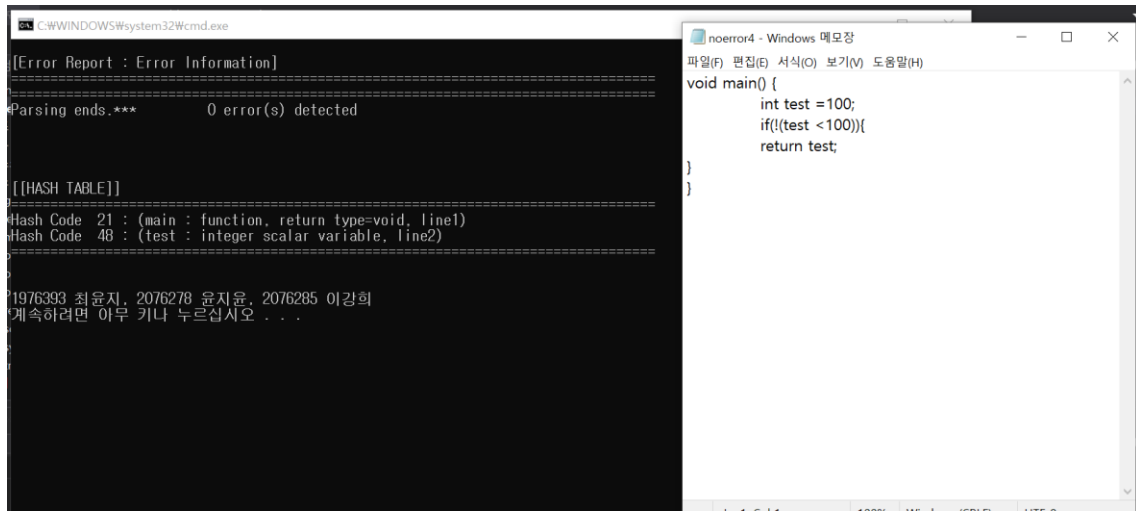
1976393 최윤지, 2076278 윤지윤, 2076285 이강희
계속하려면 아무 키나 누르십시오 . . .
```

The right window is a text editor titled 'noerror3 - Windows 메모장' containing the following C code:

```
int array(){
    int a[10];
    int i = 1;
    a[0] = i;
    return 0;
}
```

- integer scalar variable : int형 variable
- function, return type=int : 반환값이 int인 function
- integer array variable : int형 배열 variable

직접 만든 테스트데이터 4 (noError)



The screenshot shows two windows. The left window is a command prompt titled 'C:\WINDOWS\system32\cmd.exe' with the following output:

```
[Error Report : Error Information]
=====
Parsing ends.***      0 error(s) detected

[[HASH TABLE]]
=====
Hash Code 21 : (main : function, return type=void, line1)
Hash Code 48 : (test : integer scalar variable, line2)
=====

1976393 최윤지, 2076278 윤지윤, 2076285 이강희
계속하려면 아무 키나 누르십시오 . . .
```

The right window is a text editor titled 'noerror4 - Windows 메모장' containing the following C code:

```
void main() {
    int test =100;
    if(!(test <100)){
        return test;
    }
}
```

- integer scalar variable : int형 variable
- function, return type=void : 반환값이 void인 function

5) 과제에 대한 설명

1. 아래 5가지 경우에 대해 reporterror

- wrong_st: 잘못된 statement
- wrong_funcdef: 잘못된 function 정의
- nosemi: semicolon 생략된 경우
- nobrace : 중괄호 생략된 경우
- nobracket: 대괄호 생략된 경우

2. scanner error는 yyerror를 통한 에러 출력(ill identifier, too long identifier, overflow 인 경우, scanner에서 에러 발생시킴)

- TILLIDENT(ill identifier) : 지정된 규칙으로 생성되지않은 잘못된 identifier
- TLONG(too long identifier) : 11자 이상의 identifier
- TOVER (overflow) : symbol table이 overflow 된 경우