컴파일러 구성 과제#3

1605020 박소현

1642041 이경연

목차

[1.Code 3](#_Toc4491779)

[2.No error in the input data file 9](#_Toc4491780)

[3.With error in the input data file 12](#_Toc4491781)

[4.Data file given from professor 15](#_Toc4491782)

1.Code

1) tn.h

|  |
| --- |
| #ifndef YYSTYPE  #define YYSTYPE int  #endif  #define TIDEN 258  #define TNUMBER 259  #define TCONST 260  #define ELSE 261  #define TIF 262  #define TINT 263  #define TRETURN 264  #define TVOID 265  #define TWHILE 266  #define TEIF 267  #define TFLOAT 268  #define TCHAR 269  #define TSTRING 270  #define TADDASSIGN 271  #define TSUBASSIGN 272  #define TMULASSIGN 273  #define TDIVASSIGN 274  #define TMODASSIGN 275  #define TOR 276  #define TAND 277  #define TEQUAL 278  #define TNOTEQU 279  #define TGREAT 280  #define TLESS 281  #define TGREATE 282  #define TLESSE 283  #define TINC 284  #define TDEC 285  #define TPLUS 286  #define TMINUS 287  #define TMULTIPLY 288  #define TDIVIDE 289  #define TMOD 290  #define TNOT 291  #define TASSIGN 292  #define TLPAREN 293  #define TRPAREN 294  #define TCOMMA 295  #define TSEMICOLON 296  #define TLBRACKET 297  #define TRBRACKET 298  #define TLBRACE 299  #define TRBRACE 300  #define TNEWLINE 301  #define TERROR 302  #define TBLANK 303  #define TTAB 304  #define TEOF 305  #define LOWER\_THEN\_ELSE 306  #define TELSE 307  extern YYSTYPE yylval; |

2) glob.h

|  |
| --- |
| #pragma once  /\*  glob.h- global variable for the prject  programmer-  date-  \*/  #define STsize 1000 //size of string table  #define HTsize 100 //size of hash table  #define FALSE 0  #define TRUE 1  typedef struct HTentry \*HTptr;  typedef struct HTentry {  int index; //index of identifier in ST  int type;  int line;  HTptr next; //pointer to next identifier  }HTentry;  HTptr HT[HTsize];  char ST[STsize];  int nextid;  int nextfree;  int hashcode;  int sameid;  //나올 수 있는 여러 종류의 에러타입들을 미리 지정해둔다.  enum errorTypes { wrong\_st, nosemi, nobrace, nobracket, noparen, nocomma, nofuncname , assignerr, wrong\_funcdef, nofuncparam  };  typedef enum errorTypes ERRORtypes;  ERRORtypes err;  int cLine;  int cErrors;  int found;  HTptr look\_id;  HTptr look\_tmp; |

3) Hstable.c

|  |
| --- |
| #include <stdio.h>  #include <stdlib.h>  #include <string.h>  #include "glob.h"  //hash code를 계산해주는 함수  void ComputeHS(int nid, int nfree) {  int func, i;  func = 0;  //string table에 저장되어있는 변수 길이만큼 for문을 반복하며 값을 더해준다.(이 때, 저장되어있는 값은 문자이므로 정수값으로 바꿔줘야 한다.)  for (i = nid; i < nfree - 1; i++)  func += (int)ST[nid++];  //hash code값이 hash table의 크기를 벗어나면 안되므로 HTsize를 나눈 나머지를 hash code로 넣는다.  hashcode = func % HTsize;  }  //hash table에 해당 hash code값이 있는지를 확인해주는 함수  void LookupHS(int nid, int hscode) {  HTptr here;  int i, j,k;  int htid = 0, stid = 0;  found = FALSE; //found를 먼저 FALSE로 초기화시켜준다.  //hash table에서 hash code 값이 있을 경우  if (HT[hscode] != NULL) {  //hash table에 here 포인터 배치  here = HT[hscode];  while (here != NULL && found == FALSE) {  found = TRUE;  i = here->index;  j = nid;  sameid = i;  for (k = i; ST[k] != '\0'; k++) htid++;  for (k = j; ST[k] != '\0'; k++) stid++;  if (htid != stid) found = FALSE;  //다음 identifier까지 index 조정  while (ST[i] != '\0' && ST[j] != '\0' && found == TRUE) {  if (ST[i] != ST[j])  found = FALSE;  else {  i++;  j++;  }  }if (found == TRUE)return;  //hash table 다음 칸으로 조정  here = here->next;  }  }  }  //계산된 hash code값을 hash table에 저장해주는 함수  void ADDHT(int hscode) {  HTptr tmp;  //해당 hashcode 값 위치에 아무것도 들어가있지 않은 경우  if (HT[hscode] == NULL) {  tmp = (HTptr)malloc(sizeof(struct HTentry)); //HTpointer의 크기만큼 메모리 공간을 할당해준다.  tmp->type = 0;  tmp->next = NULL;  HT[hscode] = tmp;  tmp->index = nextid;  tmp->line = cLine;  }  //해당 hashcode 값 위치에 htpointer가 들어가있을 경우  else {  tmp = (HTptr)malloc(sizeof(struct HTentry)); //HTpointer의 크기만큼 메모리 공간을 할당해준다.  tmp->type = 0;  tmp->index = nextid;  tmp->next = HT[hscode];  HT[hscode] = tmp;  tmp->line = cLine;  }  look\_id = tmp;  }  //Hash Table에 저장되어진 값들을 출력해주는 함수  void PrintHStable() {  int i, j;  HTptr here;  printf("\n\n\n\n[[HASH TABLE]]\n\n");  //Hash Table의 크기만큼 반복  for (i = 0; i < HTsize; i++)  if (HT[i] != NULL) {  here = HT[i];  //Hash Table에서 here가 NULL이 될 때까지 here->next를 통해 다음으로 계속 반복  for (here = HT[i]; here != NULL; here = here->next) {  //j = here->index;  //String table에서 빈 칸이 나올 때까지 저장되어있는 문자값들을 출력함.  printf("HASH CODE %4d: (", i);  for (j = here->index; ST[j] != '\0';j++) {  printf("%c", ST[j]);  }  printf(" : ");  //여기 스위치  switch (here->type)  {  case 1: //정수 상수에 대한 변수일 경우  printf("integer scalar var");  printf(", line%d )\n", here->line);  break;  case 2: //실수 상수에 대한 변수일 경우  printf("float scalar var");  printf(", line%d )\n", here->line);  break;  case 3: //정수 배열에 대한 변수일 경우  printf("array integer var");  printf(", line%d )\n", here->line);  break;  case 4: //정수를 인자로 반환하는 함수일 경우  printf("function name, return type = int");  printf(", line%d )\n", here->line);  break;  case 5: //정의되지 않은 변수 또는 함수일 경우  printf("not defined id/func");  printf(", line%d )\n", here->line);  break;  case 6: //반환하는 인자가 없는 함수일 경우  printf("function name, return type = void");  printf(", line%d )\n", here->line);  break;  case 7: //실수를 인자로 반환하는 함수일 경우  printf("function name, return type = float");  printf(", line%d )\n", here->line);  break;  case 8: //문자를 인자로 반환하는 함수일 경우  printf("function name, return type = char");  printf(", line%d )\n", here->line);  break;  case 9: //문자에 대한 변수일 경우  printf("char scalar var");  printf(", line%d )\n", here->line);  break;  case 10: //실수 배열에 대한 변수일 경우  printf("array float var");  printf(", line%d )\n", here->line);  break;  case 11: //문자 배열에 대한 변수일 경우  printf("array char var");  printf(", line%d )\n", here->line);  break;  default: //에러  printf("id about parse err");  printf(", line%d )\n", here->line);  break;  }  // printf(" ");  }  printf("\n");  }  printf("\n\n\n <%5d characters are used in the string table > \n", nextfree);  } |

4) main.c

|  |
| --- |
| #include <stdio.h>  #include <stdlib.h>  #include "tn.h"  #include "glob.h"  extern void PrintHStable();  extern yyparse();  void main()  {  printf("=====ERROR REPORT=====\n");  printf("=================================\n");  cLine = 1; //첫 번째 줄부터 읽기 시작  cErrors = 0; //에러는 0개부터 시작(에러를 찾을 때마다 1씩 증가)  yyparse();    if(cErrors==0)printf("\n no error\n");  else printf(" %d error detected\n\n",cErrors);  PrintHStable();  } |

5) ReportError.c

|  |
| --- |
| #include <stdio.h>  #include <stdlib.h>  #include "tn.h"  #include "glob.h"  extern void line(int);  //extern void yylex();  yyerror(s)  char \*s;  {  if (strcmp("parse error", s)) {  line(cLine);  printf("< Scanner Error > => %s\n", s);  cErrors++;  }  }  //에러 종류에 따라 알맞는 문구 출력  void printError(ERRORtypes err) {  switch (err)  {  case 0:  line(cLine);  printf("< Error > => Wrong Statement!\n"); //잘못된 문장일 경우  cErrors++;  break;  case 1:  line(cLine);  printf("< Error > => Missing semicolon\n"); //세미콜론(;)이 없는 경우  cErrors++;  break;  case 2:  line(cLine);  printf("< Error > => Missing brace\n"); //'}'가 없는 경우  cErrors++;  break;  case 3:  line(cLine);  printf("< Error > => Missing bracket\n"); //']'가 없는 경우  cErrors++;  break;  case 4:  line(cLine);  printf("< Error > => Missing parenthesis\n");//')'가 없는 경우  cErrors++;  break;  case 5:  line(cLine);  printf("< Error > => Missing comma\n"); //','가 없는 경우  cErrors++;  break;  case 6:  line(cLine);  printf("< Error > => Missing function name\n"); //함수에 이름이 정의되지 않은 경우  cErrors++;  break;  case 7:  line(cLine);  printf("< Error > => Assign Error\n"); //잘못된 방식으로 assign할 경우(ex] 0=i)  cErrors++;  break;  case 8:  line(cLine);  printf("< Error > => wrong func def\n"); //함수를 잘못 정의한 경우  cErrors++;  break;  case 9:  line(cLine);  printf("< Error > => no func param\n"); //함수 파라미터가 지정되지 않은 경우  cErrors++;  break;  }  } |

6) scanner.l

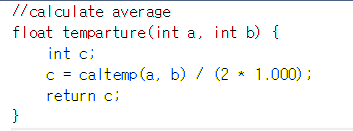
|  |
| --- |
| %{  /\*lex source for Mini C\*/  /\*실행을 위해 추가한 부분\*/  #include <stdio.h>  #include <stdlib.h>  #include "tn.h"  #include "glob.h"  extern ComputeHS(int nid, int nfree);  extern LookupHS(int nid, int nfree);  extern ADDHT(int hscode);  extern PrintHStable();  extern yyerror(char \*s);  int i;  %}  %s CMT  %%  "const" return(TCONST);  "else" return(TELSE);  "if" return(TIF);  "int" return(TINT);  "float" return(TFLOAT);  "char" return(TCHAR);  "return" return(TRETURN);  "void" return(TVOID);  "while" return(TWHILE);  "+" return(TPLUS);  "-" return(TMINUS);  "\*" return(TMULTIPLY);  "/" return(TDIVIDE);  "%" return(TMOD);  "=" return(TASSIGN);  "==" return(TEQUAL);  "!=" return(TNOTEQU);  "<=" return(TLESSE);  ">=" return(TGREATE);  "<" return(TLESS);  ">" return(TGREAT);  "!" return(TNOT);  "&&" return(TAND);  "||" return(TOR);  "++" return(TINC);  "--" return(TDEC);  "+=" return(TADDASSIGN);  "-=" return(TSUBASSIGN);  "\*=" return(TMULASSIGN);  "/=" return(TDIVASSIGN);  "%=" return(TMODASSIGN);  "(" return(TLPAREN);  ")" return(TRPAREN);  "," return(TCOMMA);  ";" return(TSEMICOLON);  "[" return(TLBRACKET);  "]" return(TRBRACKET);  "{" return(TLBRACE);  "}" return(TRBRACE);  [A-Za-z\_][A-Za-z0-9\_]\* {  if(nextfree>=STsize){ /\*다음 index가 string table의 크기를 넘는 범위에 있는 경우 overflow\*/  printf("\tError >> \tOVERFLOW!\n");  PrintHStable();  exit(1);    }  nextid = nextfree;  nextfree += (yyleng+1);  for(i = 0; i<yyleng; i++){ /\*string table에 문자 한개씩 차례로 넣어줌\*/  ST[nextid+i] = yytext[i];  }  ComputeHS(nextid,nextfree);  LookupHS(nextid,hashcode);  if(!found){  ADDHT(hashcode);  }  else  {  for(i=nextid; i<nextfree-1; i++){  ST[i]='\0';    }  nextfree=nextid;  }  }return(TIDEN);  0|[1-9][0-9]\* return(TNUMBER);  0[0-7]+ return(TNUMBER); /\*8진수\*/  0(x|X)[0-9A-Fa-f]+ return(TNUMBER); /\*16진수\*/  [+-]?([0-9]\*\.[0-9]+)([eE][-+]?[0-9]+)? return(TNUMBER); /\*실수\*/  \"(([^\"]|\\\")\*[^\\])?\" return(TSTRING);  [0-9]+[A-Za-z\_][A-Za-z0-9\_]\* {for(i=0;i<yyleng;i++)  if(yytext[i] == '\n')  cLine++;  yyerror("scanner error : start with digit identifier\n");} /\*변수가 숫자로 시작할 경우\*/  [ \t] ;  "\n" cLine++;  "//".\* ;  "/\*"([^\*]|\\*+[^\*/])\*\\*\*"\*/" {for(i=0;i<yyleng;i++)  if(yytext[i] == '\n')  cLine++;}    [A-Za-z\_]+[^A-Za-z0-9\_ \n\t(){}[]]+[A-Za-z0-9\_]\*? {yyerror("scanner error : illegal identifier1\n");}  [^A-Za-z0-9\_ \n\t(){}[]]+[A-Za-z0-9\_]\*? {yyerror("scanner error : illegal identifier2\n");}  . {yyerror("scanner error : illegal identifier3\n");}  %%  int yywrap() {  return 1;  } |

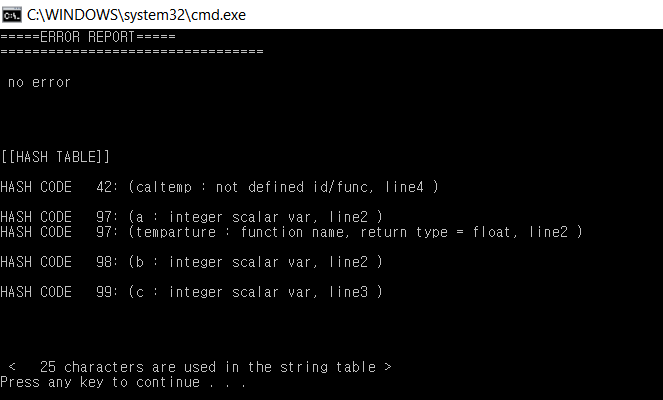
7) parser.y

|  |
| --- |
| %{  #include <stdio.h>  #include <stdlib.h>  #include <ctype.h>  #include <malloc.h>  #include "glob.h"  int type\_int = 0;  int type\_void = 0;  int type\_float = 0;  int type\_char = 0;  void line(int);  extern printError();  extern yylex();  extern yyerror(s);  %}  %token TIDEN TNUMBER TCONST TIF TINT TRETURN TVOID TWHILE TFLOAT TCHAR TSTRING  %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  %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;  printError(wrong\_st);  }  ;  function\_def : function\_header compound\_st  | function\_header TSEMICOLON  | function\_header error /\*error - 잘못된 함수 정의\*/  {  look\_tmp->type = 0; /\*type의 default 값은 0\*/  yyerrok;  printError(wrong\_funcdef);  }  ;  function\_header : dcl\_spec function\_name formal\_param  | dcl\_spec formal\_param /\*error - 함수 이름이 정의되지 않았을 경우\*/  {  yyerrok;  printError(nofuncname);  }  ;  dcl\_spec : dcl\_specifiers  ;  dcl\_specifiers : dcl\_specifier  | dcl\_specifiers dcl\_specifier  ;  dcl\_specifier : type\_qualifier  | type\_specifier  ;  type\_qualifier : TCONST  ;  type\_specifier : TINT {type\_int=1;} /\*실수형에 대한 토큰 type : int\*/  | TVOID {type\_void=1;} /\*void형에 대한 토큰 type : void\*/  | TFLOAT {type\_float=1;} /\*실수형에 대한 토큰 type : float\*/  | TCHAR {type\_char=1;} /\*문자형에 대한 토큰 type : char\*/  ;  function\_name : TIDEN  {  if(look\_id->type==0 || look\_id->type==5)  {  if(type\_int == 1)  {  look\_id->type=4; /\*정수를 인자로 반환하는 함수\*/  type\_int = 0;  type\_void = 0;  type\_float = 0;  type\_char = 0;  look\_tmp = look\_id;  }  else if(type\_void == 1)  {  look\_id->type=6; /\*반환하는 인자가 없는 함수\*/  type\_int = 0;  type\_void = 0;  type\_float = 0;  type\_char = 0;  look\_tmp = look\_id;  }  else if(type\_float == 1)  {  look\_id->type=7; /\*실수를 인자로 반환하는 함수\*/  type\_int = 0;  type\_void = 0;  type\_float = 0;  type\_char = 0;  look\_tmp = look\_id;  }  else if(type\_char == 1)  {  look\_id->type=8; /\*문자를 인자로 반환하는 함수\*/  type\_int = 0;  type\_void = 0;  type\_float = 0;  type\_char = 0;  look\_tmp = look\_id;  }  }  }  ;  formal\_param : TLPAREN opt\_formal\_param TRPAREN  | TLPAREN opt\_formal\_param error /\* error - ')'가 없을 경우\*/  {  yyerrok;  printError(noparen);  }  ;  opt\_formal\_param : formal\_param\_list  |  ;  formal\_param\_list : param\_dcl  | formal\_param\_list TCOMMA param\_dcl  | formal\_param\_list param\_dcl /\*error - ','가 없을 경우\*/  {  yyerrok;  printError(nocomma);  }  ;  param\_dcl : dcl\_spec declarator  ;  compound\_st : TLBRACE compound TRBRACE  | TLBRACE compound error  {  yyerrok;  printError(nobrace);  }  ;  compound : opt\_dcl\_list opt\_stat\_list  ;  opt\_dcl\_list : declaration\_list  |  ;  declaration\_list : declaration  | declaration\_list declaration  ;  declaration : dcl\_spec init\_dcl\_list TSEMICOLON  {  type\_int = 0; /\*값 다시 초기화\*/  type\_void = 0;  type\_float = 0;  type\_float = 0;  type\_char = 0;  }  | dcl\_spec init\_dcl\_list error /\*error - ';'가 없을 경우\*/  {  look\_tmp->type = 0;  yyerrok;  type\_int = 0; /\*값 다시 초기화\*/  type\_void = 0;  type\_float = 0;  type\_char = 0;  printError(nosemi);  }  ;  init\_dcl\_list : init\_declarator  | init\_dcl\_list TCOMMA init\_declarator  ;  init\_declarator : declarator  | declarator TASSIGN TNUMBER  | declarator TASSIGN TSTRING /\*스트링 상수가 assign될 경우\*/  | declarator TASSIGN function\_st  ;  declarator : TIDEN  {  if(look\_id->type==0)  {  if(type\_int==1)  look\_id->type=1; /\*정수 상수에 대한 변수\*/  else if(type\_float==1)  look\_id->type=2; /\*실수 상수에 대한 변수\*/  else if(type\_char==1)  look\_id->type=9; /\*문자에 대한 변수\*/  }  look\_tmp = look\_id;  }  | TIDEN TLBRACKET opt\_number TRBRACKET  {  if(look\_id->type == 0)  {  if(type\_int==1)  look\_id->type=3; /\*정수 배열에 대한 변수\*/  else if(type\_float==1)  look\_id->type=10; /\*실수 배열에 대한 변수\*/  else if(type\_char==1)  look\_id->type=11; /\*문자 배열에 대한 변수\*/  look\_tmp = look\_id;  }  }  | TIDEN TLBRACKET opt\_number error  {  yyerrok;  printError(nobracket);  }  ;  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  {  yyerrok;  printError(nosemi);  }  ;  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  ;  function\_st : TPLUS function\_expression /\*함수 뒤에 나올 연산식을 위한 nonterminal 정의\*/  | TMINUS function\_expression  | TMULTIPLY function\_expression  | TDIVIDE function\_expression  | TMOD function\_expression  ;  function\_expression : TLPAREN addtive\_exp TRPAREN  | TLPAREN addtive\_exp error /\*error - ')'가 없을 경우\*/  {  yyerrok;  printError(noparen);  }  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  | opt\_number TASSIGN assignment\_exp /\*잘못된 assign이 될 경우\*/  {  yyerrok;  printError(assignerr);  }  ;  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 : addtive\_exp  | relational\_exp TGREAT addtive\_exp  | relational\_exp TLESS addtive\_exp  | relational\_exp TGREATE addtive\_exp  | relational\_exp TLESSE addtive\_exp  ;  addtive\_exp : multiplicative\_exp  | addtive\_exp TPLUS multiplicative\_exp  | addtive\_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  | postfix\_exp TLBRACKET expression error  {  yyerrok;  printError(nobracket);  }  ;  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;  }  | TNUMBER  | TLPAREN expression TRPAREN  ;  %%  void line(int n)  {  printf("\t %d \t", n);  } |

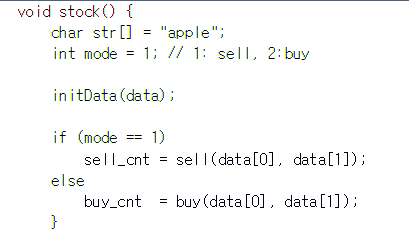
2.No error in the input data file

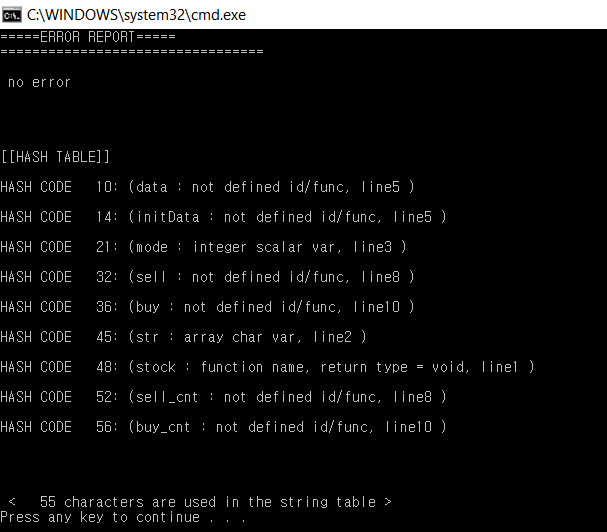
2.1 Input data\_1



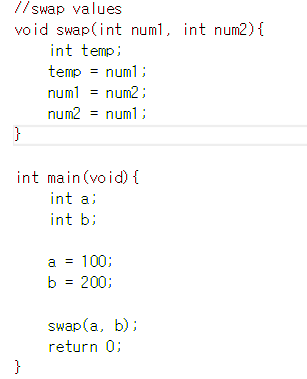


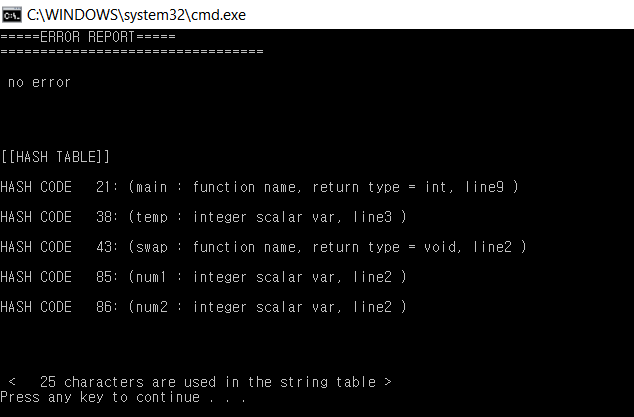
2.2 Input data\_2



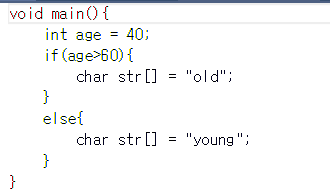


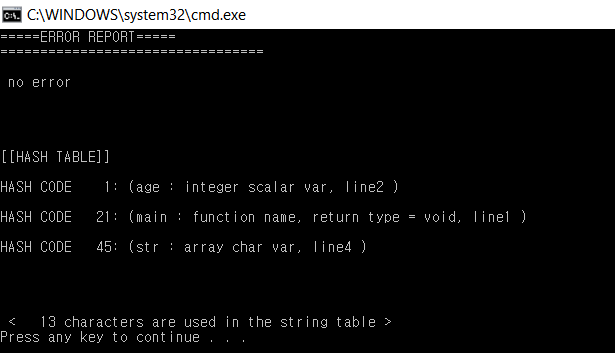
2.3 Input data\_3





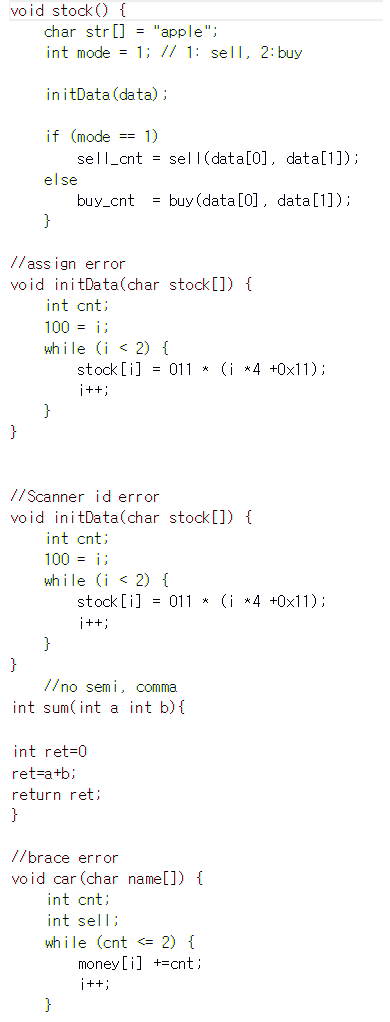
2.4 Input data\_4

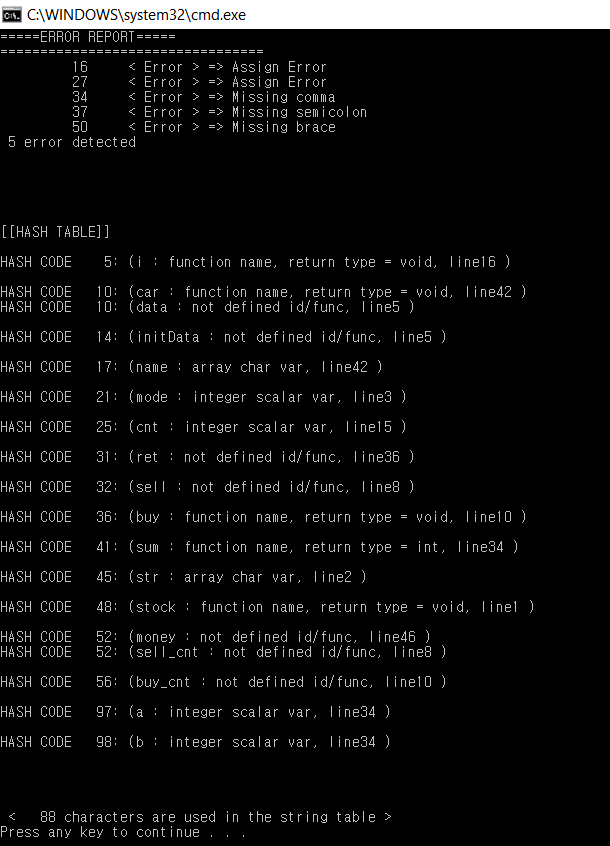




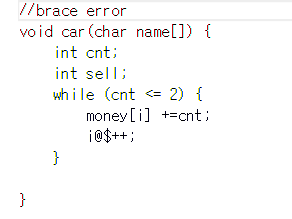
3.With error in the input data file

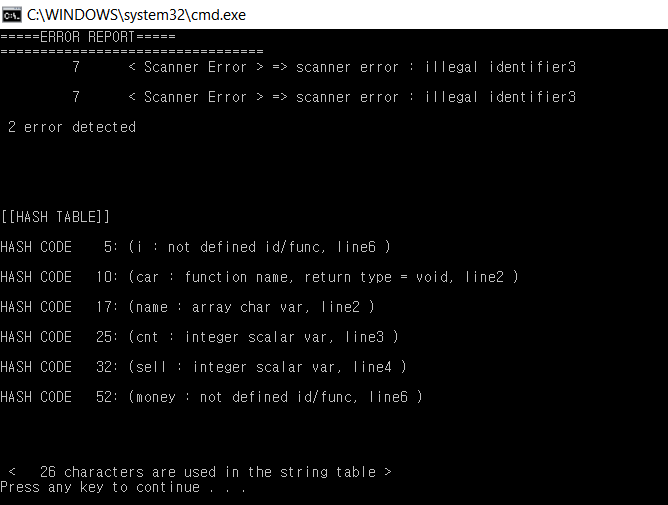
3.1 Input data\_1



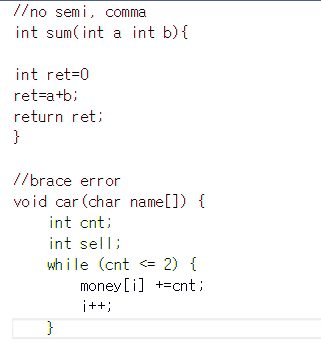


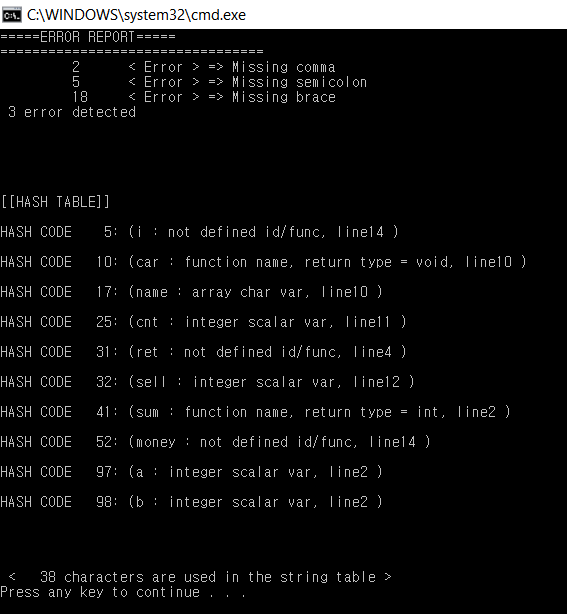
3.2 Input data\_2



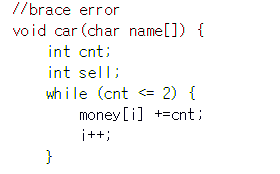


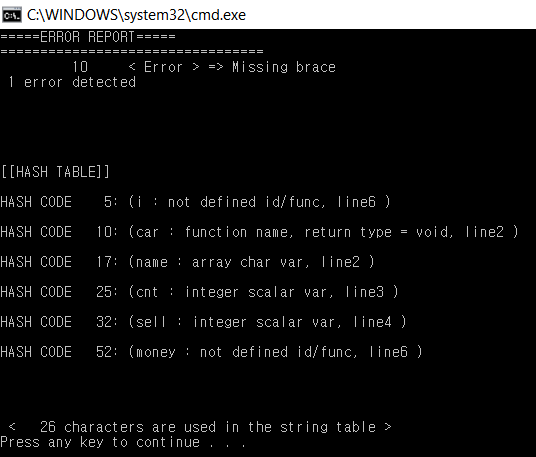
3.3 Input data\_3





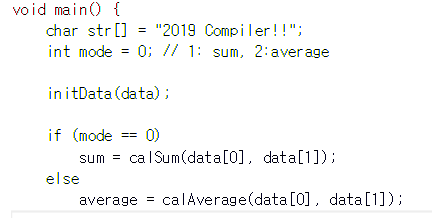
3.4 Input data\_4

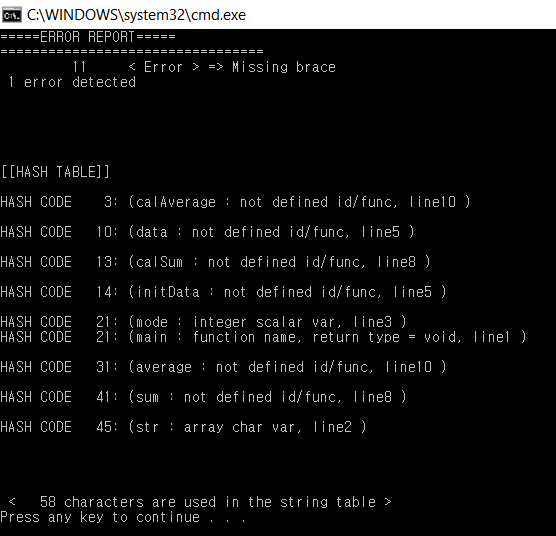




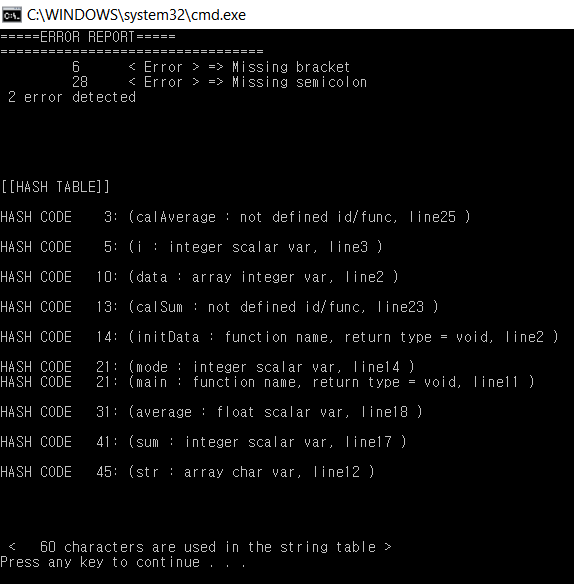
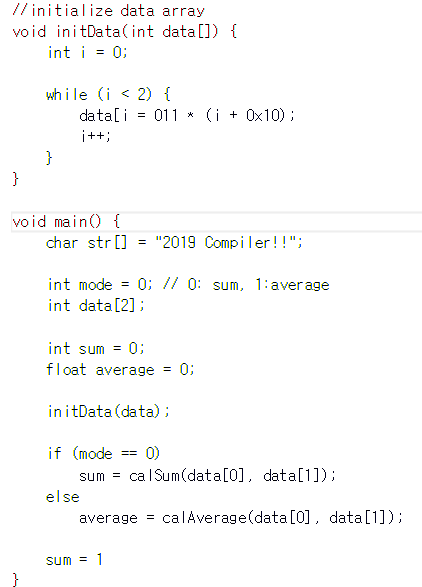
4.Data file given from professor

4.1 term\_testdata\_err1.mc

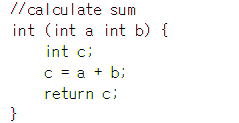


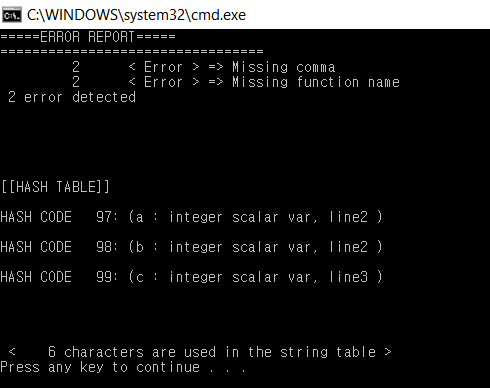


4.2 term\_testdata\_err2.mc

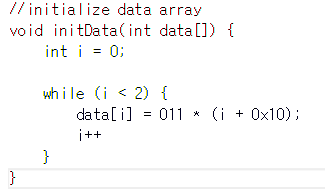


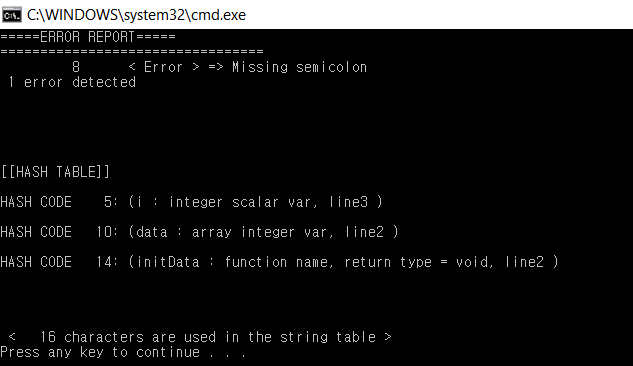
4.3 term\_testdata\_err3.mc



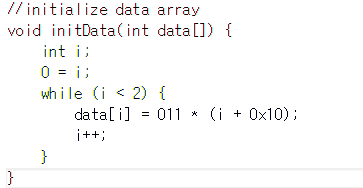


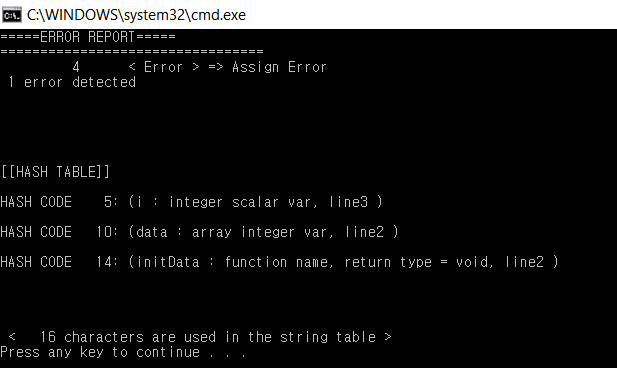
4.4 term\_testdata\_err4.mc





4.5 term\_testdata\_err5.mc





4.6 term\_testdata\_noerr.mc

