2021 - 1

Compiler Homework2 Report

Team 01

1615015 박기은

1617029 이혜인

1871056 한지수

Contents

1. Code -- 3
2. Test data --- 15
3. No Error Testdata --- 25
4. With Error Testdata -- 29
5. Contribution -- 35
6. Additional Functions -- 35

1. Code

glob.h

```
/* glob.h - Hash table 구성에 필요한 global 변수들 설정
* programmer - 박기은, 이혜인, 한지수
* date - 27/04/2021
*/

#pragma once
#define STsize 1000

int line;
char ST[STsize];
int nid = 0;   //현재 identifier 위치 index
int nfree = 0; //ST에서 다음으로 빈 index
int sameid = 0;//identifier의 첫번째 index
int flag = 0;  //0이면 중복 없음, 1이면 중복 있음
int errcnt;
```

tn.h

```
/* tn.h - tokentypes(token 정보), errorTypes(에러 정보) 정의
* programmer - 박기은, 이혜인, 한지수
* date - 27/04/2021
*/

enum tokentypes
{
        TEOF,
        TCONST,
        TELSE,
        TIF,
        TINT,
        TRETURN,
        TVOID,
        TWHILE,
        TADD,
        TSUB,
        TMUL,
        TDIV,
        TMOD,
        TASSIGN,
        TADDASSIGN,
          TSUBASSIGN,
          TMULASSIGN,
          TDIVASSIGN,
          TMODASSIGN,
          TNOT,
          TAND,
          TOR,
          TEQUAL,
          TNOTEQU,
          TLESS,
          TGREAT,
          TLESSE,
          TGREATE,
          TINC,
          TDEC,
          TLSBRACKET,
          TRSBRACKET,
          TLMBRACKET,
          TRMBRACKET,
          TLLBRACKET,
          TRLBRACKET,
          TSQUOTE,
          TBQUOTE,
          TIDENT,
          TNUMBER,
          TFLOAT,
          TSEPARATOR,
          TLINE,
          TERROR,
          TILLID,

          TBRACKET,
          TCOMMA,
          TSEMICOLON,

          TCOMMENT_SINGLE,
          TCOMMENT_MULTI

};

enum errorTypes {
          noerror,
          illsp,
          illid,
          overst,
        overfl,
        illch
};
```

scanner.l

```
%{
        /* scanner.l - lexical analyzer for the MiniC
        * programmer - 박기은, 이혜인, 한지수
        * date - 27/04/2021
        */
#include <stdio.h>
#include <stdlib.h>
#include "tn.h" /* token name definition */
#include "glob.h" /*global variable */
%}

letter [A-Za-z_]
digit [0-9]

%%

"const"                         return(TCONST);
"else"                  return(TELSE);
"if"                    return(TIF);
"int"                   return(TINT);
"return"                        return(TRETURN);
"void"                  return(TVOID);
"while"                 return(TWHILE);

"//".*                  return(TCOMMENT_SINGLE);
"/*"([^*]|"*"+[^*/])*"*"+"/"            return(TCOMMENT_MULTI);

"+"                     return(TADD);
"-"                     return(TSUB);
"*"                     return(TMUL);
"/"                     return(TDIV);
"%"                     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);

"\""                        return(TBQUOTE);
"\'"                        return(TSQUOTE);

"("                         return(TLSBRACKET);
")"                         return(TRSBRACKET);
"{"                         return(TLMBRACKET);
"}"                         return(TRMBRACKET);
"["                         return(TLLBRACKET);
"]"                         return(TRLBRACKET);

","                         return(TCOMMA);
";"                         return(TSEMICOLON);

{letter}({letter}|{digit})*             return(TIDENT);
[0-9][0-9]*                 return(TNUMBER);
[0-9]+"."[0-9]+(e[+-]?[0-9]+)?      return(TFLOAT);


[ \t]                       return(TSEPARATOR);
[\n]                        return(TLINE);

{digit}+{letter}({letter}|{digit})*     return(TILLID);
.                     return(TERROR);

%%

int yywrap()
{
        printf("\nEnd\n");
        return 1;
}
```

main.c

```
/* symtable.c - 각 token에 대한 출력
* programmer - 박기은, 이혜인, 한지수
* date - 27/04/2021
*/

#include <stdio.h>
#include <stdlib.h>
#include "tn.h"
#include "glob.h"
extern void PrintHStable();
extern void SkipSeparators();
extern yylex();
extern char* yytext;

void main()
{
        enum tokentypes tn;  // token number
        enum errorTypes err;
        line = 1;
        printf("Start\n\n");
        printf("Line number\tToken type\tST-index\tToken\n");
        while ((tn = yylex()) != TEOF) {
                printtoken(tn);
        }
        if (errcnt == 0) printf("No errors detected");
        else printf("%d errors detected\n", errcnt);
        printf("\n\n");
        PrintHStable();
}
```

printtoken.c

```
/* printtoken.c - Classify functions of classified token cases
* programmer - 박기은, 이혜인, 한지수
* date - 27/04/2021
*/
#include <stdio.h>
#include <stdlib.h>
#include "tn.h"
#include "glob.h"
extern void reporterror(char* string);
extern void countline(char* string);
extern yylex();
extern char* yytext;

void printtoken(enum tokentypes tn) {

	if (tn == TSEPARATOR) (void)0;
	else if (tn == TLINE) {
		line++;
		return;
	}
	else {
		printf("%11d\t", line);
		switch (tn) {


		case TCONST: printf("Constant\t"); break;
		case TELSE: printf("Else\t"); break;
		case TIF: printf("If\t"); break;
		case TINT: printf("Integer\t"); break;
		case TRETURN: printf("Return\t"); break;
		case TVOID: printf("Void\t"); break;
		case TWHILE: printf("While\t"); break;

		case TADD: printf("Add\t"); break;
		case TSUB: printf("Subract\t"); break;
		case TMUL: printf("Multiply\t"); break;
		case TDIV: printf("Divide\t"); break;
		case TMOD: printf("Mod\t"); break;

		case TASSIGN: printf("Assign\t"); break;
		case TADDASSIGN: printf("Add and assign\t"); break;
		case TSUBASSIGN: printf("Subtract and assign\t"); break;
		case TMULASSIGN: printf("Multiply and assign\t"); break;
		case TDIVASSIGN: printf("Divide and assign\t"); break;
		case TMODASSIGN: printf("Mod and assign\t"); break;
```

```
                    case TNOT: printf("Not\t"); break;
                    case TAND: printf("And\t"); break;
                    case TOR: printf("Or\t"); break;

                    case TEQUAL: printf("Equal\t"); break;
                    case TNOTEQU: printf("Not_Equal\t"); break;
                    case TLESS: printf("Less\t"); break;
                    case TGREAT: printf("Great\t"); break;
                    case TLESSE: printf("Less equal\t"); break;
                    case TGREATE: printf("Great equal\t"); break;

                    case TINC: printf("Increase\t"); break;
                    case TDEC: printf("Decrease\t"); break;

                    case TLSBRACKET: printf("Left Small Bracket\t"); break;
                    case TRSBRACKET: printf("Right Small Bracket\t"); break;
                    case TLMBRACKET: printf("Left Medium Bracket\t"); break;
                    case TRMBRACKET: printf("Right Medium Bracket\t"); break;
                    case TLLBRACKET: printf("Left Large Bracket\t"); break;
                    case TRLBRACKET: printf("Right Large Bracket\t"); break;
                    case TCOMMA: printf("Comma\t"); break;
                    case TSEMICOLON: printf("Semicolon\t"); break;

                    case TSQUOTE: printf("Small Quote"); break;
                    case TBQUOTE: printf("Big Quote"); break;

                    case TIDENT: reporterror(yytext); break; //인식한 identifier의 오류를 점검
                    case TNUMBER: printf("Number: %d\t", atoi(yytext)); break;
                    case TFLOAT: printf("Float: %f\t", atof(yytext)); break;

                    case TERROR: printerror(illsp, yytext); break; //illegal symbol일 때 출력
                    case TILLID: printerror(illid, yytext); break; //숫자로 시작하는 identifier일 때
오류 출력

                    case TCOMMENT_SINGLE: printf("Comment line\t");  break;
                    case TCOMMENT_MULTI: printf("Comment line\t");  countline(yytext);
break; //여러 줄 주석의 라인 개수 출력
                    }
                    printf("\n");
          }
}
```

symtable.c

```
/* symtable.c - 각 identifier의 Hash Table 구성
* programmer - 박기은, 이혜인, 한지수
* date - 27/04/2021
*/

#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "tn.h"
#include "glob.h"
#define STsize 1000
#define HTsize 100
#define Isize 25
#define isWord(x) ((((x)>='a'&&(x)<='z') || ((x)>='A'&&(x)<='Z')) || (x=='_') )
#define isNum(x) ((x) >= '0' && (x) <= '9')
#define idlen 12

typedef enum errorTypes ERRORtypes;
ERRORtypes errr;
typedef struct HTentry* HTpointer;
typedef struct HTentry
{
	int index;
	HTpointer next;
} HTentry;

char separators[] = " \t\r\n";
HTpointer HT[HTsize];
char ST[STsize];
int nid = 0;
int nfree = 0;
int sameid = 0;
int flag = 0;
int EOFflag = 1;
int initalize_done = 0;

char input;
char string[Isize];
int hashcode;
extern void reporterror(char* string);

//ReadIO 함수: string 읽어서 ST에 넣기(overfl 에러 체크)
void ReadID(char* string)
{
	nid = nfree;
```

```
        for (int i = 0; string[i] != '\0'; i++)
        {
                if (nfree >= STsize) //STSize overflow
                {
                        errr = overfl;
                        printerror(errr, string);
                        break;
                }
                else {
                        ST[nfree++] = string[i];
                }
        }
}

//ComputeHS 함수: ST에 존재하는 [nid~(nfree-2)]까지의 character를 이용한 해시함수 구현
//          H(x) = (f(x) mod m)+1
void ComputeHS(int nid, int nfree)
{
        int tot_ascii = 0;
        for (int i = nid; i < nfree - 1; i++)
                tot_ascii += (int)ST[i];
        hashcode = tot_ascii % HTsize;
}

// LookupHS: identifier의 해시 결과 중복 발생 여부에 따라 flag 값 조정
void LookupHS(int nid, int hscode)
{
        HTpointer temp;
        int a, b;
        flag = 0;

        if (HT[hscode] != NULL) {
                temp = HT[hscode];
                while (temp != NULL && flag == 0) {
                        flag = 1;
                        a = temp->index;
                        b = nid;
                        sameid = a;

                        while (ST[a] != '\0' && flag == 1) {
                                //중복 발생
                                if (ST[a] == ST[b]) {
                                        a++;
                                        b++;
                                }
```

```
                                        //중복되지 않을 경우
                                        else flag = 0;
                              }
                              temp = temp->next;
                    }
          }
}

// ADDHT 함수: HTpointer 할당 받아서 HS의 hscode에 identifier 삽입
void ADDHT(int hscode)
{
          HTpointer pt;
          pt = (HTpointer)malloc(sizeof(pt));
          pt->index = nid;
          pt->next = HT[hscode];
          HT[hscode] = pt;
}

// symtable 함수: Hash Table 전체 시스템 구성 (ReadID -> ComputerHS -> LookupHS ->
ADDDHT)
void symtable(char* string) {
          int i;
          ReadID(string);
          ST[nfree++] = '\0';
          ComputeHS(nid, nfree);
          LookupHS(nid, hashcode);

          if (!flag) {
                    printf("%-11d\t", nid);
                    i = nid;
                    while (i < nfree - 1) printf("%c", ST[i++]);
                    ADDHT(hashcode);
          }
          else {
                    printf("%-11d\t", sameid);
                    i = nid;
                    while (i < nfree - 1) printf("%c", ST[i++]);
                    nfree = nid;
          }
}

// PrintHStable 함수: HashTable 결괏값 출력 - hashcode, identifier list와 전체 character
출력
void PrintHStable()
{
	printf("[[ HASH TABLE ]]\n\n");

	int i = 0;
	while (i < HTsize) {
		if (HT[i] != NULL) {
			printf("Hash code %-3d: ", i);
			HTpointer pt = HT[i];
			while (pt != NULL) {

				int st_index = pt->index;
				while (ST[st_index] != '\0' && st_index < STsize) printf("%c",
ST[st_index++]);
				printf(" ");
				pt = pt->next;
			}
			printf("\n");
		}
		i++;
	}
	printf("\n<%d characters are used in the string table>\n", nfree);
}
```

reporterror.c

```
/* reporterror.c - Check identifier error
* programmer - 박기은, 이혜인, 한지수
* date - 27/04/2021
*/

#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "tn.h"
#include "glob.h"
#define isWord(x) ((((x)>='a'&&(x)<='z') || ((x)>='A'&&(x)<='Z')) || (x=='_')))
#define isNum(x) ((x) >= '0' && (x) <= '9')
extern void symtable(char* string);
typedef enum errorTypes ERROR;
ERROR errr = noerror;

int errflag = 0;
errcnt = 0;

//PrintError 함수: ERRORtypes 객체 내용을 바탕으로 ERROR 정보를 출력
```

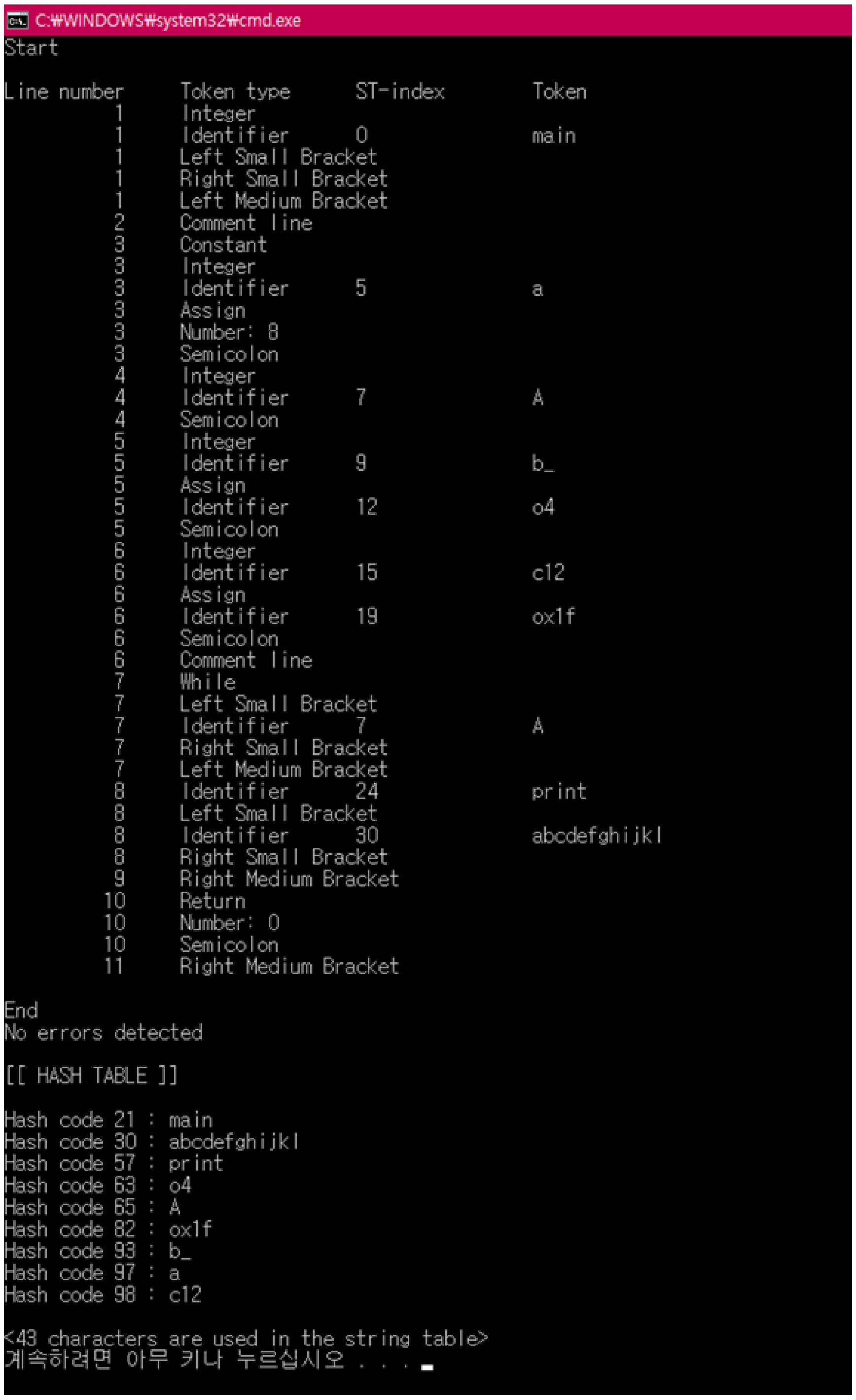
```
// error: illsp(허용되지 않는 문자 사용한 식별자 에러)
//      illid(숫자와 함께 시작하는 식별자 에러)
//      overst(식별자길이idlen을 넘는 식별자 에러)
//      overfl(크기로 overflow 발생)
void printerror(ERROR err, char* string)
{
        if (err == illid) {
                printf("**Error**\t\t%s Illegal IDENT", string);
                errcnt++;
        }
        else if (err == overst) {
                printf("**Error**\t\t\t%s over 12 words", string);
                errcnt++;
        }
        else if (err == illsp) {
                printf("**Error**\t\t%s Illegal Symbol", string);
                errcnt++;
        }

        else if (err == overfl) {
                printf("**Error**\t overflow occured\n");
                errcnt++;
        }
}


//reporterror 함수: overst(string 길이가 12자 이상) 에러 감지
void reporterror(char* string) {
        if (strlen(string) > 12) {
                errr = overst;
                printerror(errr, string);
        }

        else {
                printf("Identifier\t");
                symtable(string);
        }
}
```

countline.c

```
/* countline.c - \n, \r 에 따라 라인수 업로드 하는 함수
* programmer - 박기은, 이혜인, 한지수
* date - 27/04/2021
*/

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

// countline 함수:  여러줄 주석 /* */의  라인의 개수를 세기 위한 함수. /* */ 부분의 토큰에서
개행문자 '\n' 를 센다.
void countline(char* string)
{

	int i = 0;

	while (string[i] != '\0') {
		if (string[i] == '\n' || string[i] == '\r') line++;
		i++;
	}

}
```

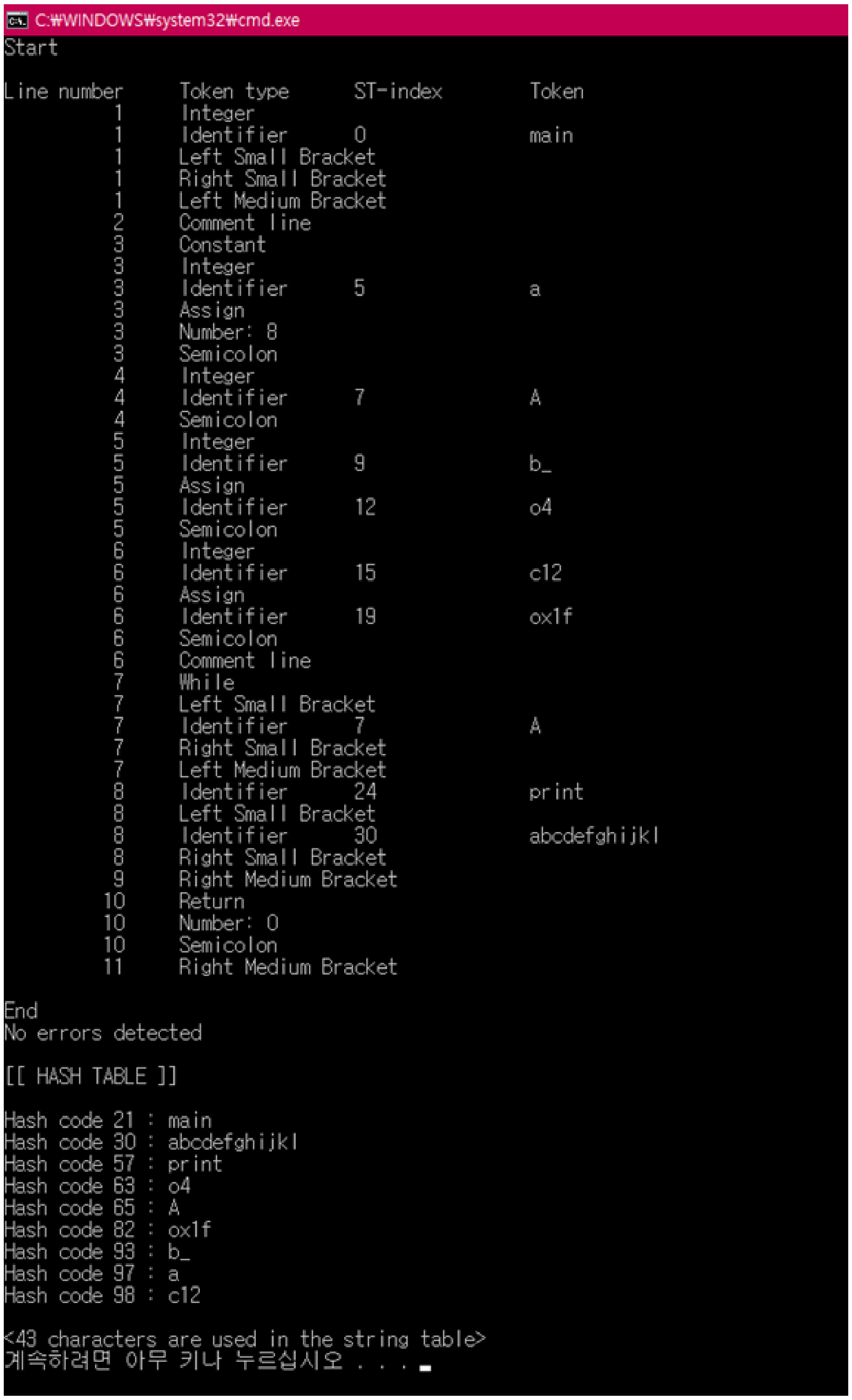
2. Test data

<testdata1.dat>



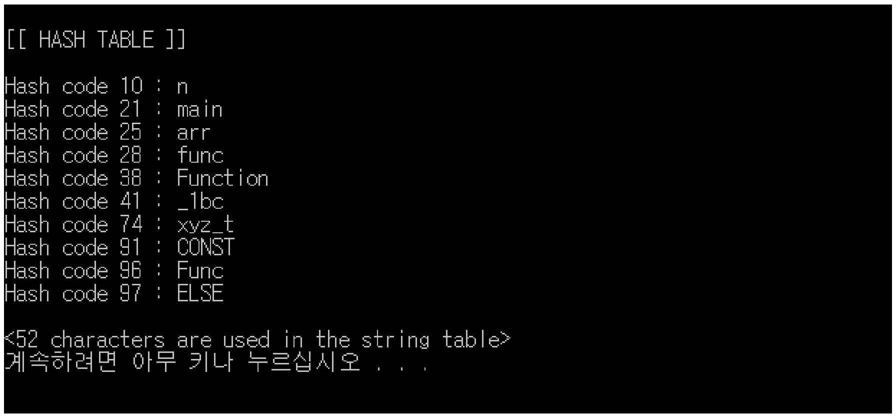
```
C:₩WINDOWS₩system32₩cmd.exe
Start

Line number      Token type      ST-index        Token
          1      Integer
          1      Identifier      0               main
          1      Left Small Bracket
          1      Right Small Bracket
          1      Left Medium Bracket
          2      Comment line
          3      Constant
          3      Integer
          3      Identifier      5               a
          3      Assign
          3      Number: 8
          3      Semicolon
          4      Integer
          4      Identifier      7               A
          4      Semicolon
          5      Integer
          5      Identifier      9               b_
          5      Assign
          5      Identifier      12              o4
          5      Semicolon
          6      Integer
          6      Identifier      15              c12
          6      Assign
          6      Identifier      19              ox1f
          6      Semicolon
          6      Comment line
          7      While
          7      Left Small Bracket
          7      Identifier      7               A
          7      Right Small Bracket
          7      Left Medium Bracket
          8      Identifier      24              print
          8      Left Small Bracket
          8      Identifier      30              abcdefghijkl
          8      Right Small Bracket
          9      Right Medium Bracket
         10      Return
         10      Number: 0
         10      Semicolon
         11      Right Medium Bracket

End
No errors detected

[[ HASH TABLE ]]

Hash code 21 : main
Hash code 30 : abcdefghijkl
Hash code 57 : print
Hash code 63 : o4
Hash code 65 : A
Hash code 82 : ox1f
Hash code 93 : b_
Hash code 97 : a
Hash code 98 : c12

<43 characters are used in the string table>
계속하려면 아무 키나 누르십시오 . . .
```

<testdata2.dat>

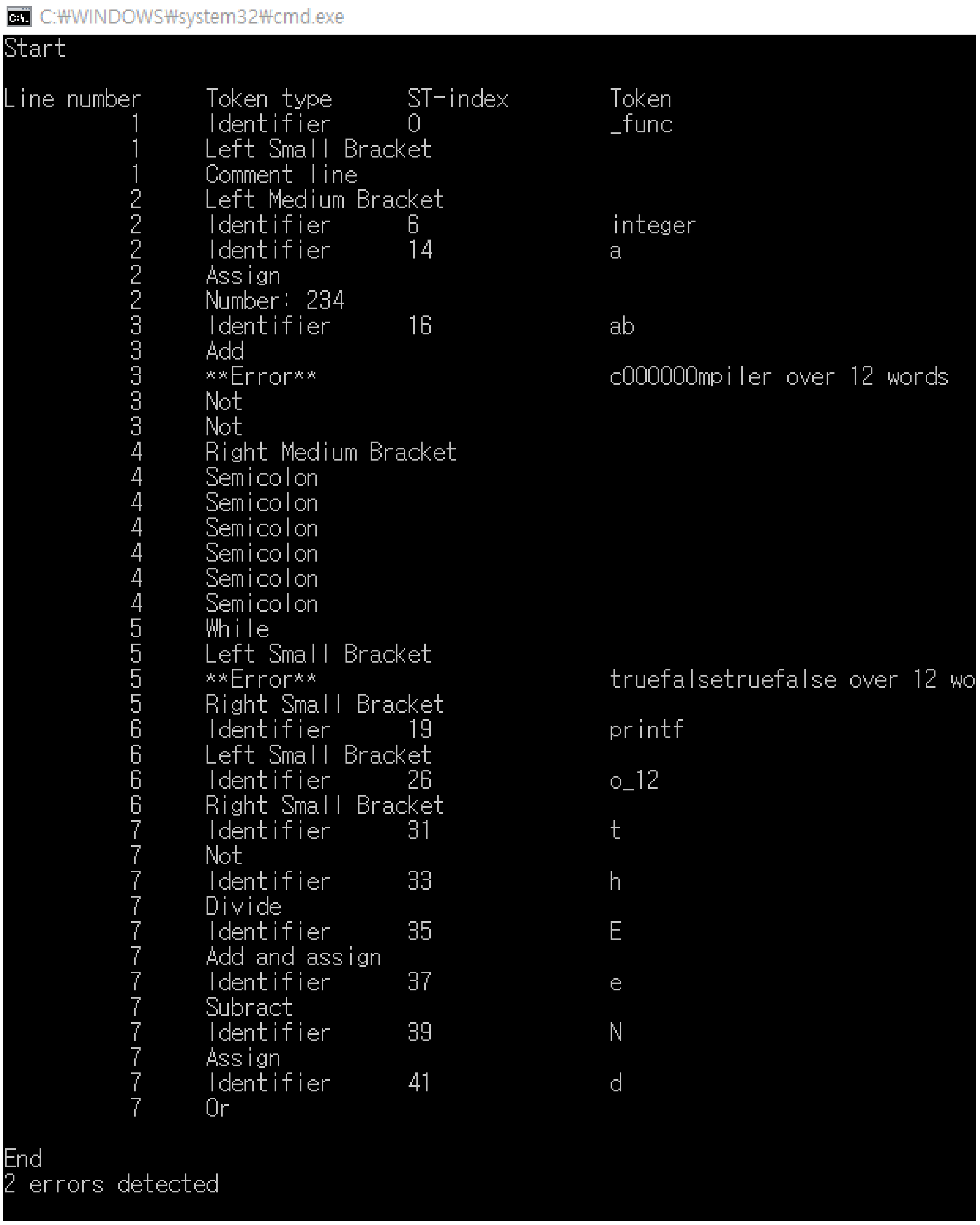


C:\WINDOWS\system32\cmd.exe

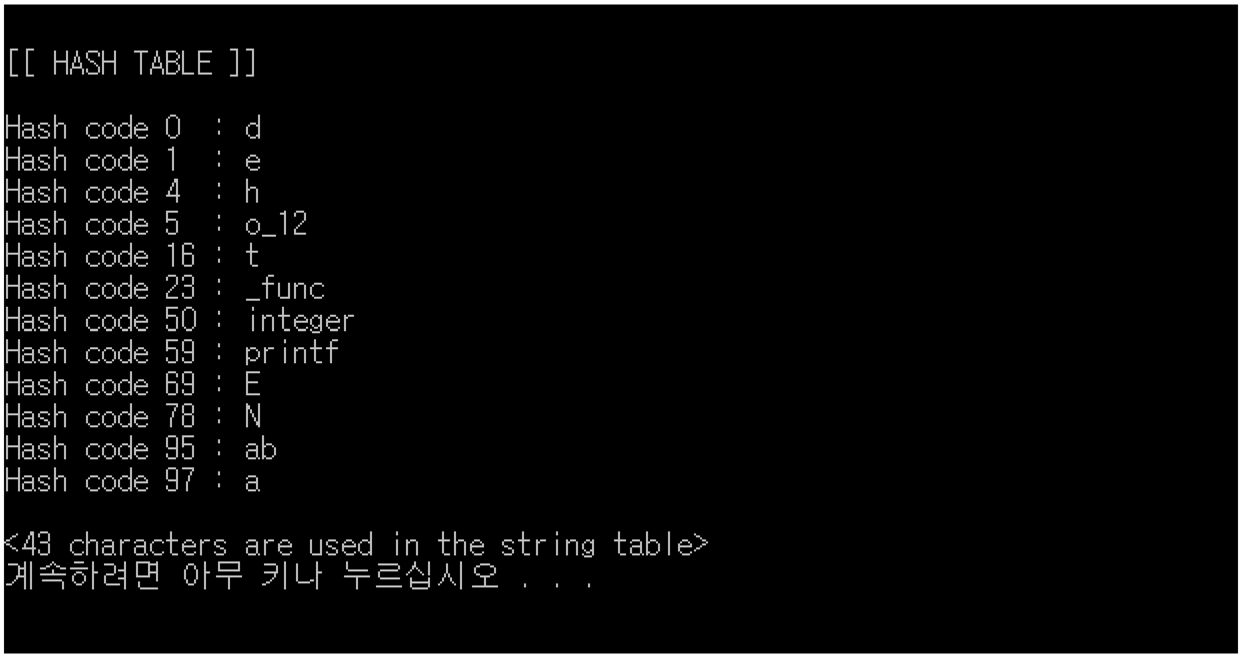
```
Start

Line number     Token type      ST-index        Token
          1     Integer
          1     Identifier      0               main
          1     Left Small Bracket
          1     Right Small Bracket
          1     Left Medium Bracket
          2     Comment line
          3     Constant
          3     Integer
          3     Identifier      5               a
          3     Assign
          3     Number: 8
          3     Semicolon
          4     Integer
          4     Identifier      7               A
          4     Semicolon
          5     Integer
          5     Identifier      9               b_
          5     Assign
          5     Identifier      12              o4
          5     Semicolon
          6     Integer
          6     Identifier      15              c12
          6     Assign
          6     Identifier      19              ox1f
          6     Semicolon
          6     Comment line
          7     While
          7     Left Small Bracket
          7     Identifier      7               A
          7     Right Small Bracket
          7     Left Medium Bracket
          8     Identifier      24              print
          8     Left Small Bracket
          8     Identifier      30              abcdefghijkl
          8     Right Small Bracket
          9     Right Medium Bracket
         10     Return
         10     Number: 0
         10     Semicolon
         11     Right Medium Bracket

End
No errors detected

[[ HASH TABLE ]]

Hash code 21 : main
Hash code 30 : abcdefghijkl
Hash code 57 : print
Hash code 63 : o4
Hash code 65 : A
Hash code 82 : ox1f
Hash code 93 : b_
Hash code 97 : a
Hash code 98 : c12

<43 characters are used in the string table>
계속하려면 아무 키나 누르십시오 . . .
```


```
[[ HASH TABLE ]]

Hash code 10 : n
Hash code 21 : main
Hash code 25 : arr
Hash code 28 : func
Hash code 38 : Function
Hash code 41 : _1bc
Hash code 74 : xyz_t
Hash code 91 : CONST
Hash code 96 : Func
Hash code 97 : ELSE

<52 characters are used in the string table>
계속하려면 아무 키나 누르십시오 . . .
```

<testdata3.dat>

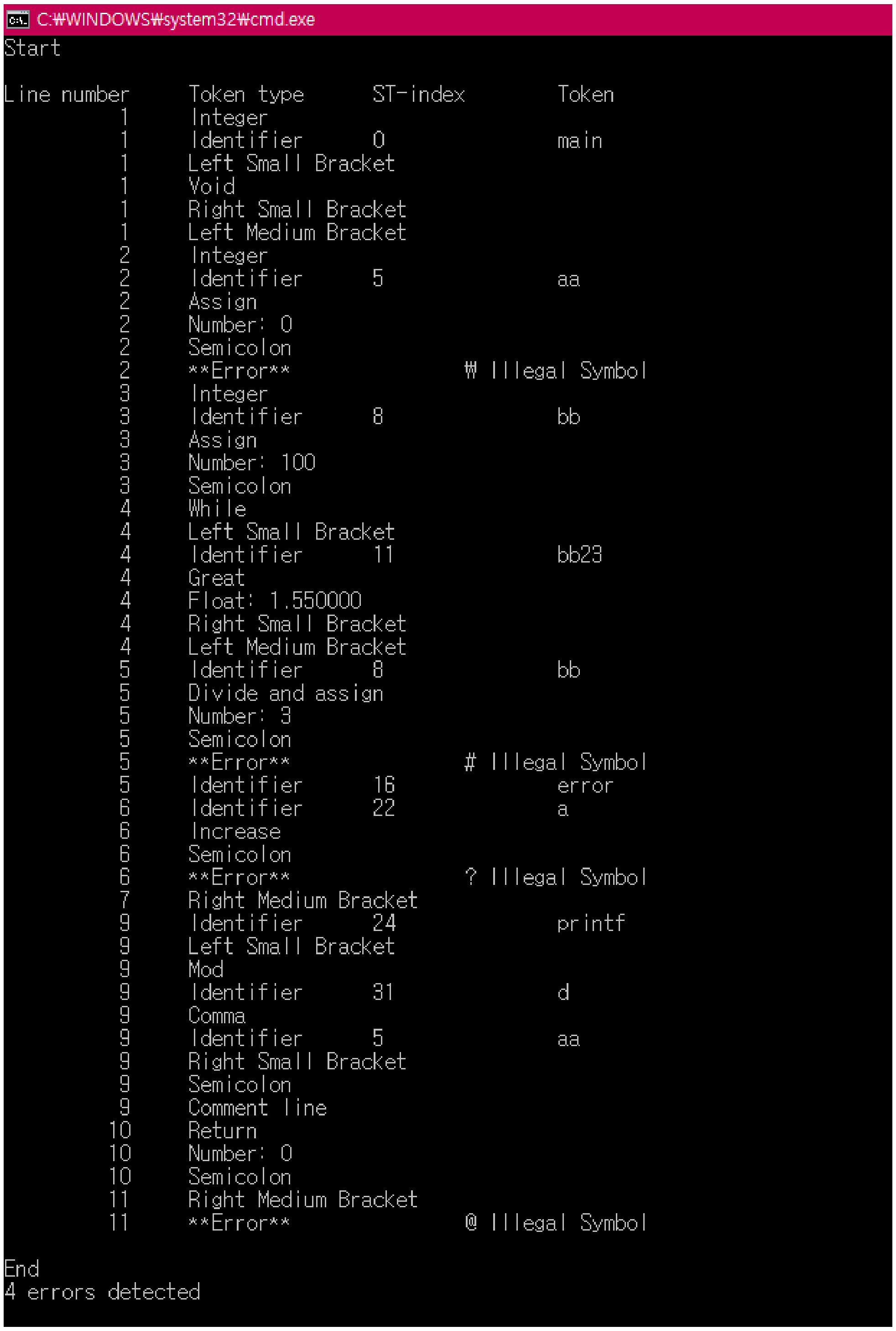


```
C:₩WINDOWS₩system32₩cmd.exe
Start

Line number     Token type      ST-index        Token
          1     Identifier      0               _func
          1     Left Small Bracket
          1     Comment line
          2     Left Medium Bracket
          2     Identifier      6               integer
          2     Identifier      14              a
          2     Assign
          2     Number: 234
          3     Identifier      16              ab
          3     Add
          3     **Error**                       c000000mpiler over 12 words
          3     Not
          3     Not
          4     Right Medium Bracket
          4     Semicolon
          4     Semicolon
          4     Semicolon
          4     Semicolon
          4     Semicolon
          4     Semicolon
          5     While
          5     Left Small Bracket
          5     **Error**                       truefalsetruefalse over 12 wo
          5     Right Small Bracket
          6     Identifier      19              printf
          6     Left Small Bracket
          6     Identifier      26              o_12
          6     Right Small Bracket
          7     Identifier      31              t
          7     Not
          7     Identifier      33              h
          7     Divide
          7     Identifier      35              E
          7     Add and assign
          7     Identifier      37              e
          7     Subract
          7     Identifier      39              N
          7     Assign
          7     Identifier      41              d
          7     Or

End
2 errors detected
```



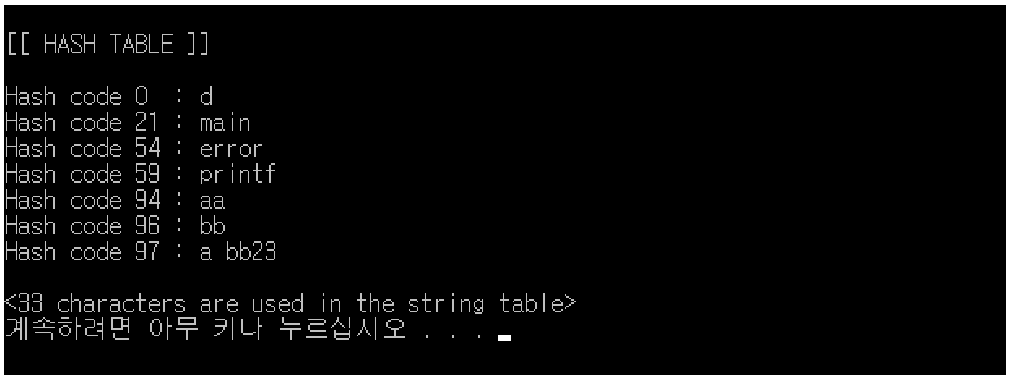
```
[[ HASH TABLE ]]

Hash code 0  : d
Hash code 1  : e
Hash code 4  : h
Hash code 5  : o_12
Hash code 16 : t
Hash code 23 : _func
Hash code 50 : integer
Hash code 59 : printf
Hash code 69 : E
Hash code 78 : N
Hash code 95 : ab
Hash code 97 : a

<43 characters are used in the string table>
계속하려면 아무 키나 누르십시오 . . .
```

<testdata4.dat>



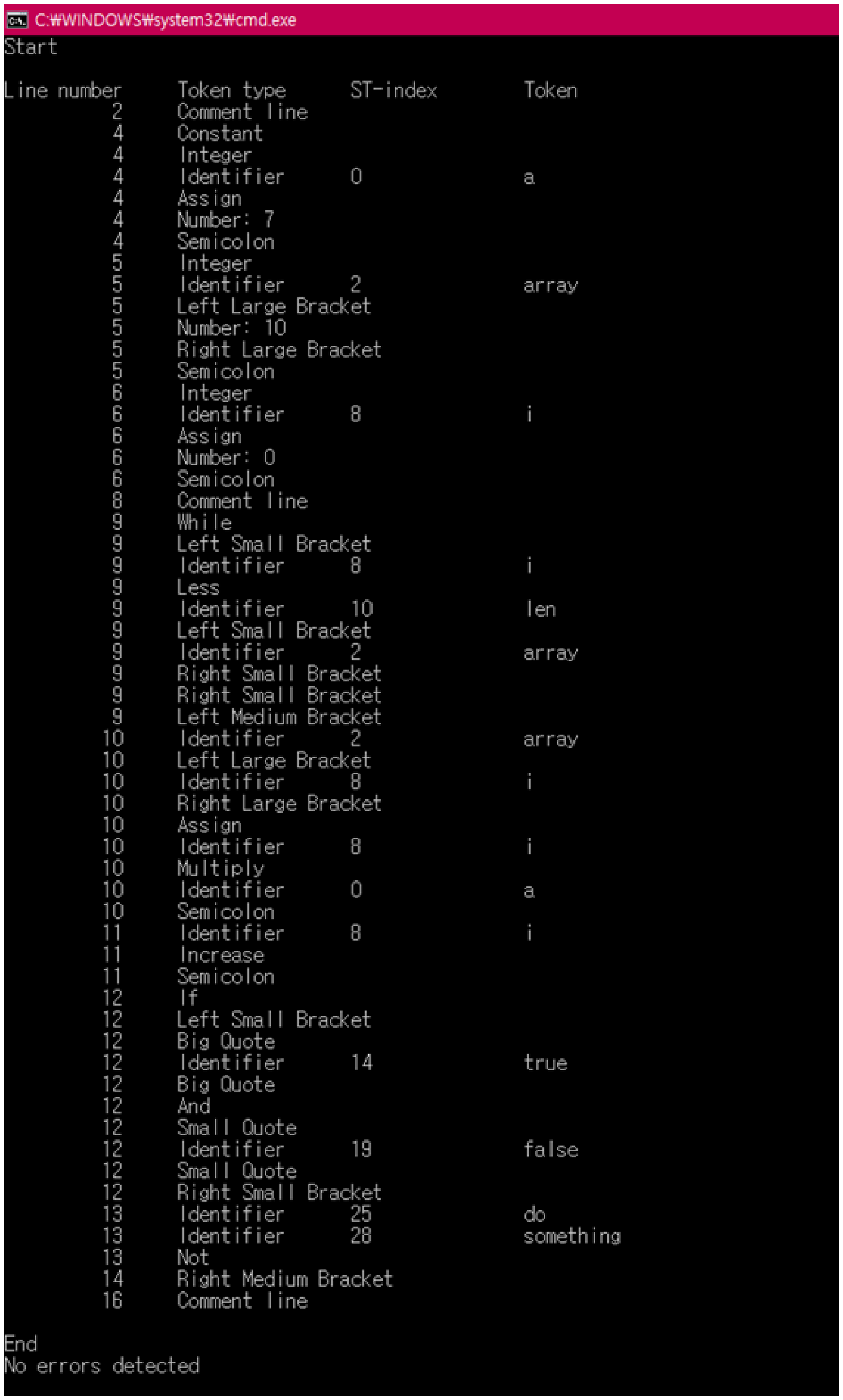
C:₩WINDOWS₩system32₩cmd.exe

```
Start

Line number      Token type       ST-index         Token
          1      Integer
          1      Identifier       0                main
          1      Left Small Bracket
          1      Void
          1      Right Small Bracket
          1      Left Medium Bracket
          2      Integer
          2      Identifier       5                aa
          2      Assign
          2      Number: 0
          2      Semicolon
          2      **Error**                ₩ Illegal Symbol
          3      Integer
          3      Identifier       8                bb
          3      Assign
          3      Number: 100
          3      Semicolon
          4      While
          4      Left Small Bracket
          4      Identifier       11               bb23
          4      Great
          4      Float: 1.550000
          4      Right Small Bracket
          4      Left Medium Bracket
          5      Identifier       8                bb
          5      Divide and assign
          5      Number: 3
          5      Semicolon
          5      **Error**                # Illegal Symbol
          5      Identifier       16               error
          6      Identifier       22               a
          6      Increase
          6      Semicolon
          6      **Error**                ? Illegal Symbol
          7      Right Medium Bracket
          9      Identifier       24               printf
          9      Left Small Bracket
          9      Mod
          9      Identifier       31               d
          9      Comma
          9      Identifier       5                aa
          9      Right Small Bracket
          9      Semicolon
          9      Comment line
         10      Return
         10      Number: 0
         10      Semicolon
         11      Right Medium Bracket
         11      **Error**                @ Illegal Symbol

End
4 errors detected
```



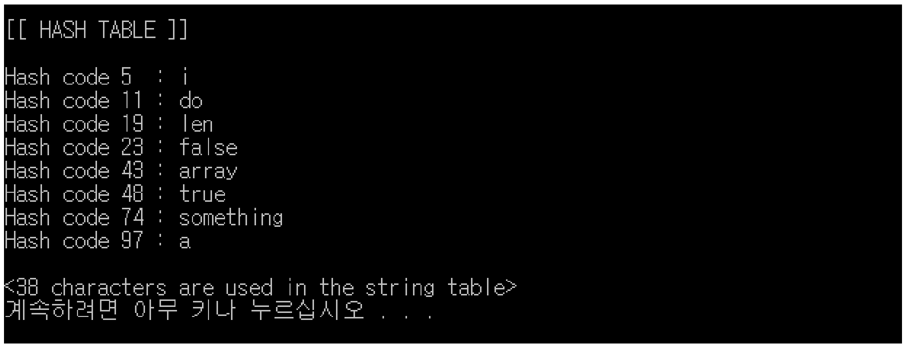
```
[[ HASH TABLE ]]

Hash code 0  : d
Hash code 21 : main
Hash code 54 : error
Hash code 59 : printf
Hash code 94 : aa
Hash code 96 : bb
Hash code 97 : a bb23

<33 characters are used in the string table>
계속하려면 아무 키나 누르십시오 . . .
```

<testdata5.dat>



C:₩WINDOWS₩system32₩cmd.exe

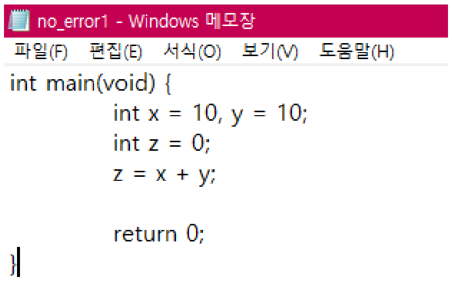
```
Start

Line number     Token type      ST-index        Token
          2     Comment line
          4     Constant
          4     Integer
          4     Identifier      0               a
          4     Assign
          4     Number: 7
          4     Semicolon
          5     Integer
          5     Identifier      2               array
          5     Left Large Bracket
          5     Number: 10
          5     Right Large Bracket
          5     Semicolon
          6     Integer
          6     Identifier      8               i
          6     Assign
          6     Number: 0
          6     Semicolon
          8     Comment line
          9     While
          9     Left Small Bracket
          9     Identifier      8               i
          9     Less
          9     Identifier      10              len
          9     Left Small Bracket
          9     Identifier      2               array
          9     Right Small Bracket
          9     Right Small Bracket
          9     Left Medium Bracket
         10     Identifier      2               array
         10     Left Large Bracket
         10     Identifier      8               i
         10     Right Large Bracket
         10     Assign
         10     Identifier      8               i
         10     Multiply
         10     Identifier      0               a
         10     Semicolon
         11     Identifier      8               i
         11     Increase
         11     Semicolon
         12     If
         12     Left Small Bracket
         12     Big Quote
         12     Identifier      14              true
         12     Big Quote
         12     And
         12     Small Quote
         12     Identifier      19              false
         12     Small Quote
         12     Right Small Bracket
         13     Identifier      25              do
         13     Identifier      28              something
         13     Not
         14     Right Medium Bracket
         16     Comment line

End
No errors detected
```



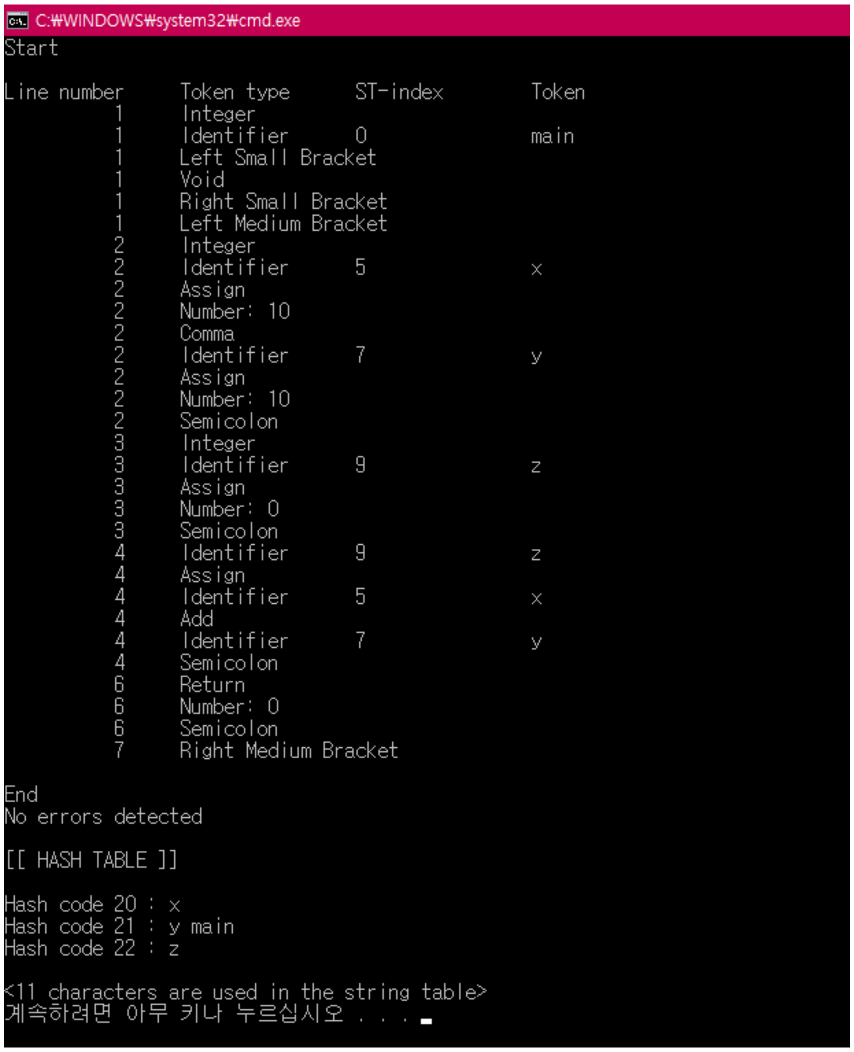
```
[[ HASH TABLE ]]

Hash code 5  : i
Hash code 11 : do
Hash code 19 : len
Hash code 23 : false
Hash code 43 : array
Hash code 48 : true
Hash code 74 : something
Hash code 97 : a

<38 characters are used in the string table>
계속하려면 아무 키나 누르십시오 . . .
```

3. No Error Test data

<no_error1.dat>

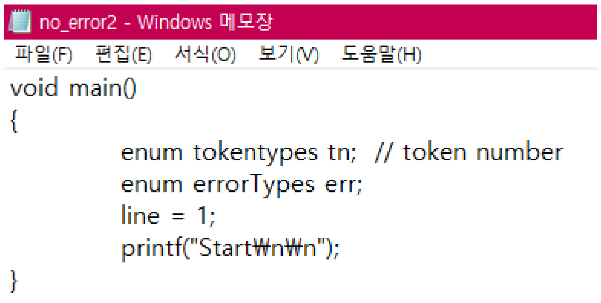

no_error1 - Windows 메모장
파일(F) 편집(E) 서식(O) 보기(V) 도움말(H)

```
int main(void) {
	int x = 10, y = 10;
	int z = 0;
	z = x + y;

	return 0;
}
```

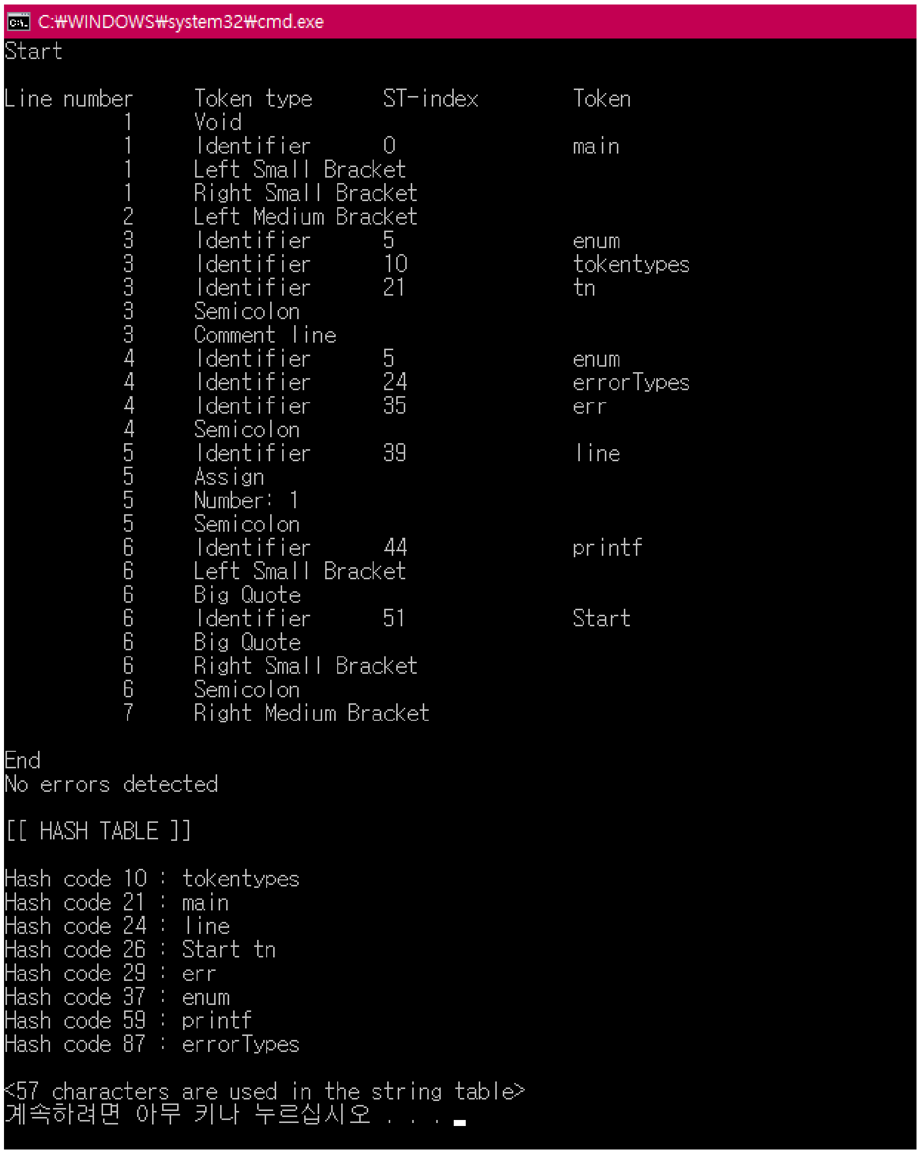


```
Start

Line number      Token type       ST-index        Token
          1      Integer
          1      Identifier       0               main
          1      Left Small Bracket
          1      Void
          1      Right Small Bracket
          1      Left Medium Bracket
          2      Integer
          2      Identifier       5               x
          2      Assign
          2      Number: 10
          2      Comma
          2      Identifier       7               y
          2      Assign
          2      Number: 10
          2      Semicolon
          3      Integer
          3      Identifier       9               z
          3      Assign
          3      Number: 0
          3      Semicolon
          4      Identifier       9               z
          4      Assign
          4      Identifier       5               x
          4      Add
          4      Identifier       7               y
          4      Semicolon
          6      Return
          6      Number: 0
          6      Semicolon
          7      Right Medium Bracket

End
No errors detected

[[ HASH TABLE ]]

Hash code 20 : x
Hash code 21 : y main
Hash code 22 : z

<11 characters are used in the string table>
계속하려면 아무 키나 누르십시오 . . .
```

<no_error2.dat>

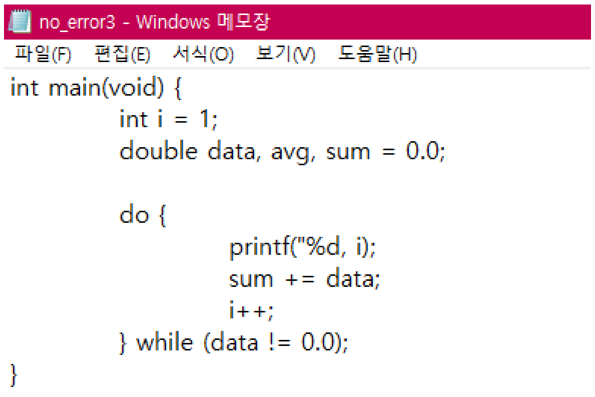

no_error2 - Windows 메모장
파일(F) 편집(E) 서식(O) 보기(V) 도움말(H)

```
void main()
{
            enum tokentypes tn;  // token number
            enum errorTypes err;
            line = 1;
            printf("Start₩n₩n");
}
```

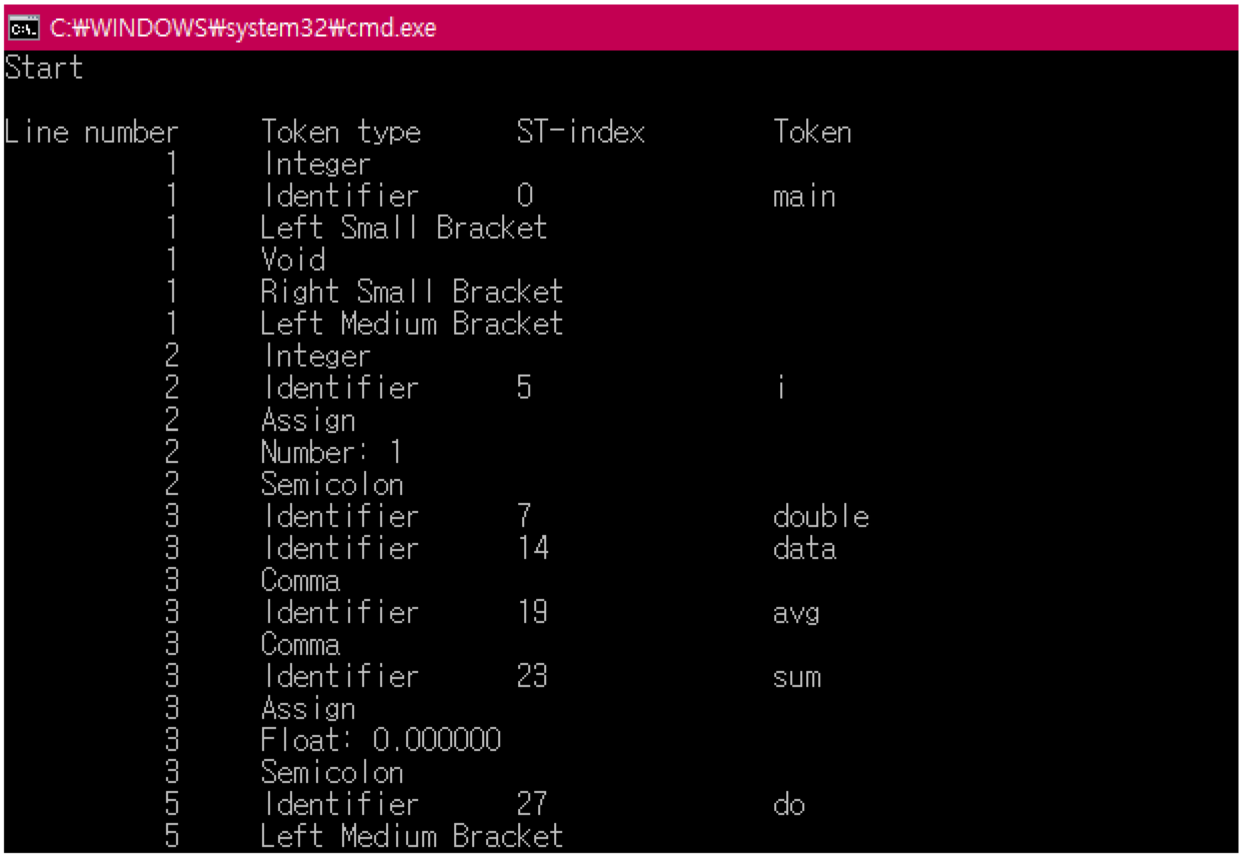


```
Start

Line number      Token type      ST-index        Token
          1      Void
          1      Identifier      0               main
          1      Left Small Bracket
          1      Right Small Bracket
          2      Left Medium Bracket
          3      Identifier      5               enum
          3      Identifier      10              tokentypes
          3      Identifier      21              tn
          3      Semicolon
          3      Comment line
          4      Identifier      5               enum
          4      Identifier      24              errorTypes
          4      Identifier      35              err
          4      Semicolon
          5      Identifier      39              line
          5      Assign
          5      Number: 1
          5      Semicolon
          6      Identifier      44              printf
          6      Left Small Bracket
          6      Big Quote
          6      Identifier      51              Start
          6      Big Quote
          6      Right Small Bracket
          6      Semicolon
          7      Right Medium Bracket

End
No errors detected

[[ HASH TABLE ]]

Hash code 10 : tokentypes
Hash code 21 : main
Hash code 24 : line
Hash code 26 : Start tn
Hash code 29 : err
Hash code 37 : enum
Hash code 59 : printf
Hash code 87 : errorTypes

<57 characters are used in the string table>
계속하려면 아무 키나 누르십시오 . . .
```

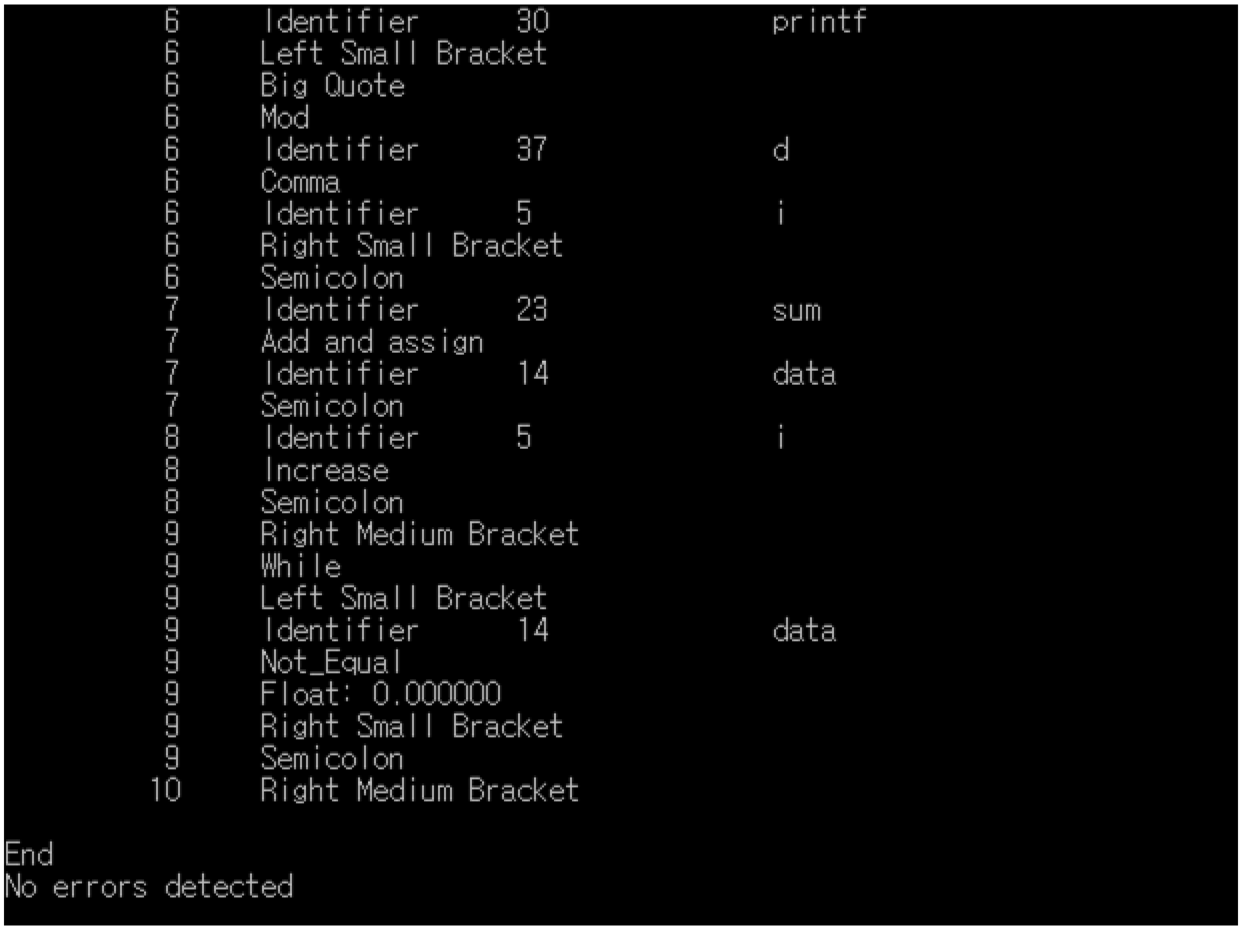
<no_error3.dat>


no_error3 - Windows 메모장
파일(F) 편집(E) 서식(O) 보기(V) 도움말(H)

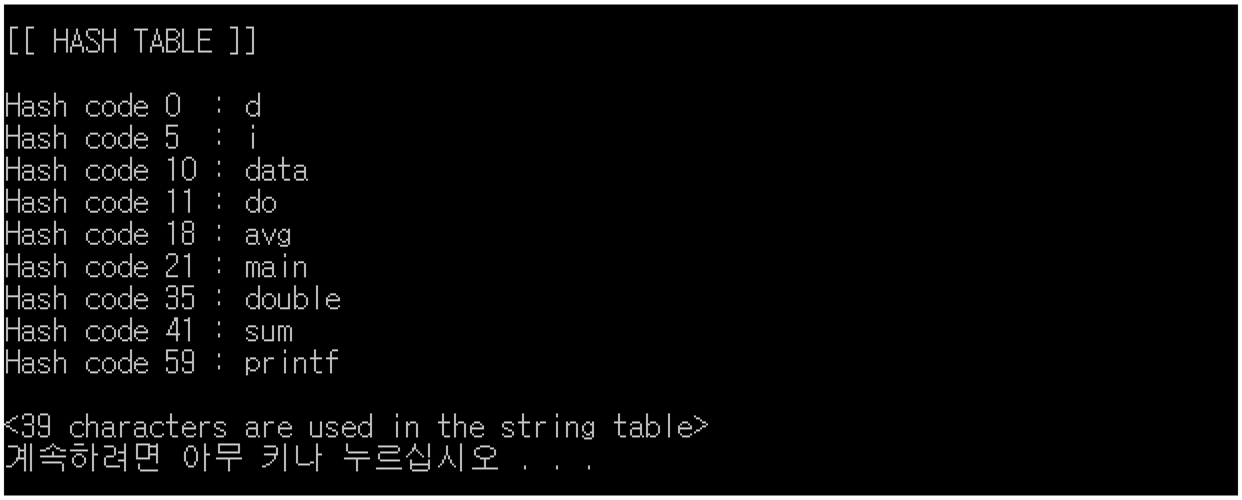
```
int main(void) {
	int i = 1;
	double data, avg, sum = 0.0;

	do {
		printf("%d, i);
		sum += data;
		i++;
	} while (data != 0.0);
}
```



```
Start

Line number      Token type       ST-index         Token
          1      Integer
          1      Identifier       0                main
          1      Left Small Bracket
          1      Void
          1      Right Small Bracket
          1      Left Medium Bracket
          2      Integer
          2      Identifier       5                i
          2      Assign
          2      Number: 1
          2      Semicolon
          3      Identifier       7                double
          3      Identifier       14               data
          3      Comma
          3      Identifier       19               avg
          3      Comma
          3      Identifier       23               sum
          3      Assign
          3      Float: 0.000000
          3      Semicolon
          5      Identifier       27               do
          5      Left Medium Bracket
```

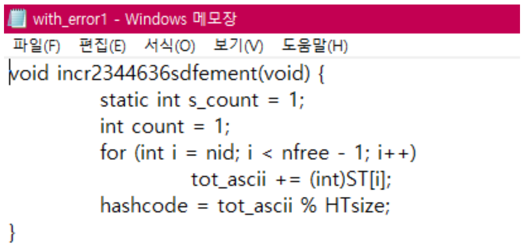


```
         6     Identifier      30              printf
         6     Left Small Bracket
         6     Big Quote
         6     Mod
         6     Identifier      37              d
         6     Comma
         6     Identifier      5               i
         6     Right Small Bracket
         6     Semicolon
         7     Identifier      23              sum
         7     Add and assign
         7     Identifier      14              data
         7     Semicolon
         8     Identifier      5               i
         8     Increase
         8     Semicolon
         9     Right Medium Bracket
         9     While
         9     Left Small Bracket
         9     Identifier      14              data
         9     Not_Equal
         9     Float: 0.000000
         9     Right Small Bracket
         9     Semicolon
        10     Right Medium Bracket

End
No errors detected
```

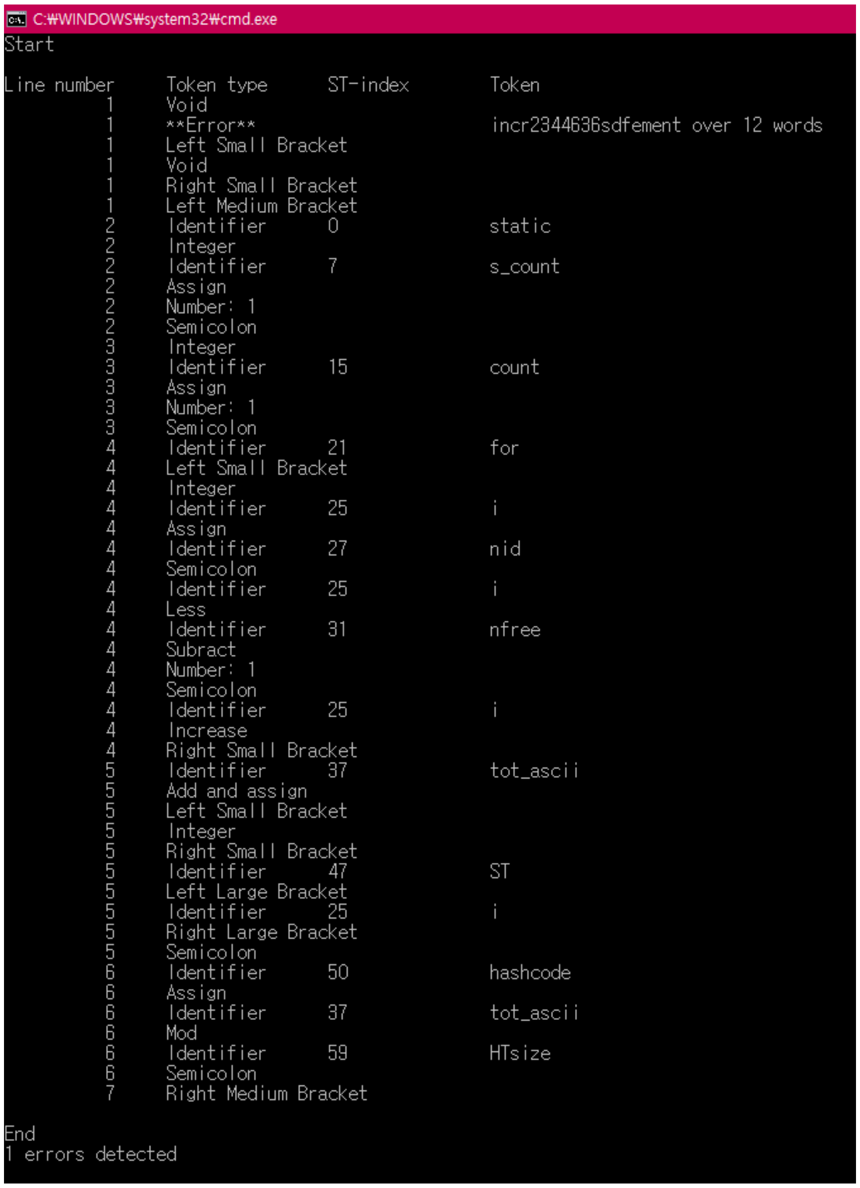


```
[[ HASH TABLE ]]

Hash code 0  : d
Hash code 5  : i
Hash code 10 : data
Hash code 11 : do
Hash code 18 : avg
Hash code 21 : main
Hash code 35 : double
Hash code 41 : sum
Hash code 59 : printf

<39 characters are used in the string table>
계속하려면 아무 키나 누르십시오 . . .
```

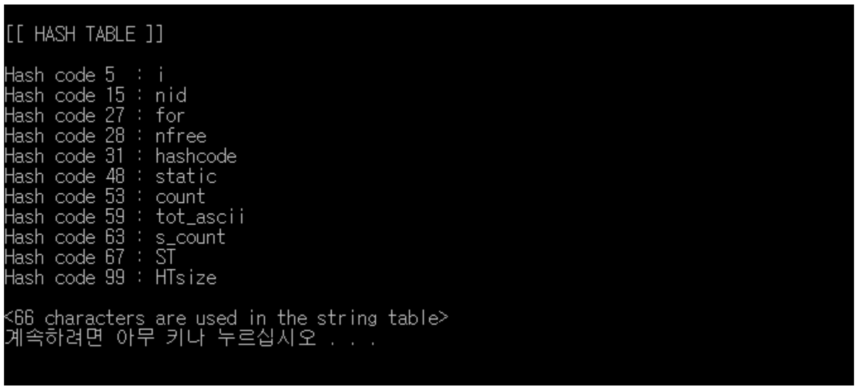
4. With Error Test data


with_error1 - Windows 메모장
파일(F) 편집(E) 서식(O) 보기(V) 도움말(H)

```
void incr2344636sdfement(void) {
		static int s_count = 1;
		int count = 1;
		for (int i = nid; i < nfree - 1; i++)
			tot_ascii += (int)ST[i];
		hashcode = tot_ascii % HTsize;
}
```

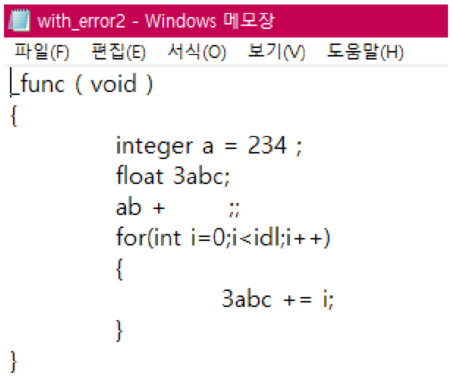


```
Start

Line number     Token type      ST-index        Token
          1     Void
          1     **Error**                       incr2344636sdfement over 12 words
          1     Left Small Bracket
          1     Void
          1     Right Small Bracket
          1     Left Medium Bracket
          2     Identifier      0               static
          2     Integer
          2     Identifier      7               s_count
          2     Assign
          2     Number: 1
          2     Semicolon
          3     Integer
          3     Identifier      15              count
          3     Assign
          3     Number: 1
          3     Semicolon
          4     Identifier      21              for
          4     Left Small Bracket
          4     Integer
          4     Identifier      25              i
          4     Assign
          4     Identifier      27              nid
          4     Semicolon
          4     Identifier      25              i
          4     Less
          4     Identifier      31              nfree
          4     Subract
          4     Number: 1
          4     Semicolon
          4     Identifier      25              i
          4     Increase
          4     Right Small Bracket
          5     Identifier      37              tot_ascii
          5     Add and assign
          5     Left Small Bracket
          5     Integer
          5     Right Small Bracket
          5     Identifier      47              ST
          5     Left Large Bracket
          5     Identifier      25              i
          5     Right Large Bracket
          5     Semicolon
          6     Identifier      50              hashcode
          6     Assign
          6     Identifier      37              tot_ascii
          6     Mod
          6     Identifier      59              HTsize
          6     Semicolon
          7     Right Medium Bracket

End
1 errors detected
```

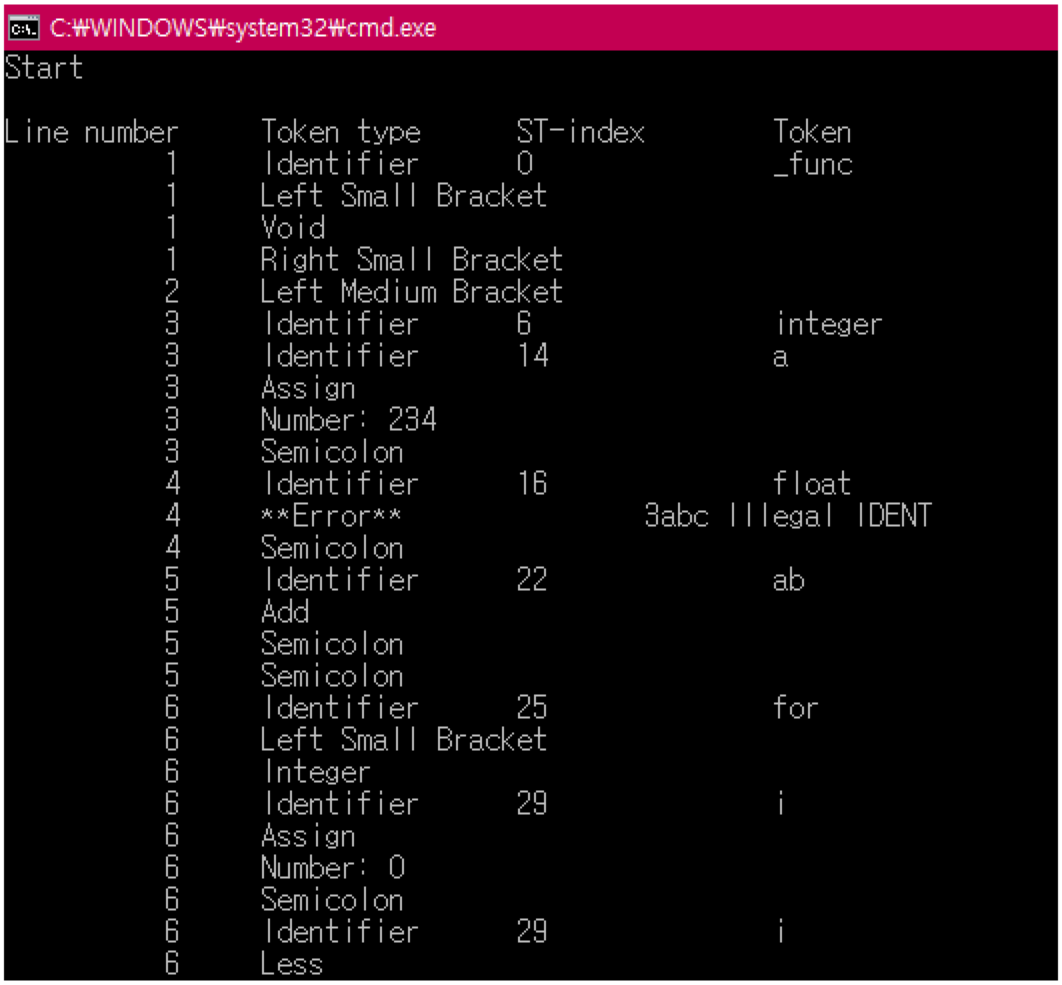


```
[[ HASH TABLE ]]

Hash code 5  : i
Hash code 15 : nid
Hash code 27 : for
Hash code 28 : nfree
Hash code 31 : hashcode
Hash code 48 : static
Hash code 53 : count
Hash code 59 : tot_ascii
Hash code 63 : s_count
Hash code 67 : ST
Hash code 99 : HTsize

<66 characters are used in the string table>
계속하려면 아무 키나 누르십시오 . . .
```

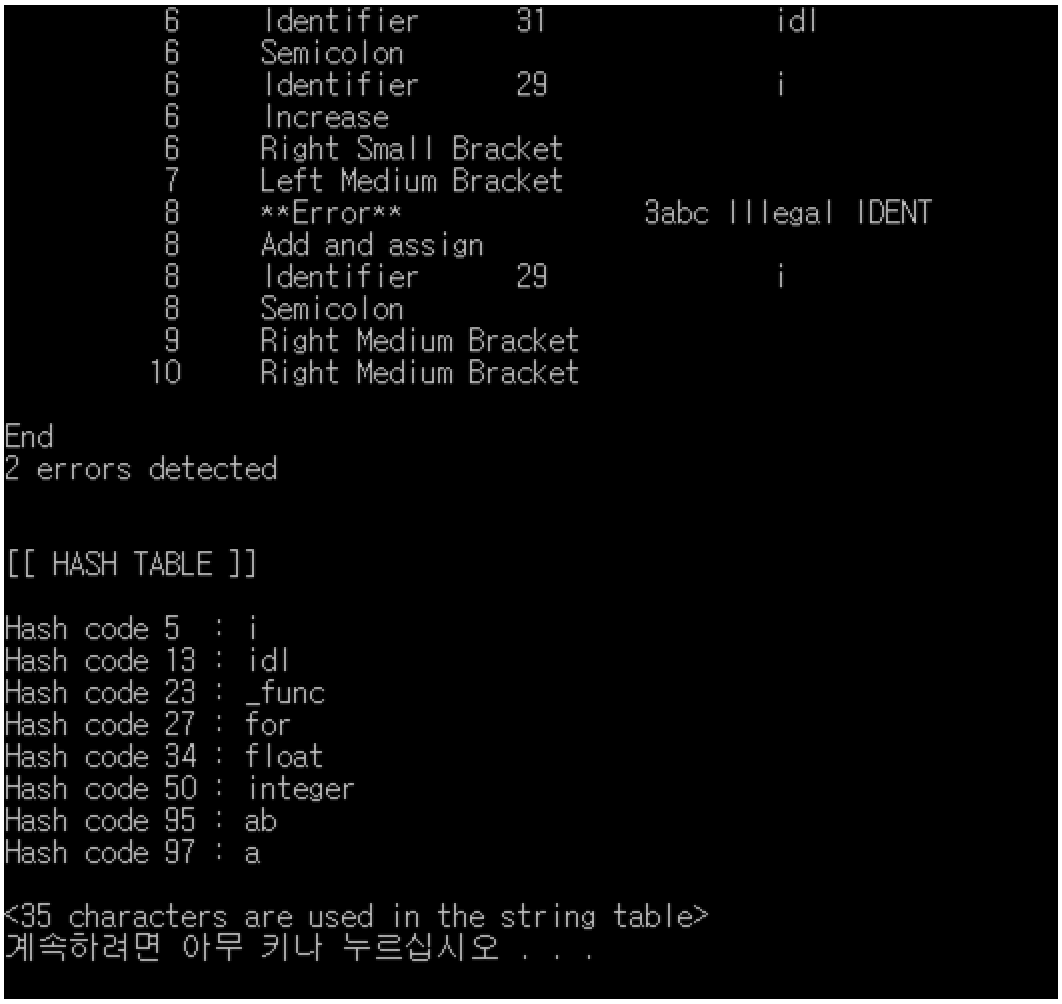
<with_error2.dat>


with_error2 - Windows 메모장
파일(F) 편집(E) 서식(O) 보기(V) 도움말(H)

```
_func ( void )
{
		integer a = 234 ;
		float 3abc;
		ab +	;;
		for(int i=0;i<idl;i++)
		{
			3abc += i;
		}
}
```

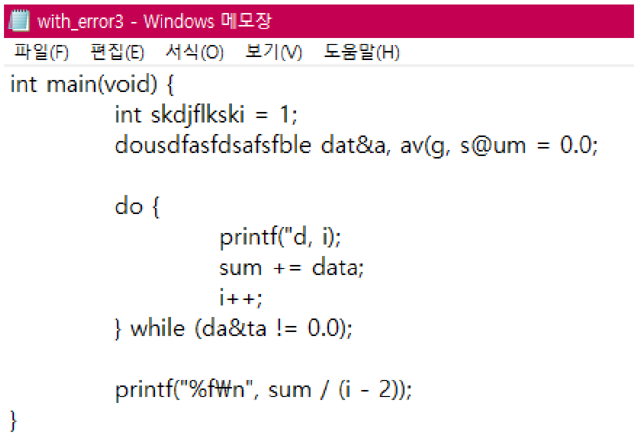


```
Start

Line number      Token type      ST-index        Token
          1      Identifier      0               _func
          1      Left Small Bracket
          1      Void
          1      Right Small Bracket
          2      Left Medium Bracket
          3      Identifier      6               integer
          3      Identifier      14              a
          3      Assign
          3      Number: 234
          3      Semicolon
          4      Identifier      16              float
          4      **Error**               3abc Illegal IDENT
          4      Semicolon
          5      Identifier      22              ab
          5      Add
          5      Semicolon
          5      Semicolon
          6      Identifier      25              for
          6      Left Small Bracket
          6      Integer
          6      Identifier      29              i
          6      Assign
          6      Number: 0
          6      Semicolon
          6      Identifier      29              i
          6      Less
```

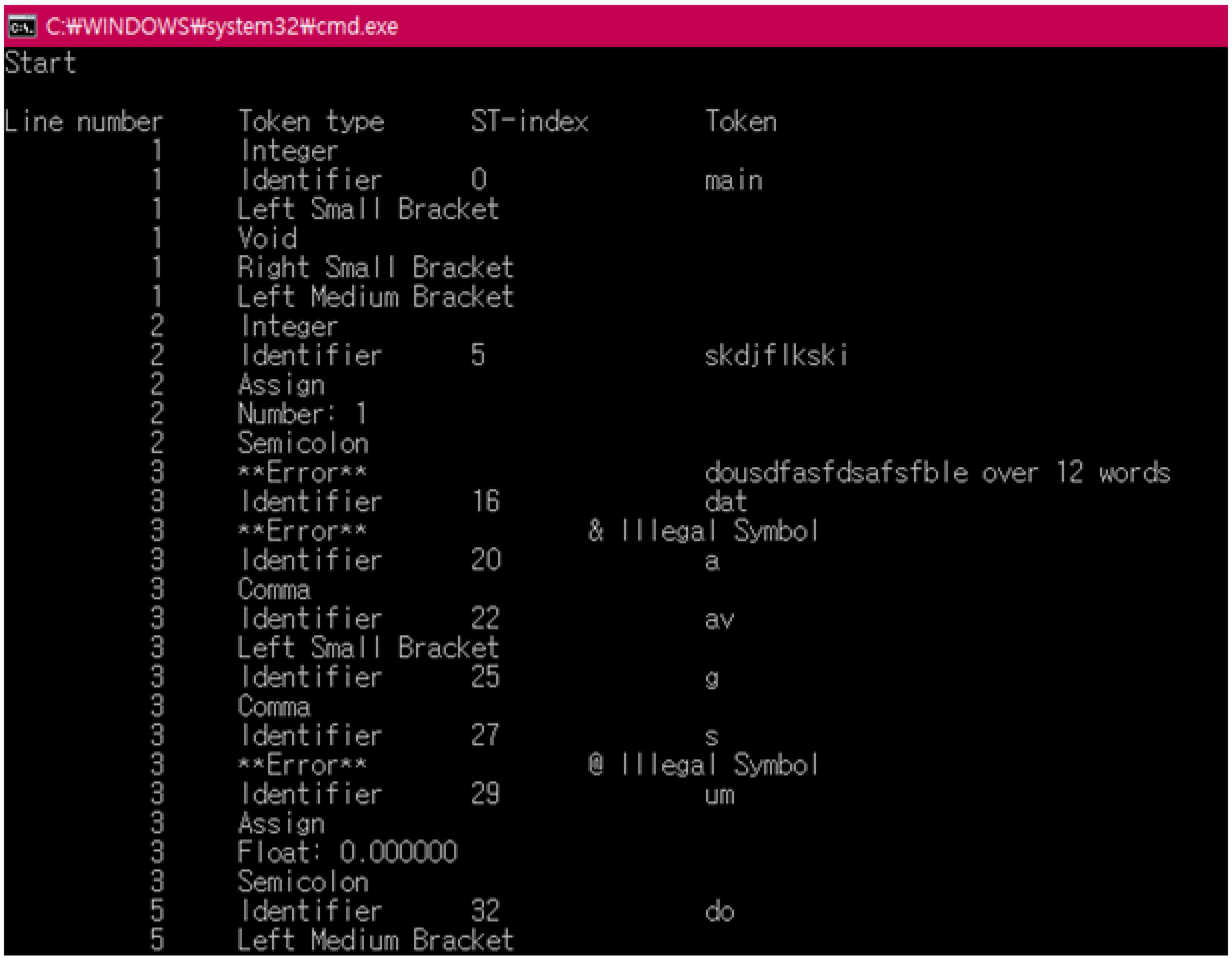


```
         6     Identifier      31              idl
         6     Semicolon
         6     Identifier      29              i
         6     Increase
         6     Right Small Bracket
         7     Left Medium Bracket
         8     **Error**               3abc Illegal IDENT
         8     Add and assign
         8     Identifier      29              i
         8     Semicolon
         9     Right Medium Bracket
        10     Right Medium Bracket

End
2 errors detected


[[ HASH TABLE ]]

Hash code 5  : i
Hash code 13 : idl
Hash code 23 : _func
Hash code 27 : for
Hash code 34 : float
Hash code 50 : integer
Hash code 95 : ab
Hash code 97 : a

<35 characters are used in the string table>
계속하려면 아무 키나 누르십시오 . . .
```

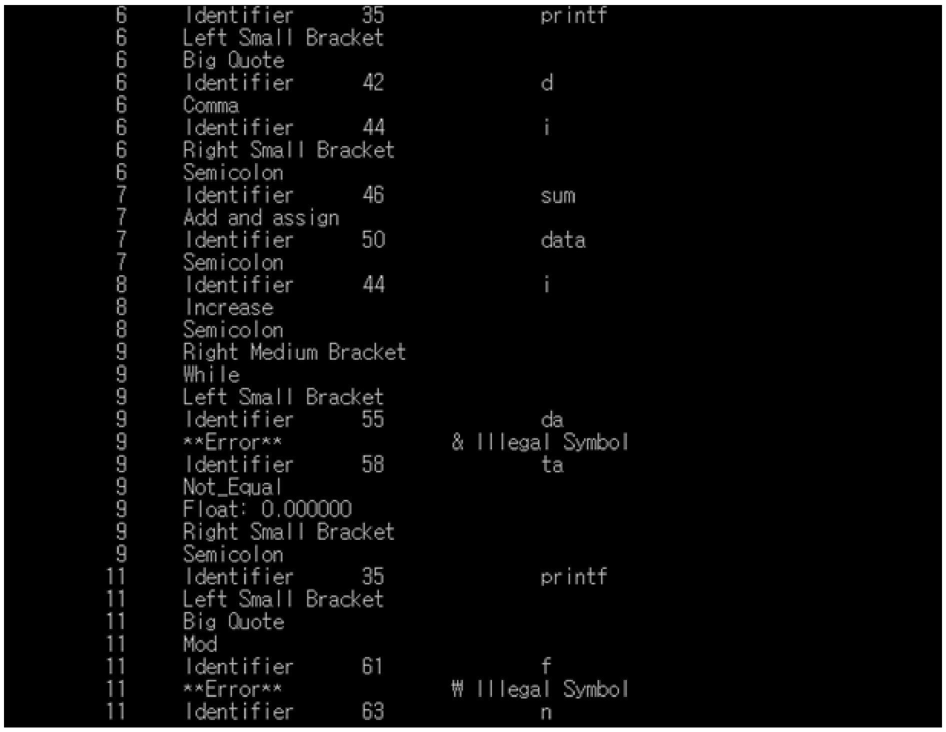
<with_error3.dat>


with_error3 - Windows 메모장
파일(F) 편집(E) 서식(O) 보기(V) 도움말(H)

```
int main(void) {
        int skdjflkski = 1;
        dousdfasfdsafsfble dat&a, av(g, s@um = 0.0;

        do {
                printf("d, i);
                sum += data;
                i++;
        } while (da&ta != 0.0);

        printf("%f₩n", sum / (i - 2));
}
```



```
Start

Line number     Token type      ST-index        Token
          1     Integer
          1     Identifier      0               main
          1     Left Small Bracket
          1     Void
          1     Right Small Bracket
          1     Left Medium Bracket
          2     Integer
          2     Identifier      5               skdjflkski
          2     Assign
          2     Number: 1
          2     Semicolon
          3     **Error**                       dousdfasfdsafsfble over 12 words
          3     Identifier      16              dat
          3     **Error**               & Illegal Symbol
          3     Identifier      20              a
          3     Comma
          3     Identifier      22              av
          3     Left Small Bracket
          3     Identifier      25              g
          3     Comma
          3     Identifier      27              s
          3     **Error**               @ Illegal Symbol
          3     Identifier      29              um
          3     Assign
          3     Float: 0.000000
          3     Semicolon
          5     Identifier      32              do
          5     Left Medium Bracket
```



```
      6     Identifier      35              printf
      6     Left Small Bracket
      6     Big Quote
      6     Identifier      42              d
      6     Comma
      6     Identifier      44              i
      6     Right Small Bracket
      6     Semicolon
      7     Identifier      46              sum
      7     Add and assign
      7     Identifier      50              data
      7     Semicolon
      8     Identifier      44              i
      8     Increase
      8     Semicolon
      9     Right Medium Bracket
      9     While
      9     Left Small Bracket
      9     Identifier      55              da
      9     **Error**               & Illegal Symbol
      9     Identifier      58              ta
      9     Not_Equal
      9     Float: 0.000000
      9     Right Small Bracket
      9     Semicolon
     11     Identifier      35              printf
     11     Left Small Bracket
     11     Big Quote
     11     Mod
     11     Identifier      61              f
     11     **Error**               ₩ Illegal Symbol
     11     Identifier      63              n
```

5. Contribution

1615015	박기은	1/3
1617029	이혜인	1/3
1871056	한지수	1/3

6. Additional function

- countline.c: 여러줄 주석 /* */의 라인의 개수를 세기 위한 함수. /* */ 부분의 토큰에서 개행문자 '\n' 를 센다.