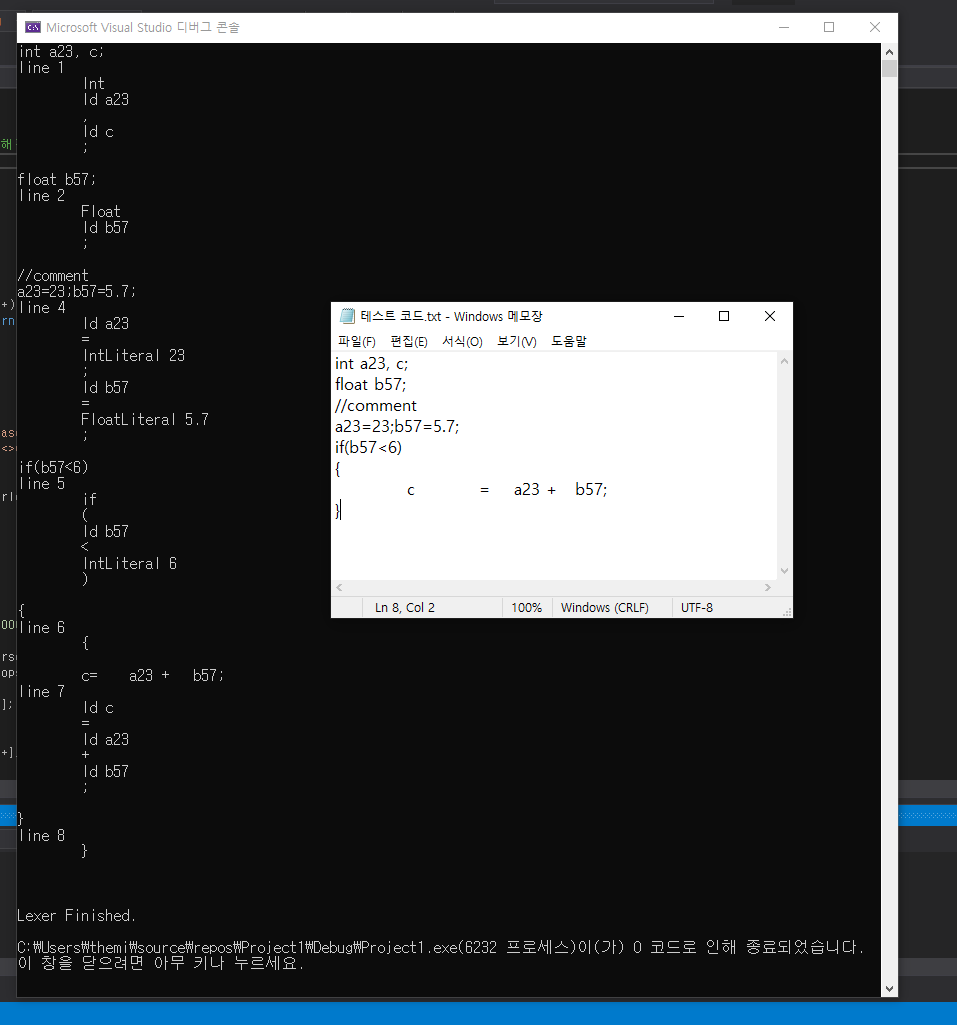
테스트 화면



어려웠던 점

1. int, float 등의 타입명과 +, <= 등 연산기호를 코드 안에서 추출하는 것
2. = 혹은 ;와 같이 문자 바로 옆에 붙어 있어도 문자열의 일부가 아닌 문자를 따로 구분하는 것
3. 기본 타입이 아닌 추상화 타입인 구조체를 인식하는 것(만들지 못함)

코드

#include <string.h>

#include <stdio.h>

//\_CRT\_SECURE\_NO\_WARNINGS 문제 해결

#pragma warning(disable:4996)

char buf[1000] = "\0";

int exit\_code = 1;

int in\_it(char chk, char\* set)

{

int len = strlen(set);

for (int i = 0; i < len; i++)

if (chk == set[i]) return 1;

return 0;

}

void input()

{

char temp[1000] = "\0";

char\* charset = "qwertyuiopasdfghjklzxcvbnmQWERTYUIOPASDFGHJKLZXCVBNM1234567890";

char\* opset = ",=+-\*/%%!&|;<>(){}[]";

gets(temp);

int i = 0, j = 0, size = strlen(temp);

if (size == 0)

{

exit\_code = 0;

return;

}

while ((i < size) && (j < 1000) && (temp[i] != '\n'))

{

if ((in\_it(temp[i], charset) && in\_it(temp[i + 1], opset))

|| (in\_it(temp[i], opset) && in\_it(temp[i + 1], charset)))

{

buf[j++] = temp[i++];

buf[j++] = ' ';

}

else buf[j++] = temp[i++];

}

for (i = 0; temp[i] != '\0'; i++) temp[i] = '\0';

for (i = j; buf[i] != '\0'; j++) buf[j] = '\0';

}

//문자열에 속한 탭 문자를 전부 공백 문자로 바꾼다

void pre\_lexer(char\* str)

{

int size = strlen(str);

for (int i = 0; i < size; i++)

if (str[i] == '\t') str[i] = ' ';

}

//문자열 str의 start에서 시작해서 end 직전에서 끝나는 부분 문자열을

//동적할당한 char타입 배열 ret에 복사해 반환한다

char\* substr(char\* str, int end)

{

char\* ret = (char\*)malloc(sizeof(char) \* (end + 2));

strncpy(ret, str, end + 1);

ret[end + 1] = '\0';

return ret;

}

//str의 idx부터 시작하는 문자가

//문자열의 집합 set 안에 있는지 판단

int in\_opers(char\* str)

{

char set[18][3] =

{

"==", "!=", "<", ">", "<=", ">=",

"%%", "\*\*", "\*", "/", "+", "-", "=",

"!", "&&", "||",

",", ";"

};

int size = 18, len;

for (int i = 0; i < size; i++)

{

len = strlen(set[i]);

if (strcmp(substr(str, strlen(set[i])), set[i]) == 0)

return len;

}

return 0;

}

//str의 idx부터 시작하는 문자가

//문자열의 집합 set 안에 있는지 판단

int in\_types(char\* str)

{

char set[4][6] = { "bool", "char", "int", "float" };

int size = 4, len;

for (int i = 0; i < size; i++)

{

len = strlen(set[i]);

if (strcmp(substr(str, strlen(set[i])), set[i]) == 0)

return len;

}

return 0;

}

//str의 idx부터 시작하는 문자가

//문자열의 집합 set 안에 있는지 판단

int in\_loops(char\* str)

{

char set[][6] = { "if", "else", "while", "for" };

int size = 4, len;

for (int i = 0; i < size; i++)

{

len = strlen(set[i]);

if (strcmp(substr(str, strlen(set[i])), set[i]) == 0)

return len;

}

return 0;

}

//str의 idx부터 시작하는 문자가

//문자열의 집합 set 안에 있는지 판단

int in\_bracs(char\* str)

{

char set[][2] = { "(", ")", "{", "}", "[", "]" };

int size = 6, len;

for (int i = 0; i < size; i++)

{

len = strlen(set[i]);

if (strcmp(substr(str, strlen(set[i])), set[i]) == 0)

return len;

}

return 0;

}

//str의 idx부터 시작하는 문자가

//문자열의 집합 set 안에 있는지 판단

int in\_nums(char\* str)

{

char set[] = "0123456789";

for (int i = 0; i < 10; i++)

if (str[0] == set[i])

return 1;

return 0;

}

//해당 단어에 .이 들어가있다면 에러거나 float이다

int is\_float(char\* str)

{

int size = strlen(str);

for (int i = 0; i < size; i++)

if (str[i] == '.') return 0;

return 1;

}

int main()

{

int line\_num = 1;

char\* ptr = NULL;

while (exit\_code)

{

for (int i = 0; buf[i] != '\0'; i++) buf[i] = '\0';

//명령어를 한 줄 입력받는다

input();

//buf 안에 있는 탭 문자를 전부 공백으로 치환한다

pre\_lexer(buf);

//입력받은 줄의 맨 앞 단어를 ptr에 저장

ptr = strtok(buf, " ");

//ptr의 맨 앞 두 글자가 "//"라면 주석이므로 이번 줄을 넘긴다

if (ptr != NULL && ptr[0] == '/' && ptr[1] == '/')

{

line\_num++;

continue;

}

//입력받은 줄이 주석이거나 공백이 아니라면 이번 줄의 line\_num을 출력한다

if(ptr != NULL) printf("line %d\n", line\_num++);

//입력받은 줄의 모든 단어에 대해

while (ptr != NULL)

{

int len = strlen(ptr);

printf("\t");

if (len = in\_types(ptr))

ptr[0] = ptr[0] - 'a' + 'A';

else if (len = in\_nums(ptr))

{

if (is\_float(ptr) == 0) printf("FloatLiteral ");

else printf("IntLiteral ");

}

else if (len = in\_loops(ptr));

else if (len = in\_opers(ptr));

else if (len = in\_bracs(ptr));

else printf("Id ");

printf("%s\n", ptr);

for (int i = 0; i < len; i++) ptr[i] = ' ';

ptr = strtok(NULL, " ");

}

printf("\n");

}

printf("Lexer Finished.\n");

return 0;

}