Project #1. Scanner Report

2018009234 한관희

1. 컴파일 환경

Windows Subsystem for Linux를 이용하여 Ubuntu 20.04.5 LTS에서 작성된 프로젝트입니다. 과제 명세서에 작성되어 있는 Makefile를 이용하기 때문에 gcc로 컴파일 됩니다. make all 명령어로 cminus_cimpl과 cminus_lex 두 개의 실행 파일을 만들 수 있습니다.

2. 구조 분석

이번 과제에서 Scanner는 2개의 공통된 소스 파일 main.c, util.c와 C 코드로 작성되어 있는 scan.c 혹은 flex를 통해 생성되는 lex.yy.c로 구성됩니다.

main.c에서는 command line argument로 들어온 파일의 이름으로 파일을 열고, 해당 파일의 EOF를 만나기 전까지 getToken 함수를 반복하여 호출합니다.

util.c에서는 token의 종류에 따라, token을 출력하는 printToken 함수가 작성되어 있습니다.

scan.c와 lex.yy.c에는 실질적으로 tokenize를 진행하는 getToken 함수가 작성되어 있습니다. TraceScan을 TURE로 설정하였기 때문에, getToken 함수는 반환되기 전에 출력까지 담당합니다.

3. Method1, Method2 공통 수정 사항

먼저 과제에 제시된 스펙에 맞추기 위하여 globals.h에서 enum을 수정하였습니다. 6개의 reserved word와 19개의 symbol의 token type을 추가하였습니다. 또, util.c에서 printToken의 출력 부분을 수정한 token type에 맞게 수정하였습니다.

4. Method 1: scan.c 수정사항

1) + - * ; , ( ) [ ] { }

해당 symbol들은 다음 문자를 확인해 볼 필요 없는 symbol들입니다. 따라서 처음으로 읽은 문자가 위에 symbol들에 해당한다면 그것에 맞는 token type을 반환합니다.

2) < 와 <= >와 >= =와 ==

해당 symbol들은 다음 문자까지 확인해야만 하는 symbol들입니다. 예를 들어 이번에 읽은 문자가 <이라면 다음 문자까지 읽습니다. 다음으로 읽은 문자가 =이라면 구해진 token은 <=입니다. 다음으로 읽은 문자가 =가 아니라면 구해진 token은 <이고, <=인지 확인하려고 읽은 문자는 되돌립니다.

3) !=

해당 symbol은 다음 문자까지 확인해야만 하는 symbol입니다. 2)와 다른 점은 !를 읽고 다음으로 읽은 문자가 =가 아니라면 ERROR인 점입니다.

4) /

해당 symbol은 다음 문자까지 확인해야만 하는 symbol입니다 /을 읽고 다음으로 읽은 문자가 *가 아니라면 구해진 token은 /입니다. /을 읽고 다음으로 읽은 문자가 *인 경우는 5)에서 설명하도록 하겠습니다.

5) /*

/을 읽고 다음으로 읽은 문자가 *인 경우에는 주석으로 판단합니다. 이 경우 */를 만나기 전까지 계속 주석으로 처리합니다. */를 만났는 지를 체크하기 위한, *를 만난 경우는 6)에서 설명하도록 하겠습니다. 만약 */를 만나기 전에 EOF를 만날 경우 더 이상의 tokenize를 종료하기 위해 ENDFILE을 반환합니다. *와 EOF를 제외한 모든 문자는 주석 처리되므로 다음 문자로 계속하여 넘어갑니다.

6) */

주석으로 계속 처리되고 있는 상태(state == INCOMMENT)에서 *를 만난 경우 주석 처리를 끝내기 위해 다음 문자를 확인해야만 합니다. 만약 다음으로 읽는 문자가 *인 경우 다시 한번 다음 문자를 읽기 위해서 현재 상태를 유지합니다. 만약 다음으로 읽는 문자가 /인 경우 더 이상의 주석 처리를 멈추기 위해 state와 tokenStringIndex를 초기값으로 초기화합니다. 만약 다음으로 읽는 문자가 *도 /도 아닌 경우, 계속 주석 처리를 해야만 하므로 5)의 처리를 반복합니다.

7) NUM, ID

NUM과 ID의 경우 해당 token의 길이를 알 수 없습니다. 따라서 처음으로 읽은 문자가 0~9인 경우, NUM으로 판단하여 더이상 0~9가 나오지 않을 때까지 문자열을 저장합니다. 만약 0~9가 아닌 문자를 읽는다면 읽은 문자를 되돌리고 현재 tokenize를 종료합니다 처음으로 읽은 문자가 a~z 혹은 A~Z인 경우, ID로 판단하여 더이상 a~z, A~Z, 혹은 0~9가 아닌 경우가 나올 때까지 문자열을 저장합니다. 해당 경우가 모두 아닌 문자를 읽는 경우 읽은 문자를 되돌리고 현재 tokenize를 종료합니다. 이후 이번에 찾은 token이 reserved word인지를 검사한 후 만약 맞다면 token type을 바꿔서 반환합니다.

5. Method 2: cminus.l

제공된 파일인 tiny.l을 수정하여 cminus.l을 작성하였습니다. Definition Section에서 letter를 과제에 제시된 스펙에 맞춰 변경하였습니다. Rule Section에서는 reserved word, symbol을 새로 수정한 token type을 반환하도록 수정하였습니다. 주석의 경우 "*/"를 만나야 주석을 끝내기 때문에 새로운 문자를 읽기 전에 다른 변수에 미리 저장을 합니다. 한번 전에 읽은 문자가 '*'이고 이번에 읽은 문자가 '/'인 경우 주석을 종료하도록 수정하였습니다.

6. 에러 처리

만약 인가되지 않은 문자(& $등등)가 읽히거나 ! 이후 =가 나오지 않는 경우에는 에러로 처리해야 합니다. 이에 대한 처리 방법이 과제 명세에는 나와있지 않지만, 조교님께서 수업 시간에 설명해주신 내용을 참고하였습니다. 비정상적인 input이 나오더라도 Segmentation fault나 infinite loop 없이 EOF를 만날 때까지 읽어야 하므로, 기존 주어진 TINY Compiler의 ERROR 출력 경우와 같이 출력한 후 끝까지 Tokenize를 진행합니다.

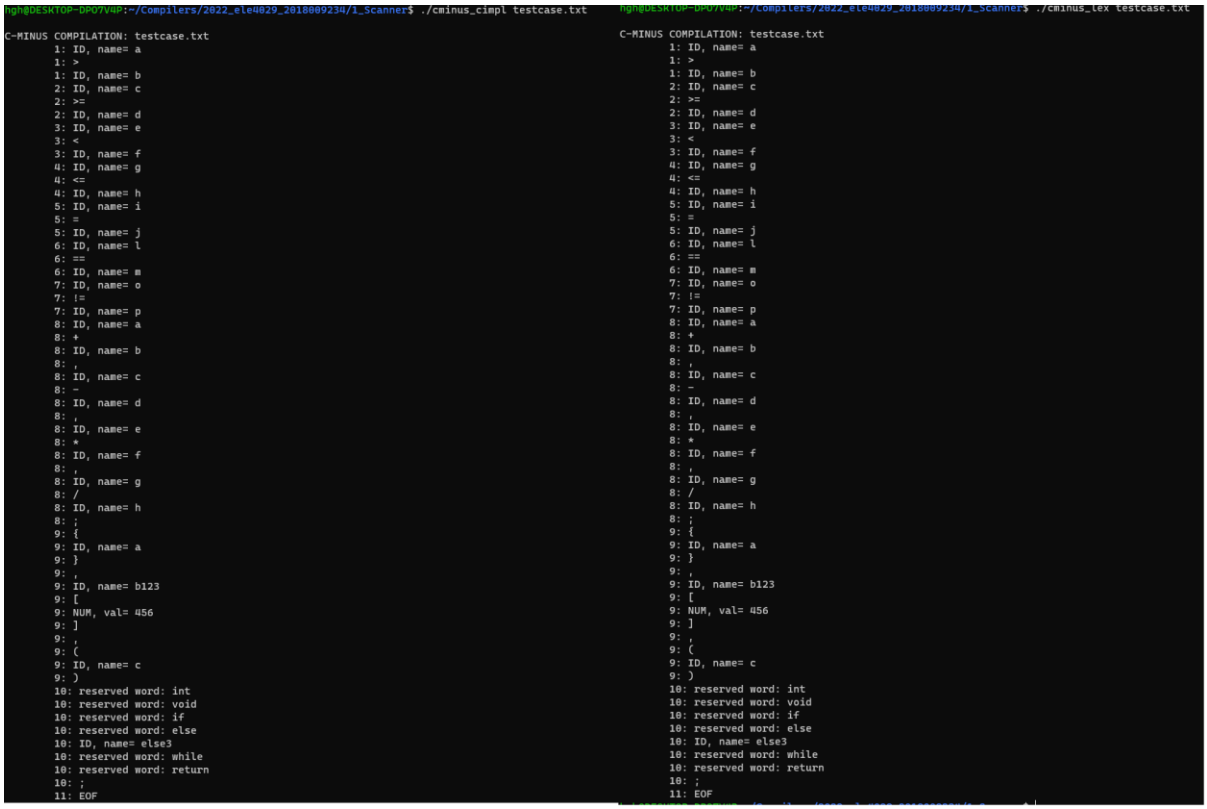
7. 테스트 케이스 결과

개인적으로 제작한 테스트 케이스에 대한 결과입니다. testcase.txt는 정상적인 결과를 보기 위해 작성하였고, testcase2.txt는 주석의 처리와 잘못된 input에 대한 결과를 보기 위해 작성하였습니다.

```
a > b
c >= d
e < f
g <= h
i = j
l == m
o != p
a + b, c - d, e * f, g / h;
{a}, b123[456], (c)
int void if else else3 while return;

```

<testcase.txt>



<testcase.txt에 대한 결과(좌: scan.c, 우: flex)>

```
1   a = 3;
2   /*****a+b*********/
3   b+-a;
4   !a;
5   /*
6   c = d * e;
7
```

<testcase2.txt>

```
hgh@DESKTOP-DPO7V4P:~/Compilers/2022_ele4029_2018009234/1_Scanner$ ./cminus_cimpl testcase2.txt

C-MINUS COMPILATION: testcase2.txt
        1: ID, name= a
        1: =
        1: NUM, val= 3
        1: ;
        3: ID, name= b
        3: +
        3: -
        3: ID, name= a
        3: ;
        4: ERROR: !
        4: ID, name= a
        4: ;
        7: EOF
hgh@DESKTOP-DPO7V4P:~/Compilers/2022_ele4029_2018009234/1_Scanner$ ./cminus_lex testcase2.txt

C-MINUS COMPILATION: testcase2.txt
        1: ID, name= a
        1: =
        1: NUM, val= 3
        1: ;
        3: ID, name= b
        3: +
        3: -
        3: ID, name= a
        3: ;
        4: ERROR: !
        4: ID, name= a
        4: ;
        7: EOF
```

<testcase2.txt(상: scan.c, 하: flex)>