컴파일러설계

이헌준 교수님

2019032160 유석원

### **Project 1**

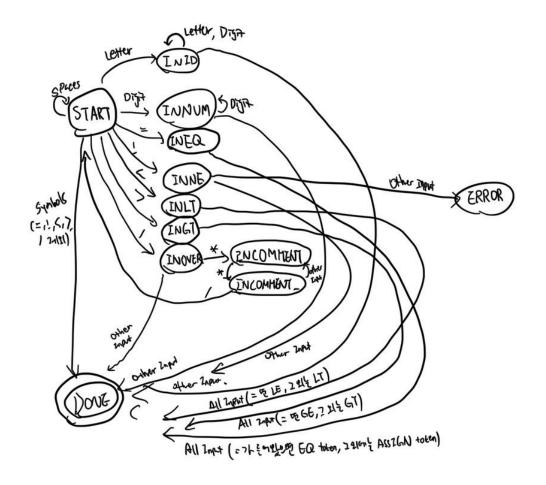
# 개요

이번 Project 1 과제는 C-Minus scanner를 만드는 것이 목표입니다. 기존의 Tiny 컴파일러의 C 코드와 Lex 코드를 고치는 것으로 C-Minus scanner를 성공적으로 만들 수 있었습니다. Scanner는 문자열로 이루어진 코드를 입력받고, 해당 문자열들을 토큰으로 만들어 식별된 토큰을 출력하게 됩니다. C 코드는 주어진 규칙에 맞춰 DFA를 만들고, state를 설정하는 것으로 완성할 수 있었고, Lex 코드는 lexical rule들을 정의하는 것으로 완성할 수 있었습니다.

이번 프로젝트에서 C-Minus scanner를 만들려면 여러 가지 규칙을 따라야 했습니다. C-minus의 예약어는 "int", "void", "if", "else", "while", "return" 6개로 정해집니다. 해당 단어들을 발견하게 되면 reserved word 토큰으로 인식하고 "\time\text{lineNo: reserved word: 해당예약어"의 형태로 출력이 됩니다. Symbol은 19종으로 "+", "-", "", "/", "<", "≤", ">", "≥", "==", "≠", "=", ";", ",", "(", ")", "[", "]", "{", "}"로 이루어져 있습니다. ID는 letter(letter|digit)의형태로 입력되며, NUM은 digit digit의 형태로 입력됩니다. 공백 ('', '\text{\text{\text{\text{W}}n'}, '\text{\t

#### C 코드

위의 규칙에 맞춰 먼저 util.c 파일을 수정했습니다. 해당 파일은 토큰 출력에 대한 함수인 printToken 함수가 있으므로, 규칙에 맞춰 수정을 해주었습니다. 다음으로 scan.c 파일에서도 마찬가지로 예약어에 관련된 변수를 수정했습니다. Scanner의 메인 로직은 getToken 함수에서 이루어지므로, 해당 함수의 로직을 수정했습니다. C 코드의 경우, 한글자씩 입력을 읽어와 state를 확인하면서 token을 식별하기 때문에 DFA가 필수적입니다. 다만, DFA로 그릴 경우, 너무나도 많은 state를 그려야 하므로 간단한 NFA를 그려표현하고, 코드로 바꾸었습니다.



현재 그림의 NFA에는 ERROR state가 따로 존재하지만, 코드상에서는 별도로 선언하지 않았습니다. 모든 잘못된 입력은 switch-case 문의 default로 떨어져 ERROR 토큰을 리턴하며, START state에서 !를 받은 뒤 =가 나오지 않는 경우에도 ERROR 토큰을 리턴합니다. 주석을 처리하는 중, 주석이 닫히지 않고 파일이 끝나는 경우, EOF를 출력합니다. 그외에는 대부분의 코드가 기존 Tiny 코드와 같습니다.

```
1: reserved word: void
1: ID, name= main
                                                                                                                                                3: reserved word: int
3: ID, name= i
                                                                                                                                                 3: ;
3: reserved word: int
                                                                                                                                                    NUM, val= 5
actus@cactus:~/compiler/CompilerDesign/Project1/files$ ./cminus cimpl test.1.txt
                                                                                                                                                5: ID. name= i
                                                                                                                                                5: NUM, val= 0
                                                                                                                                                    reserved word: while
                                                                                                                                                6: NUM, val= 5
                                                                                                                                                8: ID, name= x
           NUM, val= 0
                                                                                                                                                8: ID, name= i
                                                                                                                                               8: ID, name= input
8: (
8: )
                                                                                                                                                 10: ID, name= i
                                                                                                                                                10: ID, name= i
                                                                                                                                                10: NUM, val= 1
                                                                                                                                                11: }
                                                                                                                                                13: ID, name= i
                                                                                                                                                13: =
13: NUM, val= 0
            reserved word: void
ID, name= main
                                                                                                                                                 14: reserved word: while
                                                                                                                                                14: (
14: ID, name= i
                                                                                                                                                14: <=
                                                                                                                                                14: NUM, val= 4
                                                                                                                                               14: )
15: {
16: reserved word: if
16: (
16: ID, name= x
                                                                                                                                                16: [
16: ID, name= i
                                                                                                                                                16: NUM, val= 0
                                                                                                                                                18: ID, name= output
18: (
       14: 7
14: ;
15: ID, name= outp
15: (
15: ID, name= gcd
15: (
15: ID, name= x
15: TD, name= y
15: D, name= y
15: )
15: )
15: ;
16: }
17: EOF
                                                                                                                                                18: ID, name= x
                                                                                                                                                18: [
18: ID, name= i
                                                                                                                                                18: ]
                                                                                                                                               18: )
18: ;
19: }
```

TINY COMPILATION: test.2.txt

#### Lex 코드

Lex 코드는 globals.h, util.h, main.c를 사용하고, scan.c 파일은 사용하지 않습니다. Lex 내부의 %% //code %% 사이에 lex rule이 정의되어 해당 rule이 scan.c의 getToken 함수와 같은 역할을 하게 됩니다. 주석을 제외하고는 두 글자도 입력을 받아올 수 있기 때문에 예약어와 모든 symbol을 저장해주고, newline은 lineno를 하나 증가시키고,

whitespace는 무시하며, 주석의 시작인 /\*가 입력되면 주석을 닫는 \*/를 받을 때까지 한 글자씩 입력을 읽어오는 반복문을 실행합니다. 주석이 닫히기 전까지는 계속 무시하며,

그러다 주석이 닫히지 않은 상태로 EOF에 도달할 수 있기 때문에 읽은 문자가 EOF나 '\0'가 되면 반복문을 빠져나옵니다.

```
cactus@cactus:~/compiler/CompilerDesign/Project1/files$ ./cminus_lex test.2.txt
                                                                                                                                                                             TINY COMPILATION: test.2.txt
                                                                                                                                                                                             1: reserved word: void
1: ID, name= main
1: (
                                                                                                                                                                                             1: reserved word: void
1: )
                                                                                                                                                                                             2: {
3: reserved word: int
3: ID, name= i
              cactus:~/compiler/CompilerDesign/Project1/files$ ./cminus_lex test.1.txt
TINY COMPILATION: test.1.txt
              4: reserved word: int
4: ID, name= gcd
4: (
                                                                                                                                                                                           3: ID, name= x
3: [
3: NUM, val= 5
3: ]
3: ;
5: ID, name= i
5: =
5: NUM, val= 0
               4: reserved word: int
4: ID, name= u
               4: ,
4: reserved word: int
4: ID, name= v
               6: reserved word: if
6: (
6: ID, name= v
                                                                                                                                                                                             6: (
6: ID, name= i
                                                                                                                                                                                             6: <
               6: NUM, val= 0
                                                                                                                                                                                             6: NUM, val= 5
6: )
7: {
               6: reserved word: return
6: ID, name= u
                                                                                                                                                                                             8: I
8: [
                                                                                                                                                                                                   ID, name= x
               6:;
7: reserved word: else
7: reserved word: return
7: ID, name= gcd
7: (
7: ID, name= v
                                                                                                                                                                                             8: ID, name= i
8: ]
               7: ,
7: ID, name= u
             7: ID, name= u
7: -
7: ID, name= u
7: /
7: ID, name= v
7: M; *
7: ID, name= v
7: *
7: ID, name= v
7: }
9: }
11: reserved word: void
11: ID, name= main
11: (11: ID, name= word: void
11: ID: name= word: void
11: ID: name= main
11: (11: ID: name= word: void
11: ID: reserved word: void
                                                                                                                                                                                            8: ;
10: ID, name= i
                                                                                                                                                                                             10: =
10: ID, name= i
                                                                                                                                                                                             10: +
10: NUM, val= 1
                                                                                                                                                                                             10: ;
11: }
                                                                                                                                                                                             13: ID. name= i
                                                                                                                                                                                             13: =
13: NUM, val= 0
               11: reserved word: void
                                                                                                                                                                                             13:
                                                                                                                                                                                             13: ;
14: reserved word: while
               11: )
12: {
13: reserved word: int
13: ID, name= x
                                                                                                                                                                                             14: (
                                                                                                                                                                                             14: ID, name= i
14: <=
                                                                                                                                                                                           14: <=
14: NLM, val= 4
14: )
15: {
16: reserved word: if
16: (
16: ID, name= x
16: [
16: [
16: D, name= x
             13: ;
14: ID, name= x
14: =
14: ID, name= input
14: (
               14: )
14: ;
14: ID, name= y
14: =
                                                                                                                                                                                             16: ID, name= i
16: ]
16: !=
              14: ID, name= input
14: (
14: )
                                                                                                                                                                                             16: NUM, val= 0
16: )
                                                                                                                                                                                             17: {
18: ID, name= output
18: (
              14: ;

15: ID, name= output

15: (

15: ID, name= gcd

15: (

15: ID, name= x
                                                                                                                                                                                             18: ID, name= x
18: [
                                                                                                                                                                                            18: [

18: ID, name= i

18: ]

18: )

18: ;

19: }
              15: ID, name= x
15: ,
15: ID, name= y
15: )
15: )
15: ;
16: }
17: EOF
                                                                                                                                                                                             22: EOF
```

## 빌드

PPT의 설명을 따라 main.c 파일에서 NO\_PARSE와 TraceScan을 TRUE로 설정했고, EchoSource는 FALSE로 설정했습니다. Makefile이 제공되기 때문에 해당하는 makefile과 test/result 텍스트 파일을 기존의 컴파일러 파일들이 있는 디렉토리에 함께 뒀으며, 모든

결과를 확인한 뒤 현재 "make clean" 명령어를 실행해 오브젝트 파일과 cminus\_cimpl, cminus\_lex 파일은 삭제된 상태입니다.

LMS에 업로드되어 있는 zip 파일은 Mac에서 압축한 파일이며, 혹시 오류로 인해 압축이 해제되지 않거나 압축을 해제하고 파일이 존재하지 않는 경우 연락해 주시기 바랍니다.