2018 Compiler Project #1. Scanner

소프트웨어전공 2016024793 김유진

1. Compilation method and environment

1.1. Compilation:

- "make" for cminus (scanner using C-code)
- "make cminus_flex" for cminus_flex (scanner using flex)

1.2. Environment: Ubuntu 18.04.1 LTS

2. Implementation of C-Scanner using C-code (modify compiler code)

2.1. compiler code 수정사항

2.1.1. globals.h

키워드와 심볼 등록을 위해 enum 값을 수정한다.

MAXRESERVED 는 reserved words 의 개수를 의미한다. 이때, tiny 에서 사용하던 reserved words 를 삭제하면 오류가 발생하므로, C-Minums 사용하는 키워드는 6 개이지만 기존의 키워드를 포함시킨 12 개로 지정해주어야 한다.

2.1.2. scan.c

scan.c 파일에서 DFA 를 수행하기 위해서 코드를 수정해야 한다.

먼저, C-Minus DFA 의 states 를 추가해 준다.

- INEQ: ASSIGN(=) 와 EQ(==)를 구분
- INLT: LT(<) 와 LE(<=)를 구분
- INGT: GT(>)와 GE(>=)를 구분
- INNE: NE(!=)를 확인
- INOVER: OVER(/)임을 확인하거나 다음 심볼이 *인 경우 INCOMMENT 로 넘겨줌
- INCOMMNET: 주석 상태. 다음 심볼이 *인 경우 INCOMMENT_로 넘겨줌
- INCOMMENT_: */임을 확인. 만약 다음 심볼이 /가 아닌 경우 계속 주석 상태이다.

다음, globals.h 에서 추가해줬던 reserved words 를 scan.c 의 파일의 lookup table 에도 추가해준다.

마지막으로, getToken()함수를 수정한다. 2 글자 짜리 연산자들을 추가한다. 특히, 주석처리 관련 state 인 INOVER, INCOMMENT, INCOMMENT_에서 주석의 경우 save = FALSE; 처리해주어야 한다.

2.1.3. util.c

tiny 에서 사용하지 않았던 token 과 달라진 token 을 printToken()함수에서 수정해준다.

1.2. Result using sample code, 'test.cm' file

```
clare@ubuntu:~/Desktop/2018comp/loucomp$ ./cminus test.cm

CMINUS COMPILATION: test.cm
   1: /* A program to perform Euclid's
   2:    Algorithm to compute gcd */
   3:
   4: int gcd(int u, int v)
        4: reserved word: int
        4: ID, name= gcd
        4: (
        4: reserved word: int
        4: ID, name= u
        4: ,
        4: reserved word: int
        4: ID, name= v
        4: )
   5: {
        5: {
   6:   if(v==0) return u;
        6: reserved word: if
        6: (
        6: ID, name= v
        6: ==
        6: NUM, val= 0
        6: )
        6: reserved word: return
        6: ID, name= u
        6: ;
   7:   else return gcd(v,u-u/v*v);
        7: reserved word: else
        7: reserved word: return
        7: ID, name= gcd
        7: (
        7: ID, name= v
        7: ,
        7: ID, name= u
        7: -
        7: ID, name= u
        7: /
        7: ID, name= v
        7: *
        7: ID, name= v
        7: )
        7: ;
   8:   /* u-u/v*v == u mod v*/
   9: }
        9: }
```

```
 10:
 11: void main(void)
       11: reserved word: void
       11: ID, name= main
       11: (
       11: reserved word: void
       11: )
 12: {
       12: {
 13:   int x; int y;
       13: reserved word: int
       13: ID, name= x
       13: ;
       13: reserved word: int
       13: ID, name= y
       13: ;
 14:   x = input();
       14: ID, name= x
       14: =
       14: ID, name= input
       14: (
       14: )
       14: ;
 15:   y = input();
       15: ID, name= y
       15: =
       15: ID, name= input
       15: (
       15: )
       15: ;
 16:   output(gcd(x,y));
       16: ID, name= output
       16: (
       16: ID, name= gcd
       16: (
       16: ID, name= x
       16: ,
       16: ID, name= y
       16: )
       16: )
       16: ;
 17: }
       17: }
       18: EOF
```

2. Implementation of C-Scanner using lex(flex) by tiny.l modification

2.1. cminus.l

tiny.l 파일을 복사하여 C-Minus 의 문법에 맞게 keywords 와 symbols 을 추가해 준다.

2.2. Result using sample code 'test.cm' file

```
CMINUS COMPILATION: test.cm
        4: reserved word: int
        4: ID, name= gcd
        4: (
        4: reserved word: int
        4: ID, name= u
        4: ,
        4: reserved word: int
        4: ID, name= v
        4: )
        5: {
        6: reserved word: if
        6: (
        6: ID, name= v
        6: ==
        6: NUM, val= 0
        6: )
        6: reserved word: return
        6: ID, name= u
        6: ;
        7: reserved word: else
        7: reserved word: return
        7: ID, name= gcd
        7: (
        7: ID, name= v
        7: ,
        7: ID, name= u
        7: -
        7: ID, name= u
        7: /
        7: ID, name= v
        7: *
        7: ID, name= v
        7: )
        7: ;
        9: }
       11: reserved word: void
       11: ID, name= main
       11: (
       11: reserved word: void
       11: )
       12: {
       13: reserved word: int
       13: ID, name= x
       13: ;
       13: reserved word: int
       13: ID, name= y
       13: ;
```

```
    14: ID, name= x
    14: =
    14: ID, name= input
    14: (
    14: )
    14: ;
    15: ID, name= y
    15: =
    15: ID, name= input
    15: (
    15: )
    15: ;
    16: ID, name= output
    16: (
    16: ID, name= gcd
    16: (
    16: ID, name= x
    16: ,
    16: ID, name= y
    16: )
    16: )
    16: ;
    17: }
    18: EOF
```