2013012041

안효민

컴파일러 설계

Project #1. Scanner

- **Compilation method and Environment**

1) Compilation using tiny compiler , Ubuntu 16.04.1 LTS

"scan.c"에 getToken()에 대한 스펙이 작성되어 있어서 오브젝트파일 생성한 후 "main.o", "util.o" 등 기타 오브젝트파일을 묶어서 "tiny.exe" 파일을 생성합니다.

2) Compilation using flex , Ubuntu 16.04.1 LTS

1) 방법과 유사하지만 "scan.c"를 이용하지 않고 flex에서 "lex.yy.o" 파일을 생성하여 다른 오브젝트 파일과 묶어서 "cminus_flex" 파일을 생성합니다.

- **Explanation and How to implement**

1) Compilation using tiny compiler

➔ "globals.h"는 PDF에 명세되어 있는 내용대로 수정하였습니다. 수정한 이유는 keyword와 token의 종류와 개수가 기존 tiny compiler와 다르기 때문입니다.

➔ "main.c" 또한 PDF와 같이 수정하였고 추가적으로 처음 파일을 실행하였을 때 나오는 문구를 "TINY COMPILATION" 대신 "C-MINUS COMPILATION"으로 수정하였습니다. 수정한 이유는 "scan.h" 헤더 파일이 필요하고 스캔할 파일의 내용과 scanner 결과물을 화면상에 출력하기 위함입니다.("main.c"의 flag를 수정함으로써 "scan.c"에서 파일 내용과 결과물 출력허용)

➔ "scan.c"의 수정내용

(1) StateType 내용수정 -> token의 종류가 달라졌기 때문에 state 또한 달라짐

(2) reservedWords[MAXRESERVED]의 내용수정 -> Keyword의 종류가 달라졌기 때문에 맞춰서 수정

(3) getToken() 함수의 내용수정 -> symbol의 종류와 기능이 달라졌기 때문에 스펙에 나와있는 것과 같이 수정 (ex. 기존의 assign은 := cminus의 assign은 =)

(4) case INCOMMENT와 INCOMMENT_, 즉 주석의 처리는 '/'가 나오면 INOVER 상태로 변경해서 연속해서 '*'이 나오면 주석으로 인정하여 INCOMMENT 상태로 변경되도록 했습니다. 그리고 다음에 나오는 lexeme들은 tokenString 배열 안에 삽입하지 않고 폐기했습니다. 만약 읽어들인 lexeme이 '*'이면 INCOMMET_ 상태로 변경되어 연속으로 '/'이 나오면 주석문이 끝난걸로 보아 다시 스캔작업을 수행하도록 하였고 연속으로 '/'이 나오지 않을 경우 다시 INCOMMENT 상태로 복귀해서 주석문으로 처리하도록 했습니다. 또한 주석이 끝나지 않은 상태에서 파일의 끝에 다다를 경우 ENDFILE 토큰을 반환하며 scanner가 종료되도록 하였습니다.

➔ "util.c" 내에서는 새로 추가된 keyword들을 case로 추가해주었고 token을 스펙에 맞게 수정해주었습니다.

2) Compilation using flex

➔ "globals.h", "main.c" 그리고 "util.c"는 위의 1)방법의 파일을 사용했습니다.

➔ "cminus.l"의 수정내용

(1) 기존의 "tiny.l" 파일을 바탕으로 작성했습니다.

(2) Keyword와 symbol을 스펙에 맞게 수정했습니다.

(3) 주석 처리의 경우 1)의 처리방법과 거의 흡사하며 "/*"이 나온 후 '*'이 나올 경우 연속으로 '/'이 나오는지 검사하여 나왔을 경우 주석문이 끝난걸로 처리했습니다. 그리고 만약 주석중간에 '₩n'이 나오면 lineno에 1을 더해줘서 한 줄을 띈 것으로 처리했습니다.

- **How to operate**

➔ 모든 소스코드들이 저장되어 있는 "project1_scanner" 파일 내에서 make명령을 통해 "tiny.exe" 파일과 "cminus_flex" 파일을 생성합니다.

➔ 명령어 "./tiny.exe 샘플파일명" 와 "./cminus_flex 샘플파일명" 을 통해서 파일을 실행하면 결과물이 출력됩니다.

- **Result Screenshot**

  ■ "tiny.exe" 실행파일로 "test.cm" 을 스캔한 결과

```
ubuntu@ubuntu-VirtualBox:~/2018_ELE4029_2013012041/project1_scanner$ ./tiny.exe test.cm

C-MINUS COMPILATION: test.cm
   1: /* A program to perform Euclid's
   2:   Algorithm to computer 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(); y= input();
       14: ID, name= x
       14: =
       14: ID, name= input
       14: (
       14: )
       14: ;
       14: ID, name= y
       14: =
       14: ID, name= input
       14: (
       14: )
       14: ;
  15:     output(gcd(x,y));
       15: ID, name= output
       15: (
       15: ID, name= gcd
       15: (
       15: ID, name= x
       15: ,
       15: ID, name= y
       15: )
       15: )
       15: ;
  16: }
       16: }
       17: EOF
```

■ “cminus_flex” 파일로 “test.cm” 을 스캔한 결과

```
ubuntu@ubuntu-VirtualBox:~/2018_ELE4029_2013012041/project1_scanner$ ./cminus_flex test.cm

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

■ “tiny.exe”로 “test.2.txt”를 스캔한 결과

```
ubuntu@ubuntu-VirtualBox:~/2018_ELE4029_2013012041/project1_scanner$ ./tiny.exe test.2.txt

C-MINUS COMPILATION: test.2.txt
   1: void main(void)
        1: reserved word: void
        1: ID, name= main
        1: (
        1: reserved word: void
        1: )
   2: {
        2: {
   3:    int i; int x[5];
        3: reserved word: int
        3: ID, name= i
        3: ;
        3: reserved word: int
        3: ID, name= x
        3: [
        3: NUM, val= 5
        3: ]
        3: ;
   4:
   5:    i = 0;
        5: ID, name= i
        5: =
        5: NUM, val= 0
        5: ;
   6:    while( i < 5 )
        6: reserved word: while
        6: (
        6: ID, name= i
        6: <
        6: NUM, val= 5
        6: )
   7:    {
        7: {
   8:            x[i] = input();
        8: ID, name= x
        8: [
        8: ID, name= i
        8: ]
        8: =
        8: ID, name= input
        8: (
        8: )
        8: ;
   9:
  10:           i = i + 1;
       10: ID, name= i
       10: =
       10: ID, name= i
       10: +
       10: NUM, val= 1
       10: ;
  11:    }
       11: }
  12:
  13:    i = 0;
       13: ID, name= i
       13: =
       13: NUM, val= 0
       13: ;
  14:    while( i <= 4 )
       14: reserved word: while
       14: (
       14: ID, name= i
       14: <=
       14: NUM, val= 4
       14: )
  15:    {
       15: {
  16:           if( x[i] != 0 )
       16: reserved word: if
       16: (
       16: ID, name= x
       16: [
       16: ID, name= i
       16: ]
       16: !=
       16: NUM, val= 0
       16: )
  17:           {
       17: {
  18:                   output(x[i]);
       18: ID, name= output
       18: (
       18: ID, name= x
       18: [
       18: ID, name= i
       18: ]
       18: )
       18: ;
  19:           }
       19: }
  20:    }
       20: }
  21: }
       21: }
       22: EOF
```