read me

complier hw1

Lex 版本

```
flex 2.6.4
```

作業平台

```
Ubuntu 20.04.2 LTS
```

執行方式

- 請按以下順序
  1. make clean
  2. make
  3. make test(如果要一次餵入公開測資)
- make file

```
FILE_lex=   B073040025.l
PROG_lex=   lex.yy.c
OUTPUT_demo = demo
all:
    flex $(FILE_lex)
    g++ $(PROG_lex) -L./ -l table -lfl -static -o $(OUTPUT_demo)
test:
    ./$(OUTPUT_demo) < test.java
    @echo
    ./$(OUTPUT_demo) < test0.java
    @echo
    ./$(OUTPUT_demo) < test1.java
    @echo
    ./$(OUTPUT_demo) < test2.java
    @echo
    ./$(OUTPUT_demo) < test3.java

clean:
    rm demo $(PROG_lex)
```

如何處理這份規格書上的問題

1. 所有需要判別的token，皆按照規格書上所寫，並以對應的正則表達式篩選合法及不合法格式。
2. 不合格者以.號接收並列印錯誤訊息
3. 不須印出者，接收到後只操作計算字數、行數的變數。
4. 以額外的.h檔及cpp檔時做四個table所需的functions

寫這個作業所遇到的問題

1. 正則表達式得反覆測試才有正確答案。
2. 正則表達式測試網站合格，程式開發平台卻是不合格。
3. ++及+、<=及<等，必須將前者的優先度設高，才可以正確匹配。否則++會匹配為兩個+等。
4. comment會有\n，但並不會先被newline匹配到，須遍歷yytext元素檢查，對相應變數做調整。
5. 再因為換行格式可能為\r\n，導致必須先將其過濾起來，以免output出錯。
6. invalid ID /string實在是煞費苦心，占了至少50%作業的時間。利用了lookup找table還有前面的reserve word判斷是否為宣告。

輸出

- 依照順序為test.java、test0.java、test1.java、test2.java、test3.java。
- 前二者為自己的測資，後三者為助教附的測資。

```
./demo < test.java
Line: 1, 1st char: 1, "public" is a "reserved word".
Line: 1, 1st char: 8, "class" is a "reserved word".
Line: 1, 1st char: 14, "Test1" is an "ID".
Line: 1, 1st char: 20, "{" is a "symbol".
Line: 2, 1st char: 5, "public" is a "reserved word".
Line: 2, 1st char: 12, "static" is a "reserved word".
Line: 2, 1st char: 19, "int" is a "reserved word".
Line: 2, 1st char: 23, "add" is an "ID".
Line: 2, 1st char: 26, "(" is a "symbol".
Line: 2, 1st char: 27, "int" is a "reserved word".
Line: 2, 1st char: 31, "a" is an "ID".
Line: 2, 1st char: 32, "," is a "symbol".
Line: 2, 1st char: 34, "int" is a "reserved word".
Line: 2, 1st char: 38, "b" is an "ID".
Line: 2, 1st char: 39, ")" is a "symbol".
Line: 2, 1st char: 41, "{" is a "symbol".
Line: 3, 1st char: 9, "return" is a "reserved word".
Line: 3, 1st char: 16, "a" is an "ID".
Line: 3, 1st char: 18, "+" is an "operator".
Line: 3, 1st char: 20, "b" is an "ID".
Line: 3, 1st char: 21, ";" is a "symbol".
Line: 4, 1st char: 5, "}" is a "symbol".
Line: 5, 1st char: 1, "}" is a "symbol".
The symbol table contains:
Test1
add
a
b
```

```
Line: 1, 1st char: 1, "int" is a "reserved word".
Line: 1, 1st char: 5, "555aiii" is an invalid "ID".
Line: 2, 1st char: 1, "int" is a "reserved word".
Line: 2, 1st char: 5, "3aaa" is an invalid "ID".
Line: 3, 1st char: 1, "int" is a "reserved word".
Line: 3, 1st char: 5, "a" is an "ID".
Line: 4, 1st char: 1, "float" is a "reserved word".
Line: 4, 1st char: 7, "asdasd" is an "ID".
Line: 6, 1st char: 1, "asdasd" is an "ID".
Line: 6, 1st char: 8, "=" is an "operator".
Line: 6, 1st char: 10, "5" is an "integer".
Line: 6, 1st char: 11, ";" is a "symbol".
Line: 7, 1st char: 1, "3" is an "integer".
Line: 7, 1st char: 2, "aaaa" is a "non-matched token".
Line: 7, 1st char: 7, "=" is an "operator".
Line: 7, 1st char: 9, "5" is an "integer".
Line: 7, 1st char: 10, ";" is a "symbol".
Line: 8, 1st char: 1, "a" is an "ID".
Line: 8, 1st char: 3, "=" is an "operator".
Line: 8, 1st char: 5, "4" is an "integer".
Line: 8, 1st char: 6, ";" is a "symbol".
Line: 10, 1st char: 1, ""wts\"ertg\"ertgerg"" is a "string".
Line: 12, 1st char: 1, ""qw\\eqweqeq\"eqe"" is a "string".
Line: 12, 1st char: 19, "weqwefqfqf" is a "non-matched token".
Line: 12, 1st char: 29, """ is a invalid "string".
Line: 14, 1st char: 1, ""qweqweqweqeqweqweqwe" is a invalid "string".
Line: 16, 1st char: 1, "awdawdawdawdawdawdawd" is a "non-matched token".
Line: 18, 1st char: 1, "'a----------wdwadad'" is a invalid "string".
Line: 19, 1st char: 1, ""kjlkjlkjkllkj----------" is a invalid "string".
Line: 21, 1st char: 1, ""\"" is a invalid "string".
The symbol table contains:
a
asdasd
```

```
./demo < test1.java
Line: 2, 1st char: 1, "public" is a "reserved word".
Line: 2, 1st char: 8, "class" is a "reserved word".
Line: 2, 1st char: 14, "Test1" is an "ID".
Line: 2, 1st char: 20, "{" is a "symbol".
Line: 3, 1st char: 5, "public" is a "reserved word".
Line: 3, 1st char: 12, "static" is a "reserved word".
Line: 3, 1st char: 19, "int" is a "reserved word".
Line: 3, 1st char: 23, "add" is an "ID".
Line: 3, 1st char: 26, "(" is a "symbol".
Line: 3, 1st char: 27, "int" is a "reserved word".
Line: 3, 1st char: 31, "a" is an "ID".
Line: 3, 1st char: 32, "," is a "symbol".
Line: 3, 1st char: 34, "int" is a "reserved word".
Line: 3, 1st char: 38, "b" is an "ID".
Line: 3, 1st char: 39, ")" is a "symbol".
Line: 3, 1st char: 41, "{" is a "symbol".
Line: 4, 1st char: 9, "return" is a "reserved word".
Line: 4, 1st char: 16, "a" is an "ID".
Line: 4, 1st char: 18, "+" is an "operator".
Line: 4, 1st char: 20, "b" is an "ID".
Line: 4, 1st char: 21, ";" is a "symbol".
Line: 5, 1st char: 5, "}" is a "symbol".
Line: 7, 1st char: 5, "public" is a "reserved word".
Line: 7, 1st char: 12, "static" is a "reserved word".
Line: 7, 1st char: 19, "void" is a "reserved word".
Line: 7, 1st char: 24, "main" is a "reserved word".
Line: 7, 1st char: 28, "(" is a "symbol".
Line: 7, 1st char: 29, ")" is a "symbol".
Line: 7, 1st char: 31, "{" is a "symbol".
Line: 9, 1st char: 9, "int" is a "reserved word".
Line: 9, 1st char: 13, "c" is an "ID".
Line: 9, 1st char: 14, ";" is a "symbol".
Line: 10, 1st char: 9, "int" is a "reserved word".
Line: 10, 1st char: 13, "a" is an "ID".
Line: 10, 1st char: 15, "=" is an "operator".
Line: 10, 1st char: 17, "5" is an "integer".
Line: 10, 1st char: 18, ";" is a "symbol".
Line: 11, 1st char: 9, "c" is an "ID".
Line: 11, 1st char: 11, "=" is an "operator".
Line: 11, 1st char: 13, "add" is an "ID".
Line: 11, 1st char: 16, "(" is a "symbol".
Line: 11, 1st char: 17, "a" is an "ID".
Line: 11, 1st char: 18, "," is a "symbol"
```

```
Line: 11, 1st char: 18, ";" is a "symbol".
Line: 11, 1st char: 20, "10" is an "integer".
Line: 11, 1st char: 22, ")" is a "symbol".
Line: 11, 1st char: 23, ";" is a "symbol".
Line: 12, 1st char: 9, "if" is a "reserved word".
Line: 12, 1st char: 12, "(" is a "symbol".
Line: 12, 1st char: 13, "c" is an "ID".
Line: 12, 1st char: 15, ">" is an "operator".
Line: 12, 1st char: 17, "10" is an "integer".
Line: 12, 1st char: 19, ")" is a "symbol".
Line: 13, 1st char: 13, "print" is a "reserved word".
Line: 13, 1st char: 18, "(" is a "symbol".
Line: 13, 1st char: 19, ""c = "" is a "string".
Line: 13, 1st char: 26, "+" is an "operator".
Line: 13, 1st char: 28, "-" is an "operator".
Line: 13, 1st char: 29, "c" is an "ID".
Line: 13, 1st char: 30, ")" is a "symbol".
Line: 13, 1st char: 31, ";" is a "symbol".
Line: 14, 1st char: 9, "else" is a "reserved word".
Line: 15, 1st char: 13, "print" is a "reserved word".
Line: 15, 1st char: 18, "(" is a "symbol".
Line: 15, 1st char: 19, "c" is an "ID".
Line: 15, 1st char: 20, ")" is a "symbol".
Line: 15, 1st char: 21, ";" is a "symbol".
Line: 16, 1st char: 9, "print" is a "reserved word".
Line: 16, 1st char: 14, "(" is a "symbol".
Line: 16, 1st char: 15, ""Hello World"" is a "string".
Line: 16, 1st char: 28, ")" is a "symbol".
Line: 16, 1st char: 29, ";" is a "symbol".
Line: 18, 1st char: 5, "}" is a "symbol".
Line: 20, 1st char: 1, "}" is a "symbol".
The symbol table contains:
Test1
add
a
b
c
```

```
./demo < test2.java
Line: 1, 1st char: 1, "// this is a comment // line */ /* with /* delimiters */ before the end" is a "comment".
Line: 3, 1st char: 1, "public" is a "reserved word".
Line: 3, 1st char: 8, "class" is a "reserved word".
Line: 3, 1st char: 14, "Test2" is an "ID".
Line: 3, 1st char: 20, "{" is a "symbol".
Line: 4, 1st char: 5, "int" is a "reserved word".
Line: 4, 1st char: 9, "i" is an "ID".
Line: 4, 1st char: 11, "=" is an "operator".
Line: 4, 1st char: 13, "-100" is an "integer".
Line: 4, 1st char: 17, ";" is a "symbol".
Line: 5, 1st char: 5, "double" is a "reserved word".
Line: 5, 1st char: 12, "d" is an "ID".
Line: 5, 1st char: 14, "=" is an "operator".
Line: 5, 1st char: 16, "12.25e+6" is a "float".
Line: 5, 1st char: 24, ";" is a "symbol".
Line: 7, 1st char: 5, "public" is a "reserved word".
Line: 7, 1st char: 12, "static" is a "reserved word".
Line: 7, 1st char: 19, "void" is a "reserved word".
Line: 7, 1st char: 24, "main" is a "reserved word".
Line: 7, 1st char: 28, "(" is a "symbol".
Line: 7, 1st char: 29, ")" is a "symbol".
Line: 7, 1st char: 31, "{" is a "symbol".
Line: 8, 1st char: 1, "/* this is a comment // line with some /* and
// delimiters */" is a "comment".
Line: 10, 1st char: 5, "}" is a "symbol".
Line: 11, 1st char: 1, "}" is a "symbol".
The symbol table contains:
Test2
i
d
```

```
./demo < test3.java
Line: 2, 1st char: 1, "public" is a "reserved word".
Line: 2, 1st char: 8, "class" is a "reserved word".
Line: 2, 1st char: 14, "Test3" is an "ID".
Line: 2, 1st char: 20, "{" is a "symbol".
Line: 3, 1st char: 5, "int" is a "reserved word".
Line: 3, 1st char: 9, "A" is an "ID".
Line: 3, 1st char: 10, ";" is a "symbol".
Line: 4, 1st char: 5, "int" is a "reserved word".
Line: 4, 1st char: 9, "a" is an "ID".
Line: 5, 1st char: 5, "double" is a "reserved word".
Line: 5, 1st char: 12, "b" is an "ID".
Line: 5, 1st char: 13, ";" is a "symbol".
Line: 6, 1st char: 5, "double" is a "reserved word".
Line: 6, 1st char: 12, "A" is an "ID".
Line: 6, 1st char: 13, ";" is a "symbol".
Line: 8, 1st char: 5, "public" is a "reserved word".
Line: 8, 1st char: 12, "Test3" is an "ID".
Line: 8, 1st char: 17, "(" is a "symbol".
Line: 8, 1st char: 18, ")" is a "symbol".
Line: 8, 1st char: 20, "{" is a "symbol".
Line: 9, 1st char: 9, "a" is an "ID".
Line: 9, 1st char: 11, "=" is an "operator".
Line: 9, 1st char: 13, "1" is an "integer".
Line: 9, 1st char: 14, ";" is a "symbol".
Line: 10, 1st char: 9, "A" is an "ID".
Line: 10, 1st char: 11, "=" is an "operator".
Line: 10, 1st char: 13, "2" is an "integer".
Line: 10, 1st char: 14, ";" is a "symbol".
Line: 11, 1st char: 9, "b" is an "ID".
Line: 11, 1st char: 11, "=" is an "operator".
Line: 11, 1st char: 13, "-1.2" is a "float".
Line: 11, 1st char: 17, ";" is a "symbol".
Line: 12, 1st char: 5, "}" is a "symbol".
Line: 13, 1st char: 1, "}" is a "symbol".
The symbol table contains:
Test3
A
a
b
```