一、**Lex 版本：**lex 2.6.4

```
Terry@LAPTOP-E53TOILL ~
$ lex --version
lex 2.6.4
```

二、**作業平台：**Cygwin

三、**執行方式：**

- 透過 makefile 輸入 “make all”

```
Terry@LAPTOP-E53TOILL /home/compiler/TestFile_Lab1_2023/LexDemo
$ make all
flex B092040016.l
gcc lex.yy.c -o demo -lfl
```

- 輸入指令 ./demo < 檔案名稱

```
Terry@LAPTOP-E53TOILL /home/compiler/TestFile_Lab1_2023/LexDemo
$ ./demo < test1.java
```

四、**如何處理規格書上的問題：**

- Regular expression 處理 token

  透過正規表達式去 match 到合法的 symbols、operators、reserved words、identifiers、integer、float、string、comments…等，並在 lex rules 中去定義 match 到之後所要做的事，像是更新行數及字元數以便輸出，或是呼叫函式檢查數字溢位等，同時我也有去記錄當前是否為數字或是運算元，用來作為判斷 +/- 是正負還是運算符號的根據。

- Symbol table

  我利用 struct 定義節點的型別然後實作 linked list，並有 front、tail 指標指向 linked list 的第一個及最後一個節點，每個節點所存放的資料有要存入的 identifier（型別為 char*）、存入的 index（從 0 開始）以及指向下一個節點的指標，並在此 linked list 上實作 create()、lookup(s)、insert(s)、dump() 四個函式，一開始透過 create() 函式初始化 table，之後在遇到 identifier（ID）時，我透過呼叫 lookup(s) 函式查看是否該 ID 已存在於 symbol table 中，若 lookup(s)返回值等於 -1（ID 不存在於 table 中），則插入至 symbol table 中（透過 insert(s) 函式），最後 dump() 函式走訪 linked list 每個節點並印出 ID。

- Error message

  我有針對不合法的 ID 及 string 做錯誤訊息輸出，包括以數字開頭的變數以及字串中使用單引號及含有雙引號（”）卻沒有搭配使用 ”\”，此外我還有做數字溢位的錯誤訊息，透過創建檢查數字溢位的函式，在遇到 integer 及 float 時呼叫，此檢查也適用於換行的數字，其中 integer 數字溢位的處理還有包括八進位及十六進位。

五、**遇到的困難：**

- 實作 linked list 時遇到 segmentation fault
  在實作 linked list 中的 lookup(s) 及 insert(s) 時，不確定是不是因為創造指向節點的指標變數所存入的東西超過記憶體所配置的空間的原因進而造成 segmentation fault，後來改使用動態記憶體配置的方式，這個問題才得以解決。

- 處理有換行數字的溢位時，型別轉換問題及考慮全盤情況
  在處理有換行數字的溢位時，過程中我有遇到字元陣列轉換成 char* 型別的問題以及需要考慮太多的情況（包括換很多行、八進位及十六進位等情況）還有 windows 系統每一行結尾的格式為“\r\n”，因此在字串處理時這類問題也很容易被遺忘，後來，再全面細心考慮大部分情況並搭配使用動態記憶體配置及 strcpy 函式，這個問題才得以解決。

六、**結果：**

- Test1. java

```
Terry@LAPTOP-E53TOILL /home/compiler/TestFile_Lab1_2023/LexDemo
$ ./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 a "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 a "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 a "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 a "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 a "ID".
Line: 4, 1st char: 18, "+" is a "operator".
Line: 4, 1st char: 20, "b" is a "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 a "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 a "ID".
Line: 10, 1st char: 15, "=" is a "operator".
Line: 10, 1st char: 17, "5" is a "integer".
Line: 10, 1st char: 18, ";" is a "symbol".
```

```
Line: 11, 1st char: 9, "c" is a "ID".
Line: 11, 1st char: 11, "=" is a "operator".
Line: 11, 1st char: 13, "add" is a "ID".
Line: 11, 1st char: 16, "(" is a "symbol".
Line: 11, 1st char: 17, "a" is a "ID".
Line: 11, 1st char: 18, "," is a "symbol".
Line: 11, 1st char: 20, "10" is a "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 a "ID".
Line: 12, 1st char: 15, ">" is a "operator".
Line: 12, 1st char: 17, "10" is a "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 a "operator".
Line: 13, 1st char: 28, "-" is a "operator",but more like "sign".
Line: 13, 1st char: 29, "c" is a "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 a "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
```

- Test2. java

```
Terry@LAPTOP-E53TOILL /home/compiler/TestFile_Lab1_2023/LexDemo
$ ./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 a "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 a "ID".
Line: 4, 1st char: 11, "=" is a "operator".
Line: 4, 1st char: 13, "-100" is a "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 a "ID".
Line: 5, 1st char: 14, "=" is a "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
```

- Test3. java

```
Terry@LAPTOP-E53TOILL /home/compiler/TestFile_Lab1_2023/LexDemo
$ ./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 a "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 a "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 a "ID".
Line: 5, 1st char: 5, "double" is a "reserved word".
Line: 5, 1st char: 12, "b" is a "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 a "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 a "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 a "ID".
Line: 9, 1st char: 11, "=" is a "operator".
Line: 9, 1st char: 13, "1" is a "integer".
Line: 9, 1st char: 14, ";" is a "symbol".
Line: 10, 1st char: 9, "A" is a "ID".
Line: 10, 1st char: 11, "=" is a "operator".
Line: 10, 1st char: 13, "2" is a "integer".
Line: 10, 1st char: 14, ";" is a "symbol".
Line: 11, 1st char: 9, "b" is a "ID".
Line: 11, 1st char: 11, "=" is a "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
```