系統程式Project1

 程式作業一:Lexical Analysis實作

程式使用說明書

系級 : 資訊三乙

學號 : 10727211

姓名 : 林彥輝

開發平台 : C、C++

開發平台 : Windows 10

開發環境 : Dev C++

程式語言 : C 、 C++

選擇的組合語言 : SIC

程式使用說明

直接編譯執行(F11)會出現此畫面，兩種指令打對檔名都可順利Work



指令1與指令2的差別在於

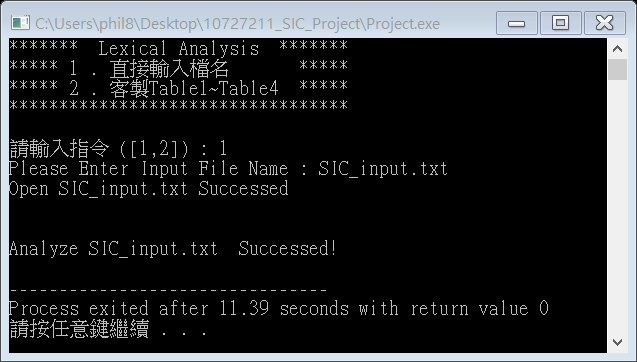
指令1 : 只需輸入input檔名 (Table1~Table4會自動根據Table1~Table4讀檔)

指令2: 需要手動輸入Table1~Table4的檔名以及input程式檔名

Table1~4的檔名

指令1示範

檔名輸入正確結果 示範如下

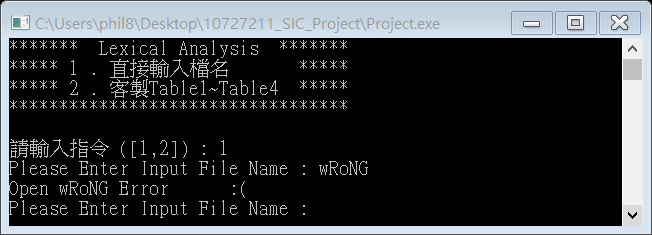


程式執行完畢後即可在 目錄下找到輸出結果

檔名為: “{輸入檔名} + \_LexialAnalyze.txt” <- 此為輸出之檔案

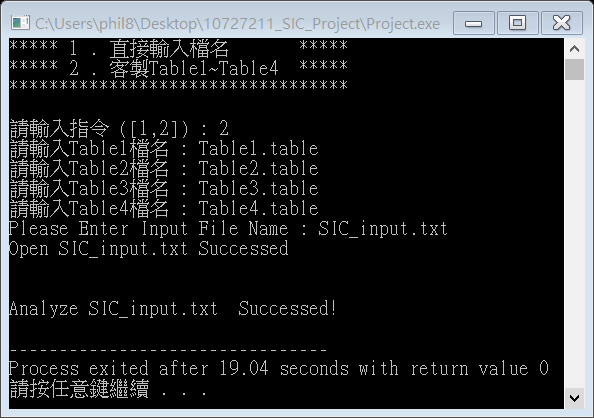


檔名輸入錯誤 會跑出以下結果 ， 此時不必關閉視窗重新打一次檔名即可



指令2示範

檔名輸入正確結果 示範如下

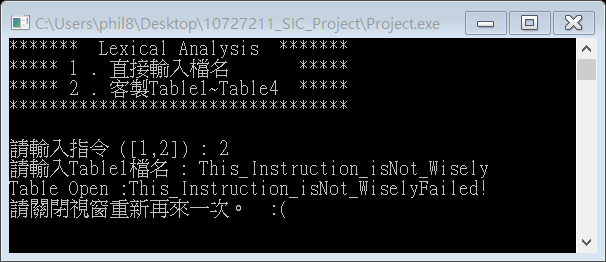


程式執行完畢後即可在 目錄下找到輸出結果

檔名為: “{輸入檔名} + \_LexialAnalyze.txt” <- 此為輸出之檔案

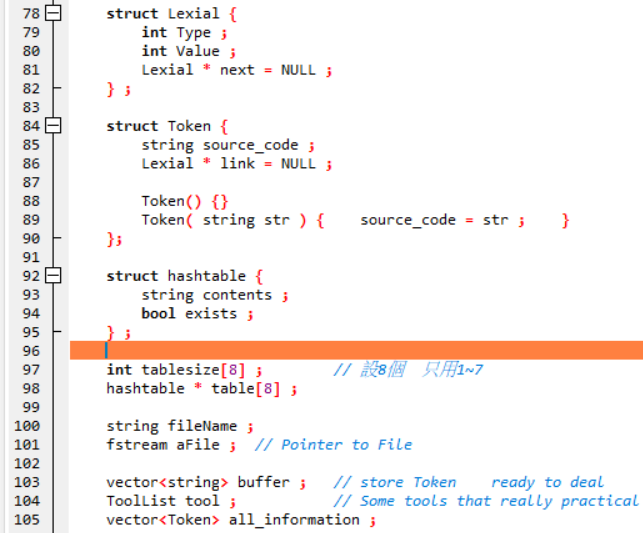


檔名輸入錯誤 會跑出以下結果



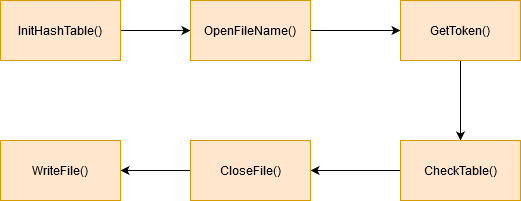
程式解釋

採用之DataBase如下

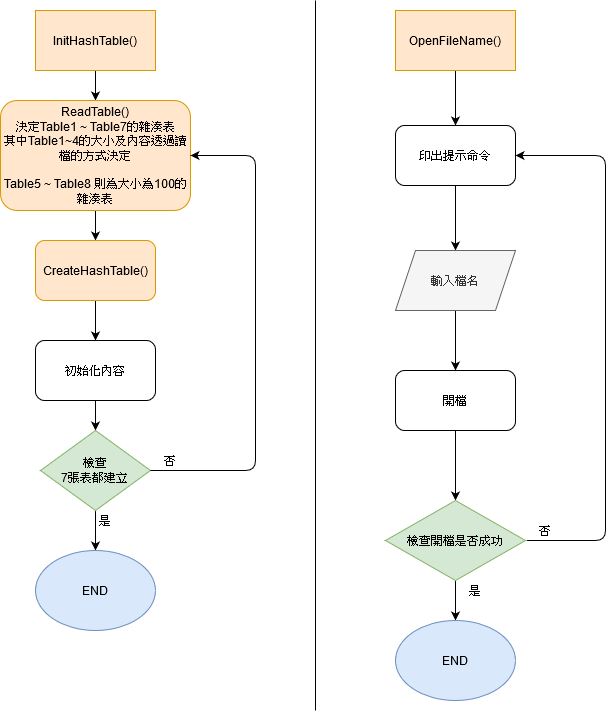


在all\_information的Vector中會儲存所有的資訊(包括source code, Token資訊)。

一個Token記錄著每一行的Source code以及link-list中紀錄每一個Token的TokenType及TokenValue。 以此基底進行程式設計。

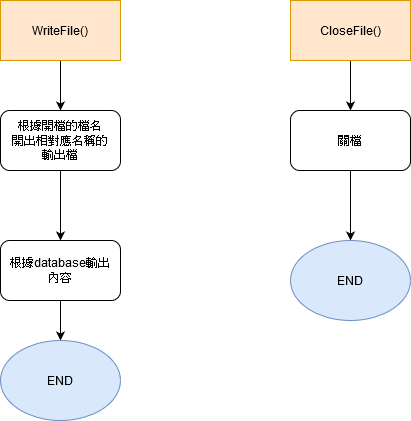


▲程式概略圖

▼InitHashTable() 初始化HashTable ▼OpenFileName()開檔

#HashTable選擇使用動態陣列

▼WriteFile() 寫檔 ▼CloseFile() 關檔



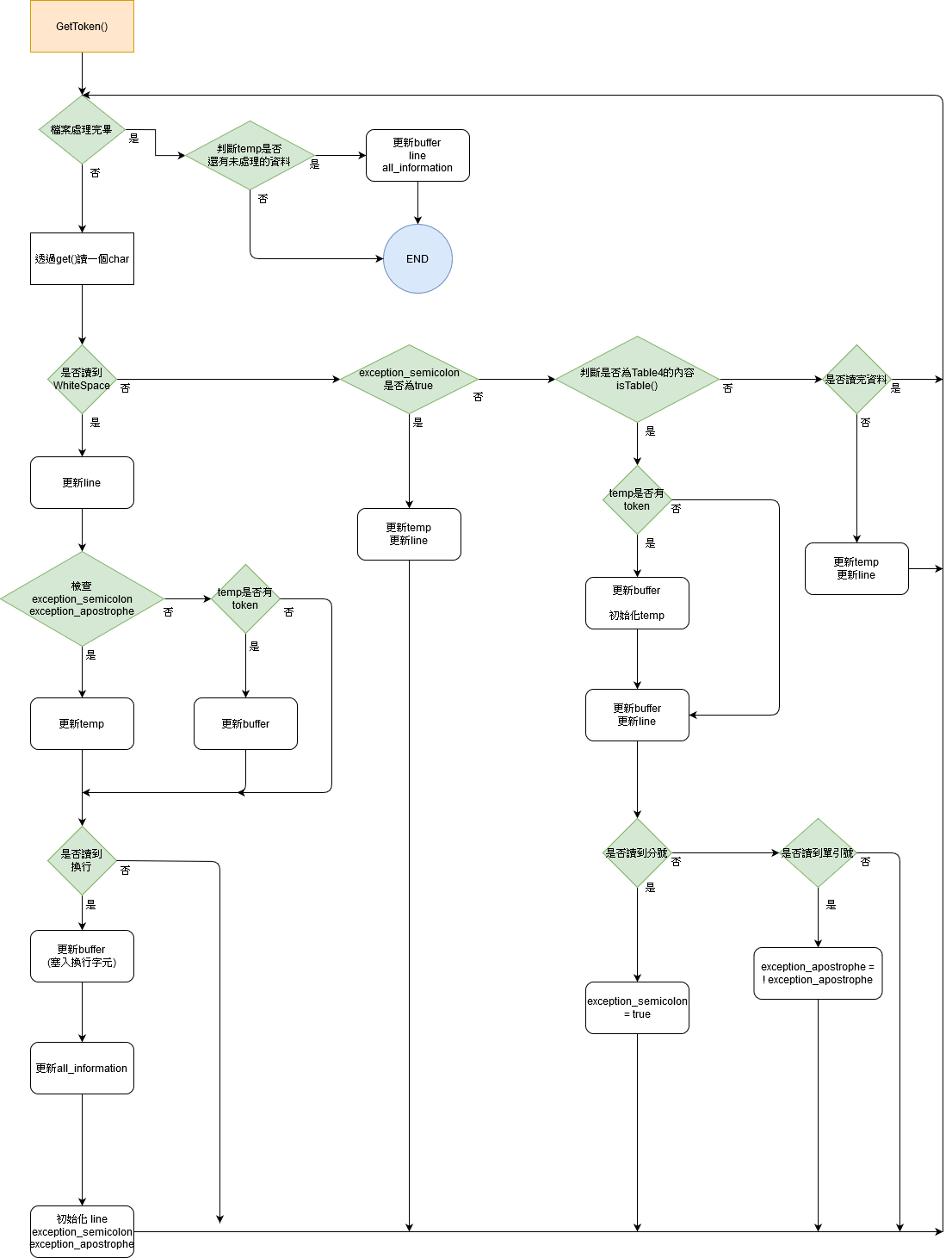
▼GetToken()

此Function負責讀取檔案的內容，並負責切Token到 存放String的Buffer中

切分時除了判斷WhiteSpace、Delimeter為基準切Token之外，為求精準度，新增了Boolean的exception\_semicolon (讀Comment用的exception)、exception\_apostrophe (允許Label 有空白的情況 如’Green ‘ 之情況)。

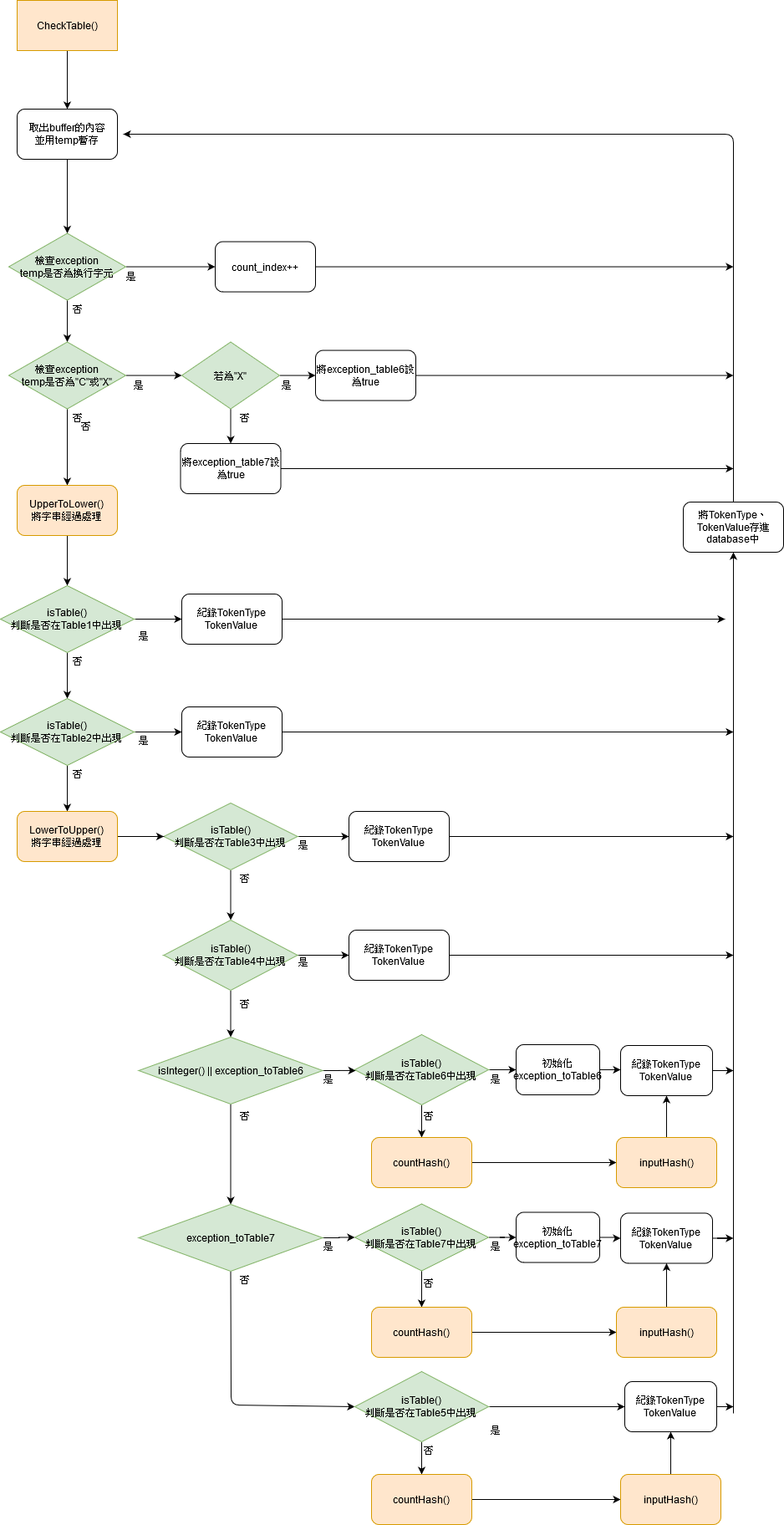
過程中也會記錄source code(透過string的line)，之後將這些line先存放進

all\_information的Vector中，至於Token部分會在CheckTable()中進行處理。



▼CheckTable()

此Function會從Buffer中提取每個Token，並將這些Token進行Table的檢查，最後放到合適之處，其中(Instruction)與 (Register)的Table會自動轉換大小寫。



程式中

1. input檔的程式碼 之 註解的內容會被記錄至source code中(題目要求) ， 但註解之後的內容 ”不”被當Token處理
2. 程式允許input檔的程式碼有排版(整段只有換行)，放寬程式碼輸入的規格
3. 程式中有關於Table1(Instruction) 與 Table3(Register) 中，

若碰到 “ADD” 、“add”、 “AdD” 會自動翻轉成”add”；

碰到”PC”、”pC”、”pc” 等會自動翻轉成”PC”。

1. 碰到string字串 允許空白字元 eg : ‘Green hi ‘，經過Token判斷Sring將為->” Green hi “
2. 程式碼的最後一個char可以不必為Enter做結尾 (程式會再幫忙檢查一次)