資工三甲10627122陳俐欣

* ***開發平台***

Ans: MacOS

* ***使用開發環境***

Ans: xcode

* ***使用的程式語言***

Ans: C++

* ***所選擇的組合語言(X86,SIC)***

Ans: 因為SIC和SICXE的指令格式相較X86較為單純固定，因此我選擇寫

SIC

* ***說明你的程式設計(功能、流程、使用的data structure)***
* 功能：
  1. 讀Input檔：手動輸入檔名(.txt不用)，並讀取完整組合語言程式碼的

檔案作為整個等待被切token的input檔，並將該input檔以行為單位放置vector<string> buffer中。

* 1. 讀Table檔：手動輸入需要的table編號起始和結尾，例如如果我們需

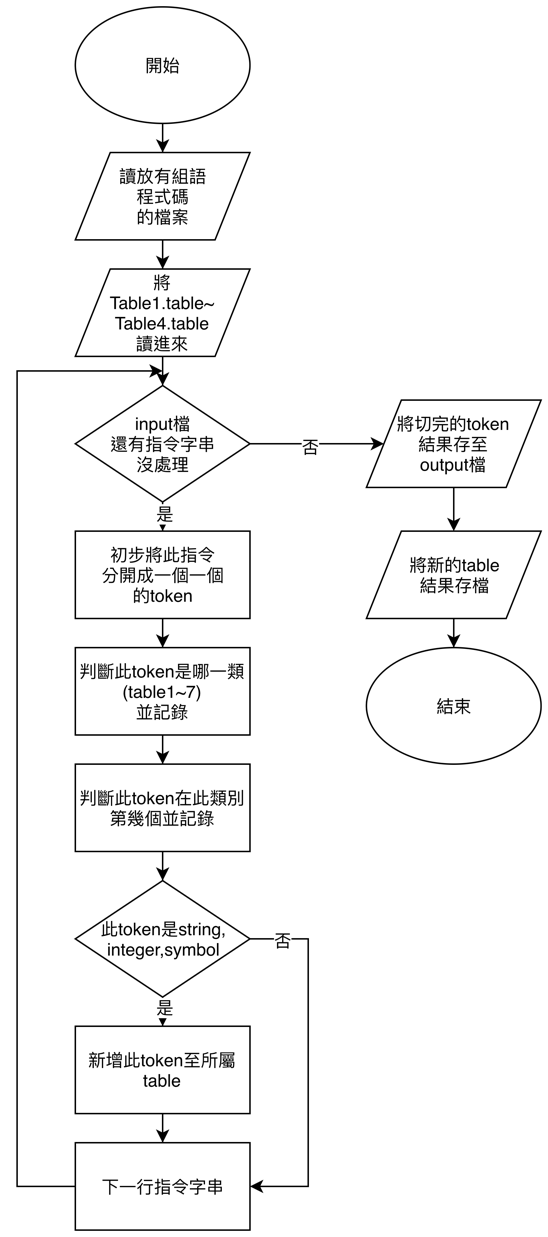
要table1到table4的資料，我們輸入“1”和“4”，程式會自動讀取“Table1.table”、“Table2.table”、 “Table3.table”、“Table4.table”4個table進來，並將每個table的資料整理(將字體都轉成小寫，清除空白和換行符號)後放進各自的buffer中，之後在找token是屬於哪一個table中時，直接從buffer中找，這樣當Table#.table中的檔案新增資料時，程式能更新判斷標準。

* 1. 切token：處理從組語程式碼的input檔讀進來的buffer，分析每一

行程式碼，將一整行字串切成一個一個的token，先切出有幾個token(先不判斷是在哪個table，除了像C‘string’這種會先賦予“string”在table7)，先將token初步分開後，再將每一行程式碼的每個token一個一個去table尋找，並將token附上table編號和在該table第幾個欄位。

* 1. 輸出token切完的結果至console
  2. 存檔：將切完token的結果存入 “原始檔名”+“\_output.txt”檔

中，檔名會依照讀入時的檔名改變；將table的資料寫檔(Table1.table ~ Table7.table)

* 流程：

(右邊流程圖)

* data structure：
  1. struct:
     1. struct Token {

string str ; // 存該token的字串

int table ; // 屬於哪個table

int no ; // 在該table的哪個欄位

} ;

* + 1. struct originStr {

int lineNo ; // 第幾行程式碼

string str ; // 程式碼字串

int numOfToken ; // 該行程式碼的token數

vector<Token> tokens ;

} ;

* + 1. struct tableInfo {

int no ; // 編號幾的table

string str ; // .table檔中的字串

} ;

* 1. class：
     1. class ProcessTable {

private:

fstream file ; // 用來開檔關檔

Tool tool ; // 小工具

vector<tableInfo> buffer\_table\_1to4[4] ;

// 放table1~4的資料

tableInfo buffer\_table\_5to7[100][3] ;

// 放table5~7的資料

public:

void findTokenTable() ; // 尋找token位置

void addToTable() ;

// 將string,integer,symbol新增至table5~7

bool readFile() ; // 讀.table檔

bool saveFile() ; // 存成.table檔

}

* + 1. class ProcessData {

private:

fstream file ; // 用來開檔讀檔

string fileName ; // 組語程式碼檔案名稱，存檔會用

vector<originInstr> buffer ; // 存組語程式碼字串

Tool tool ; // 小工具

ProcessTable processTable ; // 裡面會有table資料

void printTokens() ; // 輸出所有切完的token結果

public:

bool readFile() ; // 讀組語程式碼的input檔

void readTable() ; // 將所有.table的檔案讀進來

void getToken() ; // 切token

bool saveFile() ; // 將切完結果存檔

void printBuffer() ; // 將獨進來的組語程式碼印出

}

* + 1. class Tool {

public:

void setTableContent() ; // 將.table檔內容存起來

bool isSpace() ; // 判斷是否空白

bool isINSTR() ; // 判斷是否指令

bool isPSEUDO() ; // 判斷是否pseudo instruction

bool isREG() ; // 判斷是否暫存器

bool isDELIMITER() ; // 判斷是否delimiter

bool isINT() ; // 判斷是否數字

bool isSTR() ; // 判斷是否字串

int classificyToken() ; // 判斷此token是哪個類別

string itoa() ; // 將數字轉成字串

int transferToAscii() ; // 計算ascii碼總和

void skipSpace() ; // 跳過空白

vector<Token> splitInstr() ;

// 將整行程式碼字串的token分開

string upperToLower() ; // 將字母都轉成小寫

string lowerToUp() ; // 將字母都轉成大寫

vector<Token> parseString() ;

// 解析字串(C‘string’和x‘1234’)

}

* ***未完成的功能***
  1. table擴充性
  2. comment處理
  3. 更嚴謹的防呆措施