10727124資訊三甲 劉宇廷 Lexical Analysis : Cut Token

開發平台: Windows 10家用版 64位元作業系統 x64型處理器

使用開發環境: Dev C++

使用的程式語言: C/C++

所選擇的組合語言: x86

程式設計:

* 功能

以讀檔的方式讀入input檔(檔名為手動輸入)，完成切token的操作，利用老師提供的Table1~4.table進行opcode、pseudocode、register、delimiter的分類，其餘狀況使用sum(ASCII) % 100的hash function放入(碰撞則index+1) table5~7，依序分類label、integer、string；寫檔格式以一行source，一行為該行各token的類別(table1~7)以及在table內的序號。

* 流程

main()

一開始利用inputTable()先將table1~4讀入各自的vector中，這裡要注意從第1個位置開始放起(0不放)，接者請使用者輸入要讀檔的檔案名稱，利用inFile開始讀檔，並且把outFile也預先準備好，最後呼叫執行cutToken.Run()。

cutToken()

一開始讀進一整行(讀到‘\n’為止)，接者一個字元一個字元判斷，先判斷是否為delimiter或是空白字元(因為空白字元可以切出完整的token)，確定是delimiter就可以比對table3，但如果delimiter是註解符號(‘;’)，後面就可以不用再處理了；其餘情況還要判斷singleToken是否為空的，若裡面有東西就可切token。若非上述狀況，代表該字元確定是character，直接放入singleToken，但要注意如果該字元是整行的最後一個字元，代表也可以切token了，最後把處理完的資料寫檔到outFile中。

processing()

這是用來處理哪一種類別的函式，首先判斷的是stringTable，例如: 'Green'的情形，判斷最後一個字元是否為‘\’’，若是，前面的數字可放入stringTable；接者是記憶體位址e.g.: 1234H，判斷最後一個字元是否為‘H’，但不能是暫存器:AH、BH、CH、DH，符合後可放入integerTable；接者依序判斷opcode、pseudocode、register的table內有沒有符合的，特別注意以上三項內容是沒有分大小寫的；最後排除以上所有情況後剩下的就是label的情形了，可放入labelTable。

* 使用的data structure

// 儲存單一個token的類別、序號以及內容

typedef struct TokenData {

int type ;

int number ;

string token ;

} TokenData ;

// table內的資料利用vector串列來儲存

vector<TokenData> opCodeTable ;

vector<TokenData> pseudoTable ;

vector<TokenData> registerTable ;

vector<TokenData> delimiterTable ;

vector<TokenData> labelTable ;

vector<TokenData> integerTable ;

vector<TokenData> stringTable ;

* 未完成的功能

目前考慮到的狀況都已處理完畢。

心得:

一開始沒想太多就決定寫x86的格式了，但後來發現大部分同學都寫SIC，導致有些小bug同學無法給予太多的幫助，而且之後要寫Assembler也很頭痛，但其實x86的架構算是滿清楚明瞭的，舉例來說: 直接定址在x86只需判斷數字及‘H’字元即可，但在SIC還要注意各種情形，例如: X'F1'代表16進位的F1，C'EOF'代表EOF字串，其中‘X’及‘C’不用翻，雖然沒寫過SIC的，但感覺很麻煩。