資工三A 107502508 黃印榕

1. Linux編譯指令

bison -d -o y.tab.c code.y

gcc -c -g -I.. y.tab.c

flex -o lex.yy.c code.l

gcc -c -g -I.. lex.yy.c

gcc -o code y.tab.c lex.yy.c –ll

1. 程式設計簡述

* Lex和Yacc的語法架構參考教材＂MiniLisp.docx＂完成
* yyparse執行過程中，解析後的輸入被做成node建成樹，在non-terminal: stmt等節點再作執行。
* 通過基本題共8題和加分題第1, 2題的公開測資。

1. 程式碼功能說明

* node\* makeN(char type, int ival, char\* sval, node\* L, node\* R);
  + 將輸入解析出來的資訊做成node，供之後執行時做參考。
* node\* exeN(node\* n);
  + 根據type決定處理node的方式，回傳處理結果。
* node\* numN(node\* n);
  + 功能 + - \* / mod 在此處理後回傳NUMBER。
  + 輸入參數如果不是NUMBER會報錯。
* node\* cmpN(node\* n);
  + 功能 > < = 在此處理後回傳BOOL\_VAL。
  + 輸入參數如果不是BOOL\_VAL會報錯。
* node\* lgcN(node\* n);
  + 功能 and or not 在此處理後回傳BOOL\_VAL。
  + 輸入參數如果不是BOOL\_VAL會報錯。
* node\* ifN(node\* n);
  + 處理test\_exp後，根據結果決定回傳then\_exp或else\_exp。
  + test\_exp的結果如果不是BOOL\_VAL會報錯。
* void defDataN(node\* n);
  + 將變數名稱和對應的NUMBER或BOOL\_VAL或FUNCTION儲存。
  + 變數數量上限為256。
* node\* dataN(node\* n);
  + 根據變數取出過去儲存的NUMBER或BOOL\_VAL或FUNCTION。
* node\* exeF(node\* n);
  + 執行匿名函式。
  + 函式參數與與全域變數儲存相同地方，會在函式結束後釋放。
* node\* exeNameF(node\* n);
  + 執行命名函示。
  + 除了在最開始通過dataN取得變數名稱對應的FUNCTION外，其餘與exeF相同。