

OurC Project 4

OurC projects 的目的是(用你寫的 OurC interpreter 來) run OurC program。

Project 4 主要是測試 function 的呼叫，但 Proj.3 該有的功能也都在測試範圍內(只是沒有特別重視而已)。這包括以下的 operators:

=, +=, -=, *=, /=, %=, ? :, &&, ||, !, ==, !=, <, >, <=, >=, <<, >>, +, -, *, /, %

(由於 PAL 封鎖&|^的使用，所以&|^不在以上測試範圍內。但 comma expression 是在測試範圍內。)

error message 的部分，共有三種 error messages。其例如下：

Line 3 : unrecognized token with first char : '\$'	// lexical error
Line 2 : unexpected token : '*'	// syntactical error (token recognized)
Line 5 : undefined identifier : 'bcd'	// semantic error (grammar ok)

有鑒於 line number 的計算有時會有爭議，Project 4 的 test data 將會遵守以下兩項原則：

- (a) 一旦有 error 發生，此“error 行”(the line where the error occurs)的下一行將不會是空行，而且下一個 user input 將不會有 error。
- (b) 如果 token 是個 ID、且下一個 token 是'('，那麼此'('一定會與 ID 出現在同一行(the same line)。

至於型別錯誤(type error)的判斷，這事實上有相當多的 issues。Functions 允不允許 overload？要不要檢查參數的數目與型別？要不要檢查 return type 或該不該 return a value？Functions 能不能與 global variables 同名？這些都是重要問題。

有關「(在呼叫 function 方面)型別錯誤(type error)的判斷」的決定如下：

- (a) 不需要檢查 function 參數的型別與數目(i.e., 不會有這方面的 error)，也不需要檢查 return type 或該不該 return a value(i.e., 不會有這方面的 error)。
- (b) array 參數的宣告只能是 passed by value (void F(int a[5]))。同時，儘管是用如此方式宣告 array 參數，事實上 array 參數是 passed by reference。
- (c) 如果寫的是 void F(int a[5]) {...}, 那呼叫者呼叫 F()時一定是 pass an int array of size 5 過去。
- (d) Function 可以跟 global variables 同名(因為由「ID 之後有沒有出現'('」即可判斷 ID 是 function 名還是 global variable 名)

其他有關 type checking 的規定請參考 Proj. 3 的說明。

最後，是有關測試數據的設計(總共有 12 題)：

A. Proj. 4 不同於先前 Proj. 2 與 Proj. 3 之處是：Proj. 4 無“similar 數據”與“isomorphic 數據”之分

OurC Project 4

- B.** Prob. 1-9 的測試數據是拿 Proj. 3 的 Prob. 3 的測試數據來 extend ;
Prob. 10-11 的測試數據是拿 Proj. 3 的 Prob. 13 的測試數據來 extend ;
Prob. 12 的測試數據是拿 Proj. 3 的 Prob. 15 的測試數據來 extend

以下是各題的“測試主題”

1. primitive + global, by-value only, void + not void (int|string|bool|char)
2. array + global by-value (but actually by-ref), void + not void (int|string|bool|char)
3. array element + global, by-value only, void + not void (int|string|bool|char)
4. primitive + global, by-value and/or by-ref, void + not void (int|string|bool|char)
5. array + local/global by-value (but actually by-ref), void + not void (int|string|bool|char)
6. array element + global, by-value and/or by-ref, void + not void (int|string|bool|char)
7. value+ref+ref. as ref. + global, primitive only, void + not void (int|string|bool|char)
8. value+ref+ref. as ref. + global, primitive and/or array, void + not void (int|string|bool|char)
9. value+ref+ref. as ref. + global primitive and/or array and/or array element, void + not void (int|string|bool|char)
10. Complex function.
11. 9+10
12. 11 + error handling