Programming Assignment 2: Pascal Parser
B083040029 邱品諺

- Environment
  - Operating System
    - macOS 11.4
    - Ubuntu Linux 18.04.6
  - Lex
    - flex 2.6.4
  - Yacc
    - bison (GNU Bison) 3.0.4

- 執行方式
  - 本次作業的目的為撰寫一個 Pascal 的 Parser，簡單來說就是撰寫一個可以判斷 Pascal 語法是否正確的程式，但在判斷語法正確與否之前，需要有其他工具可以幫忙把程式中的每個 token 切出來，而這就是 Scanner 的工作，其負責每個 token 的定義方式及名稱，使得 Parser 能透過每個 token 去定義符合該程式語言的語法，最後得以判斷指定程式的語法是否正確。而本次作業執行了 Compiler 的 Lexical Analysis 以及 Syntax Analysis 的部分。

  - 我對於本作業的撰寫方式主要為：當 Parser 開始依據撰寫的文法 parsing，透過 Scanner 讀取 token 時，便利用 strcat 函式將 yytext 串接至自行定義的字串中，每當遇到換行符號便印出該行，如果有文法錯誤時，Parser 便會停止向下處理，因此需定義 error recovery 的方式使得 parser 繼續執行。此外，除了文法錯誤外，還需要在 Parser 中可能會出現錯誤訊息的文法中自行定義、撰寫判斷錯誤的方式。

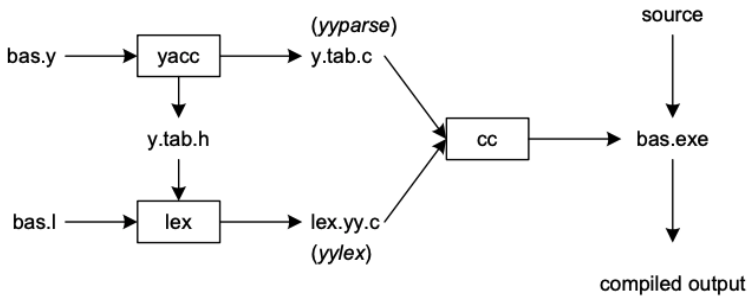
- Error Handling
  - 文法錯誤，如：缺少括弧、定義變數型態時利用 '=' 而非 ':'、缺少必要的符號，像是最後 end 沒加 Period 等等。
    - 這些錯誤其實只要在 Parser 的程式最上面 define 以下圖片中的程式碼，Parser 在 parsing 時就會回傳詳細的錯誤訊息，再自行加上行號及 parsing 至的字元數即為一則完整的錯誤訊息。

```
%define parse.error verbose
%define parse.trace
```

  - 使用未定義的變數、賦值型態與宣告型態不同、不同型態的變數相加

- 需使用 Symbol Table 紀錄每個 identifier 的變數名稱以及宣告型態，並自行定義函式提供查找功能以判斷當前遇到的變數是否有事先宣告過，並自行定義錯誤訊息。
- 本程式中有 5 個 function 來執行 Symbol Table：
  - void insert_name (char *x);
    - 將變數名稱，亦為在宣告變數時冒號左邊的 identifier 加入 Symbol Table。
  - void insert_type (int x);
    - 記錄變數的宣告型態。
  - int lookup (char *x);
    - 用以查看使用的變數是否有先宣告過，如果沒有便產生錯誤訊息。
  - void chktype (int type);
    - 用以檢查變數型態和賦值型態是否相同，如果不同便產生錯誤訊息。
  - void idctype (int x);
    - 用以標示該行當前變數的型態。利用 chktype function 和 assign (:=) 右邊的變數進行比較，檢查是否有型態不同的 expression。

- 撰寫時遇到的問題
  - Lex 和 Yacc 的執行方式
    - 由於本程式是設計在 lex 印出每一行的輸出，且利用換行符號作為印出的標示，因此最後一行需要在 Yacc 印出，也表示 Lex 和 Yacc 需要共用一個用以記錄輸出訊息的變數，而在 Yacc 宣告變數為 extern 即可，由於透過 Yacc 會先編譯出 y.tab.h，而 Lex 只需 include 這個 header，即可共同使用一個變數。



  - reduce-reduce conflict
    - 由於 PDF 提供的文法為簡易版的 Pascal BNF Syntax，所以需要再基於該文件自行增加一些文法，而增加的過程中容易遇到 conflict，因此需要想清楚在 Parser 時是如何執行的。

- Output

```
aryenchiu@ubuntu:~/Desktop/hw2_yacc/b083040029$ ./parser < 1.pas
line 1: program test;
line 2: var
line 4:    i: integer;
line 5:    j: integer;
line 6:    ans: array[0 .. 10] of integer;
line 7: begin
line 8:      i := -1+3;
line 9:      j := +7*8;
line 10:      ans[0] := 7;
line 15:      for i:=1 to 9 do
line 16:      begin
line 17:          for j:=1 to i do
line 18:              write(i*j);
line 19:      end;
line 20: end.
```

```
aryenchiu@ubuntu:~/Desktop/hw2_yacc/b083040029$ ./parser < 2.pas
line 1: program test;
line 2: var
line 3:    i: integer;
line 4: begin
line 5:    i := 3;
Line 6: at char 9, "j" is not defined.
Line 7: at char 17, Wrong type expression between "integer" and "string".
line 8:    write(i);
line 9: end.
```

RP 為 Right Parenthesis、右括弧、’)’

```
aryenchiu@ubuntu:~/Desktop/hw2_yacc/b083040029$ ./parser < 3.pas
line 1: program test;
line 2: var
line 3:    i, j : integer;
line 4: begin
line 5:    i := 20;
line 6:    j := i-7;
Line 7: at char 16, syntax error, unexpected THEN, expecting RP.
line 8:      Write('ok');
line 9: end.
```

R_BEGIN 為保留字 begin，SEMICOLON 為分號、’;’

```
aryenchiu@ubuntu:~/Desktop/hw2_yacc/b083040029$ ./parser < 4.pas
line 1: program test;
line 2: var
line 3:   i : integer
Line 4: at char 5, syntax error, unexpected R_BEGIN, expecting SEMICOLON.
line 5:   i := 5;
line 6: end.
```

PLUS 為加號、’+’，ASSIGN 為’:=’

```
aryenchiu@ubuntu:~/Desktop/hw2_yacc/b083040029$ ./parser < 5.pas
line 1: program test;
line 2: var
line 3:   i : integer;
line 4:   c : string;
line 5: begin
line 6:   i := 2020;
line 7:   c := 'compiler';
Line 8: at char 4, syntax error, unexpected PLUS, expecting ASSIGN.
Line 8: at char 6, Wrong type expression between "integer" and "string".
line 9: end.
```

```
aryenchiu@ubuntu:~/Desktop/hw2_yacc/b083040029$ ./parser < correct.pas
line 1: program test;
line 2: var
line 4:   i, j: integer;
line 5:   ans: array[0 .. 81] of integer;
line 6: begin
line 7:     i := -1+3;
line 8:     j := +7*8;
line 9:     ans[0] := 7;
line 14:     for i:=1 to 9 do
line 15:     begin
line 16:         for j:=1 to i do
line 17:             ans[i*9+j] := i*j;
line 18:     end;
line 20:     for i:=1 to 9 do
line 21:     begin
line 22:         for j:=1 to i do
line 23:             if ( ans[i*9+j] mod 2 = 0) then
line 24:                 write(i, '*', j, '=', ans[i*9+j], ' ');
line 25:         writeln;
line 26:     end;
line 27: end.
```

RELOP 為 Relation Operator，本 pascal 檔為’=’，ASSIGN 為’:=’

```
aryenchiu@ubuntu:~/Desktop/hw2_yacc/b083040029$ ./parser < error1.pas
line 1: program test;
line 2: var
line 3:   i: integer;
line 4: begin
Line 5: at char 5, syntax error, unexpected RELOP, expecting ASSIGN.
Line 6: at char 5, "j" is not defined.
Line 6: at char 5, syntax error, unexpected RELOP, expecting ASSIGN.
Line 7: at char 12, "j" is not defined.
line 8:     Write('ok');
line 9: end.
```

```
aryenchiu@ubuntu:~/Desktop/hw2_yacc/b083040029$ ./parser < error2.pas
line 1: program test;
line 2: var
line 3:   i, j : integer;
line 4: begin
line 5:   i := 5*2;
line 6:   j := 9;
line 7:   if (i > j) then
line 8:     Write('ok');
line 9: end.
```

COLON 為冒號、’:’，COMMA 為逗號、’,‘，PERIOD 為句點、’.‘

```
aryenchiu@ubuntu:~/Desktop/hw2_yacc/b083040029$ ./parser < error3.pas
line 1: program test;
line 2: var
Line 3: at char 9, syntax error, unexpected ASSIGN, expecting COLON or COMMA.
line 4: begin
line 5:   i := 5;
Line 6: at char 3, syntax error, unexpected $end, expecting PERIOD.
```

RELOP 為 Relation Operator，本 pascal 檔為’=’，ASSIGN 為’:=’

```
aryenchiu@ubuntu:~/Desktop/hw2_yacc/b083040029$ ./parser < error4.pas
line 1: program test;
line 2: var
line 3:   i, j : integer;
line 4:   c : string;
line 5: begin
line 6:   i := 5;
line 7:   c := 'aa';
Line 8: at char 5, syntax error, unexpected RELOP, expecting ASSIGN.
Line 8: at char 10, Wrong type expression between "integer" and "string".
line 9: end.
```