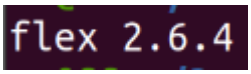
【README】

B083040010 李其儒

1. Lex, 版本：flex 2.6.4 （ 

）
2. 作業平台：Ubuntu 20.04 (Linux)
3. 執行方式：製作 makefile，如圖：

（
```
1 FILE_lex=       B083040010.l
2 PROG_lex=       lex.yy.c
3 all:    $(PROG_lex)
4         gcc $(PROG_lex) -lfl
5
6 $(PROG_lex):    $(FILE_lex)
7         flex $(FILE_lex)
8
9 clean:
10        rm a.out $(PROG_lex)
```
）

執行步驟：

I. 進入相關資料夾，並“make”。

II. “./a.out < (要測試的檔案)”

4. 你/妳如何處理這份規格書上的問題：

I. **Reserved words**：將規格書上附的「保留字列表」，以”|(or)”連接即可。要注意的是因 Pascal 是 case-insensitive，所以我在上方加了”%option caseless”，使其不論大小寫，一視同仁。（如下圖）

```
%option caseless
reserved absolute|and|begin|break|case|const|continue|do|else|end|for|function|if|mod|nil|not|object|of|or|program|-
then|to|var|while|array|integer|double|write|writeln|string|float|read|array|integer|double|write|writeln|string|-
float

{reserved} {
        printf("Line: %d, 1st char: %d, \"%s\" is a \"reserved word\".\n", lineCount, charCount, yytext);
        charCount += yyleng;
}
```

II. **Identifiers**：分成合法的 identifiers 及非法的 identifiers

兩項 token 來判斷－

I. **合法 id**：第一個字需是 a-z 或_；之後的字元可以是 a-z 或 0-9 或_。最後使用{0,14}來判斷其長度，超過 15 字則非法。

II. **非法 id**：分三種情況 1）開頭為非合法 id 之開頭(猜測)；2）為合法 id 之形式，但超過 15 個字元；3）合法開頭字元，但其後有非法 id 字元。

```
identifiers [a-z|_](?:[a-z0-9_]{0,14})
invalid_identifiers [0-9|^|#|$|@|!|`|%|^|&|/][0-9a-z_|^|#|$|@|!|`|%|^|&|/]*|[a-z|_][0-9a-z_]{15,}|[a-z_][a-z0-9#$@!-
`%^&/]*[#$@!`%^&/][a-z0-9#$@!`%^&/]*

{invalid_identifiers} {
        printf("Line: %d, 1st char: %d, \"%s\" is an invalid \"ID\".\n", lineCount, charCount, yytext);
        charCount += yyleng;
}
{identifiers} {
        printf("Line: %d, 1st char: %d, \"%s\" is an \"ID\".\n", lineCount, charCount, yytext);
        charCount += yyleng;
}
```

III. **Symbols**：純將所給的 symbol list 丟入，並用|(or)隔開即可。其後遇到兩個問題－

I. 發現 2.pas 測資，會將:=判斷成:及=兩 symbol。解決：建立對應 token，將其丟入 symbol list 即可。

II. 發現 5.pas 測資，會有“,”之可能 symbol 未判斷到。解決：將其丟入 symbol list（不知這樣是否合理，因其未在規格書給的 symbol list 上)。

```
a :=
b ==
c <=
d >=
symbol {a}|{b}|{c}|{d}|[;|\(|\)|:|>|<|=|\[|\]|+|-|*|/|.|,]
```

```
{symbol} {
        printf("Line: %d, 1st char: %d, \"%s\" is a \"symbol\".\n", lineCount, charCount, yytext);
        charCount += yyleng;
}
```

IV. **Real constant**：分成合法的 real constant 及非法的 real constant 兩項 token 來判斷－

I. **合法 R**：分 3 part 來討論，1. 有無-(負號) 2. 小數點前後的數字形式(xx. xx) 3. 科學記號(Ee +- 0-9)。依此規則判斷為合法。

II. **非法 R**：分多個 part，但核心概念是：1. 任何位置有 00 出現 2. 小數點前後任一邊沒數字 3. 0 開頭，且其後緊接著數字。上述概念即判斷為非法。

```
valid_R -?(?:[0-9]+[.]?[0-9]*)(?:[Ee][+|-][1-9]+)?
invalid_R [0][0][.][0-9]+|[0-9]+[.][0-9]*[0][0]|[0][0-9]+[.][0-9+]|[.][0-9]+|[0-9]+[.]|[0][0][0-9]*

{invalid_R} {
        printf("Line: %d, 1st char: %d, \"%s\" is a invalid \"real constant\".\n", lineCount, charCount, yytext);
        charCount += yyleng;
}
{valid_R} {
        printf("Line: %d, 1st char: %d, \"%s\" is an valid \"real constant\".\n", lineCount, charCount, yytext);
        charCount += yyleng;
}
```

V. **Quoted string**：分成合法的 quoted string 及單邊的非法 quoted string 兩項 token 來判斷－

valid 的設計，原則上只需注意其字串長度不能超過 30。
重點在於 invalid 的判斷，事實上除了單邊的 quote、字串長度超過 30 為非法外，我不是很清楚還有哪些情況會是非法的，因此此部分較含糊，希望 demo 的時候能放寬一點，或給我解釋的機會。

```
quoted_string '([^'\n]|''){0,30}'
invalid_quoted_string ('([^'\n; ])*)|([^'\n; ]*')|('([^'\n]|''){31,}')

{quoted_string} {
        printf("Line: %d, 1st char: %d, \"%s\" is a valid \"string\".\n", lineCount, charCount, yytext);
        charCount += yyleng;
}
{invalid_quoted_string} {
        printf("Line: %d, 1st char: %d, \"%s\" is an invalid \"string\".\n", lineCount, charCount, yytext);
        charCount += yyleng;
}
```

VI. **Comment**：分成合法的 comment(並透過判斷式判斷一些非

法 comment)及單邊的非法 comment 兩項 token 來判斷－

comment 設計為”將整個(**)”都抓出來，並使用 if 條件式，判斷若(**)中出現了*)，即為非法 comment，否則為合法 comment；另外設計了判斷只出現單邊(*或*)的 token，來判斷非法 comment。

```
comment \(\*([^\(]*)\*\)
one_side_comment (\(\*[^\n \*\);]*)|([^\n \(\*]*\*\))
```

```
{comment} {
        int i, flag = 0;
        for (i = 0; i <= yyleng-3; i++)
        {
                if (yytext[i] == '*' && yytext[i+1] == ')')
                        flag = 1;
        }
        if (flag == 1)
                printf("Line: %d, 1st char: %d, \"%s\" is an invalid \"comment\".\n", lineCount, charCount,
yytext);
        else
                printf("Line: %d, 1st char: %d, \"%s\" is a valid \"comment\".\n", lineCount, charCount, yytext);
        charCount += yyleng;
}
{one_side_comment} {
        printf("Line: %d, 1st char: %d, \"%s\" is an invalid \"comment\".\n", lineCount, charCount, yytext);
        charCount += yyleng;
}
```

VII. **Error recovery**：output 出哪些為非法的字(error message)，供使用者判讀。

5. 你/妳寫這個作業所遇到的問題

I. case-insensitive 的寫法。判斷 6.pas 時發現了這個問題，上網查了好多方法，例如將所有要判斷的字元轉成 lower-case……之後才使用 option caseless 的方法來規避。

II. symbol 判斷問題。上面有提到，我在測試 2.pas 時，發現:=被判斷成:及=兩 symbol。後來透過設計:=等 token，並將其丟入 symbol list 才解決。

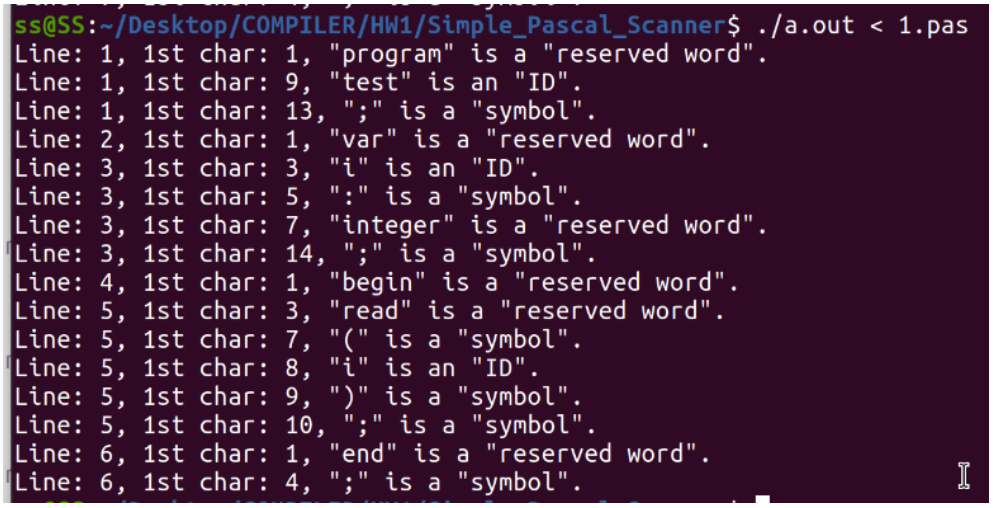
III. 發現 5.pas 測資，會有 “,” 之可能 symbol 未判斷到。解決：將其丟入 symbol list（不知這樣是否合理，因其未在規格書給的 symbol list 上）。

IV. 測資的不明確。規格書上給的 valid/invalid 測資範例不太明確，會讓人不清楚要如何設計 invalid 的 Regex，導致我們一直問助教...

V. 不清楚隱藏測資的複雜度。與同學想了很多種怪異測資，但若要真的全部判斷出來，會弄得非常非常複雜。很難去判斷自己寫的東西，足不足夠應付助教的隱藏測資。

6. 所有測試檔執行出來的結果，存成圖片或文字檔

I. 1.pas：



II. 2.pas：

```
ss@SS:~/Desktop/COMPILER/HW1/Simple_Pascal_Scanner$ ./a.out < 2.pas
Line: 1, 1st char: 1, "program" is a "reserved word".
Line: 1, 1st char: 9, "test" is an "ID".
Line: 1, 1st char: 13, ";" is a "symbol".
Line: 2, 1st char: 1, "var" is a "reserved word".
Line: 3, 1st char: 3, "3i" is an invalid "ID".
Line: 3, 1st char: 6, ":" is a "symbol".
Line: 3, 1st char: 8, "string" is a "reserved word".
Line: 3, 1st char: 14, ";" is a "symbol".
Line: 4, 1st char: 1, "begin" is a "reserved word".
Line: 5, 1st char: 3, "3i" is an invalid "ID".
Line: 5, 1st char: 6, ":=" is a "symbol".
Line: 5, 1st char: 9, "'ab" is an invalid "string".
Line: 5, 1st char: 12, ";" is a "symbol".
Line: 6, 1st char: 1, "end" is a "reserved word".
Line: 6, 1st char: 4, ";" is a "symbol".
```

III. 3.pas：

```
ss@SS:~/Desktop/COMPILER/HW1/Simple_Pascal_Scanner$ ./a.out < 3.pas
Line: 1, 1st char: 1, "(* comment 1
   comment 2 *)" is a valid "comment".
Line: 2, 1st char: 1, "program" is a "reserved word".
Line: 2, 1st char: 9, "test" is an "ID".
Line: 2, 1st char: 13, ";" is a "symbol".
Line: 3, 1st char: 1, "var" is a "reserved word".
Line: 4, 1st char: 3, "i" is an "ID".
Line: 4, 1st char: 5, ":" is a "symbol".
Line: 4, 1st char: 7, "integer" is a "reserved word".
Line: 4, 1st char: 14, ";" is a "symbol".
Line: 5, 1st char: 1, "begin" is a "reserved word".
Line: 6, 1st char: 3, "read" is a "reserved word".
Line: 6, 1st char: 7, "(" is a "symbol".
Line: 6, 1st char: 8, "i" is an "ID".
Line: 6, 1st char: 9, ")" is a "symbol".
Line: 6, 1st char: 10, ";" is a "symbol".
Line: 7, 1st char: 1, "end" is a "reserved word".
Line: 7, 1st char: 4, ";" is a "symbol".
```

IV. 4.pas：

```
ss@SS:~/Desktop/COMPILER/HW1/Simple_Pascal_Scanner$ ./a.out < 4.pas
Line: 1, 1st char: 1, "program" is a "reserved word".
Line: 1, 1st char: 9, "test" is an "ID".
Line: 1, 1st char: 13, ";" is a "symbol".
Line: 2, 1st char: 1, "var" is a "reserved word".
Line: 3, 1st char: 3, "f" is an "ID".
Line: 3, 1st char: 5, ":" is a "symbol".
Line: 3, 1st char: 7, "float" is a "reserved word".
Line: 3, 1st char: 12, ";" is a "symbol".
Line: 4, 1st char: 1, "begin" is a "reserved word".
Line: 5, 1st char: 3, "f" is an "ID".
Line: 5, 1st char: 5, ":=" is a "symbol".
Line: 5, 1st char: 8, "12.25e+6" is an valid "real constant".
Line: 5, 1st char: 16, ";" is a "symbol".
Line: 6, 1st char: 1, "end" is a "reserved word".
Line: 6, 1st char: 4, ";" is a "symbol".
```

V. 5.pas：

```
ss@SS:~/Desktop/COMPILER/HW1/Simple_Pascal_Scanner$ ./a.out < 5.pas
Line: 1, 1st char: 1, "(* a**b) *)" is a valid "comment".
Line: 2, 1st char: 1, "program" is a "reserved word".
Line: 2, 1st char: 9, "test" is an "ID".
Line: 2, 1st char: 13, ";" is a "symbol".
Line: 3, 1st char: 1, "var" is a "reserved word".
Line: 4, 1st char: 3, "i" is an "ID".
Line: 4, 1st char: 5, ":" is a "symbol".
Line: 4, 1st char: 7, "integer" is a "reserved word".
Line: 4, 1st char: 14, ";" is a "symbol".
Line: 5, 1st char: 3, "_s" is an "ID".
Line: 5, 1st char: 5, "," is a "symbol".
Line: 5, 1st char: 7, "_s2" is an "ID".
Line: 5, 1st char: 10, "," is a "symbol".
Line: 5, 1st char: 12, "_s3" is an "ID".
Line: 5, 1st char: 15, "," is a "symbol".
Line: 5, 1st char: 17, "_s4" is an "ID".
Line: 5, 1st char: 20, "," is a "symbol".
Line: 5, 1st char: 22, "_s5" is an "ID".
Line: 5, 1st char: 26, ":" is a "symbol".
Line: 5, 1st char: 28, "string" is a "reserved word".
Line: 5, 1st char: 34, ";" is a "symbol".
Line: 6, 1st char: 1, "begin" is a "reserved word".
Line: 7, 1st char: 3, "i" is an "ID".
Line: 7, 1st char: 5, ":=" is a "symbol".
Line: 7, 1st char: 8, "-100" is an valid "real constant".
Line: 7, 1st char: 12, ";" is a "symbol".
Line: 8, 1st char: 3, "_s" is an "ID".
Line: 8, 1st char: 6, ":=" is a "symbol".
Line: 8, 1st char: 9, "'db lab'" is a valid "string".
Line: 8, 1st char: 17, ";" is a "symbol".
Line: 9, 1st char: 3, "_s2" is an "ID".
Line: 9, 1st char: 7, ":=" is a "symbol".
Line: 9, 1st char: 10, "'You''ll see'" is a valid "string".
Line: 9, 1st char: 23, ";" is a "symbol".
Line: 10, 1st char: 3, "_s3" is an "ID".
Line: 10, 1st char: 7, ":=" is a "symbol".
Line: 10, 1st char: 10, "''" is a valid "string".
Line: 10, 1st char: 12, ";" is a "symbol".
Line: 11, 1st char: 3, "_s4" is an "ID".
Line: 11, 1st char: 7, ":=" is a "symbol".
Line: 11, 1st char: 10, "''''" is a valid "string".
Line: 11, 1st char: 14, ";" is a "symbol".
Line: 12, 1st char: 3, "_s5" is an "ID".
Line: 12, 1st char: 7, ":=" is a "symbol".
Line: 12, 1st char: 10, "' '" is a valid "string".
Line: 12, 1st char: 13, ";" is a "symbol".
Line: 13, 1st char: 1, "end" is a "reserved word".
Line: 13, 1st char: 4, ";" is a "symbol".
```

VI. 6.pas：

```
ss@SS:~/Desktop/COMPILER/HW1/Simple_Pascal_Scanner$ ./a.out < 6.pas
Line: 1, 1st char: 1, "ProGram" is a "reserved word".
Line: 1, 1st char: 9, "test" is an "ID".
Line: 1, 1st char: 13, ";" is a "symbol".
Line: 2, 1st char: 1, "var" is a "reserved word".
Line: 3, 1st char: 3, "#db" is an invalid "ID".
Line: 3, 1st char: 7, ":" is a "symbol".
Line: 3, 1st char: 9, "float" is a "reserved word".
Line: 3, 1st char: 14, ";" is a "symbol".
Line: 4, 1st char: 3, "_f2" is an "ID".
Line: 4, 1st char: 7, ":" is a "symbol".
Line: 4, 1st char: 9, "float" is a "reserved word".
Line: 4, 1st char: 14, ";" is a "symbol".
Line: 5, 1st char: 1, "begin" is a "reserved word".
Line: 6, 1st char: 3, "#db" is an invalid "ID".
Line: 6, 1st char: 7, ":=" is a "symbol".
Line: 6, 1st char: 10, ".1" is a invalid "real constant".
Line: 6, 1st char: 12, ";" is a "symbol".
Line: 7, 1st char: 3, "_f2" is an "ID".
Line: 7, 1st char: 7, ":=" is a "symbol".
Line: 7, 1st char: 10, "12.100" is a invalid "real constant".
Line: 7, 1st char: 16, ";" is a "symbol".
Line: 8, 1st char: 1, "end" is a "reserved word".
Line: 8, 1st char: 4, ";" is a "symbol".
```

VII. 7.pas：

```
ss@SS:~/Desktop/COMPILER/HW1/Simple_Pascal_Scanner$ ./a.out < 7.pas
Line: 1, 1st char: 1, "(* This line is a comment. *)" is a valid "comment".
Line: 2, 1st char: 1, "program" is a "reserved word".
Line: 2, 1st char: 9, "test" is an "ID".
Line: 2, 1st char: 13, ";" is a "symbol".
Line: 3, 1st char: 1, "var" is a "reserved word".
Line: 4, 1st char: 3, "i" is an "ID".
Line: 4, 1st char: 5, ":" is a "symbol".
Line: 4, 1st char: 7, "integer" is a "reserved word".
Line: 4, 1st char: 14, ";" is a "symbol".
Line: 5, 1st char: 1, "begin" is a "reserved word".
Line: 6, 1st char: 3, "i" is an "ID".
Line: 6, 1st char: 5, ":=" is a "symbol".
Line: 6, 1st char: 8, "1" is an valid "real constant".
Line: 6, 1st char: 9, "+" is a "symbol".
Line: 6, 1st char: 10, "2" is an valid "real constant".
Line: 6, 1st char: 11, ";" is a "symbol".
Line: 7, 1st char: 1, "end" is a "reserved word".
Line: 7, 1st char: 4, ";" is a "symbol".
```