112-2 系統程式期末報告

個人

1102966 邱翊銨

YACC (Yet Another Compiler-Compiler)

1. YACC 格式介紹

Definitions

- 定義或宣告
  - 通常在整個檔案的最頂端，匯入資料庫、標頭檔、定義全域變數或函式 (C code)
  - 需要用 '%{' 與 '%}' 包起來
  - 例：

```
%{
#include <stdio.h>
#include <stdlib.h>
#include "main.h"
void yyerror(const char *s);
int yylex(void);
%}

%token NUMBER
```

```
%{
#include <stdio.h>
#include <stdlib.h>

/* 全域變數定義 */
int result;

/* 函數聲明 */
void yyerror(const char *s);
int yylex(void);
%}
```

- 標記 (token) 等等的定義

  預設會個別自行產生對應的值（如 ASCII 字元對應值），定義詞法分析器（Lex）中會用到的：

  a. %token <自訂型別> name (terminal)
  b. %type <自訂型別> name (non-terminal)
  c. %left, %right, %nonassoc (eg: '='): 左、右、不 結合
     - %left 舉例：'+', '-', '*', '/' 由左向右計算
     - %right 舉例：'=', '^' 賦值與指數運算由右向左結合
     - %nonassoc UMINUS：處理一元減號（負號），避免衝突；'>', '<' 也通常在此被定義
     - 越後定義的優先度越高
     - %nonassoc > %left = %right
  d. %start typename: 從此開始分析語法

- YYSTYPE (自訂型別) 的定義

  在進行標記時，可以用來設定 yylval 的型別: by union

範例 1

```
%token NUMBER
```

範例 2

```
%token NUMBER
%token PLUS MINUS MULTIPLY DIVIDE
%token LPAREN RPAREN

%left PLUS MINUS
%left MULTIPLY DIVIDE

%start expr
```

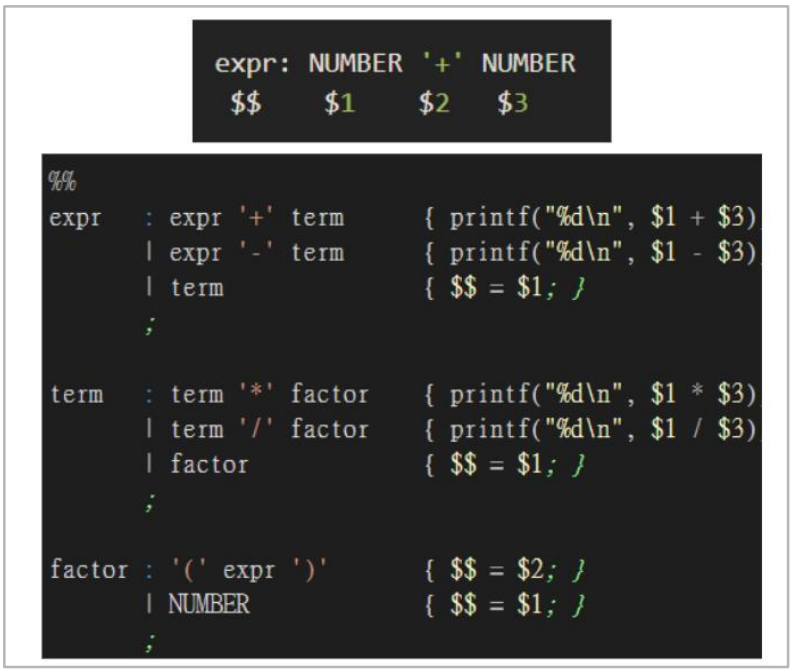
範例 3

```
%union {
    int ival;
    float fval;
    char *sval;
}

%token <ival> NUMBER
%type  <ival> expr term factor
```

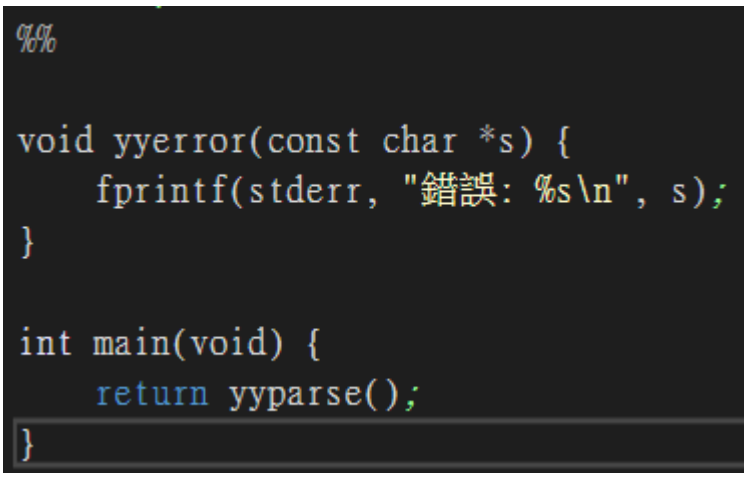
Rules

- 使用者自定義 YACC 的語法規則
- 描述了如何解析輸入的結構及對應的動作 (C code)，依此產生語法分析器 (LR)
- 盡量在解析到規則的結尾後才執行對應的動作



Subroutines

如果不想要作 error handling，就不用寫 yyerror (記得%%還是要加)，前面也不用作 definition



YACC 實作

```
%{
#include <stdio.h>
#include <stdlib.h>

void yyerror(const char *s);
int yylex(void);
%}

%token NUMBER

%%
expr    : expr '+' term      { printf("%d\n", $1 + $3); }
        | expr '-' term      { printf("%d\n", $1 - $3); }
        | term               { $$ = $1; }
        ;

term    : term '*' factor    { printf("%d\n", $1 * $3); }
        | term '/' factor    { printf("%d\n", $1 / $3); }
        | factor             { $$ = $1; }
        ;

factor : '(' expr ')'        { $$ = $2; }
        | NUMBER             { $$ = $1; }
        ;
%%

void yyerror(const char *s) {
    fprintf(stderr, "錯誤: %s\n", s);
}

int main(void) {
    return yyparse();
}
```

Definitions

- 導入函式庫和標頭檔
- 定義 yyerror 和 yylex 函數

標記 (token) 等等的定義

- 使用 %token 定義了一個記號 NUMBER，表示數字。
- NUMBER 將由詞法分析器（即 yylex 函數）返回。

Rules

- 解析並計算簡單的算術表達式，例如 1 + 2 * (3 - 4)。
- 規則定義

  expr term factor
- 這些規則定義了算術表達式的語法結構，並且在匹配到相應的語法結構時，計算結果並印出來。

Subroutines

- yyerror 函數 用於錯誤處理，在語法錯誤發生時印出錯誤訊息。

編譯與執行

- 使用 bison 產生 calc.y

  bison -d calc.y
- 使用 gcc 編譯 calc 生成的 C 原始碼，產生可執行文件

  gcc -o calc calc.tab.c
- 產生的可執行檔：

  calc.exe

1. YACC 參考資料

    a. Lex & Yacc (epaperpress.com)

    b. Lex & Yacc 學習筆記 by BarleyTea (wchsutw)

    c. 使用巴科斯範式(BNF/EBNF/ABNF)定義語言 - HackMD