calc.y

calc.y - Notepadqq

File Edit Search View Encoding Language Settings Run Window ?

calc.y

```
double dval;
};
%token <dval> NUMBER
%left '-' '+'
%left '*' '/'
%nonassoc UMINUS
%type <dval> expression
%%
lines
    : {/*empty string*/}
    | lines expression '\n'  { printf(PURPLE"answer = %lf\n"NONE,$2);}
    | error '\n'{/*when syntax error occurs, yacc skips every token until it recognizes the nex

expression: expression '+' expression {$$ = $1+$3;/* $N references the value of token or type, I
          | expression '-' expression {$$ = $1-$3;}
          | expression '*' expression {$$ = $1*$3;}
          | expression '/' expression {
                    if($3 == 0)
                        yyerror("divide by zero");
                    else
                        $$ = $1/$3;
                }
          | '-' expression %prec UMINUS {$$ = -$2;}
          | '(' expression ')'  {$$ = $2;}
          | NUMBER {$$ = $1;}
%%
int main(){
    yyparse();
    return 0;
```

Text Ln 1, col 1 Sel 0 (1) 2107 chars, 61 lines UNIX / OS X UTF-8 w/o BOM INS

Open test... ~/Des... Save

```
1+5*6-7
1+5
```

Tab Width: 8 Ln 1, Col 1 INS

chin@ubuntu: ~/Desktop/calc

File Edit View Search Terminal Help

```
chin@ubuntu:~/Desktop/calc$ ./a.out < testfile
answer = 24.000000
answer = 6.000000
chin@ubuntu:~/Desktop/calc$
```

TA：林晉廷 2019.5.20

Grammar Loop(Pascal)

```
%%
prog:
    PROGRAM prog_name ';' VAR dec_list ';' Begin stmt_list ';' END '.'
    |error
    ;
      $$
stmt_list:
    stmt $1          $3
    |stmt_list ';' stmt
    |error       重複自己
    ;

stmt:
    assign
    |read_
    |write_
    |for_
    |ifstmt
    ;
```

$X: lex傳過來之傳遞值
num:位置

TA : 林晉廷 2019.5.20

Grammar Loop(Java)

calc.lex

```
%{
    #include <stdio.h>
    #include "y.tab.h"
%}

type long
symbol [;]
identifier [$_a-zA-Z][$_a-zA-Z0-9]*
character   .

%%

"," {return(',');}
{type} {return(TYPE);}
{symbol} {return(SYMBOL);}
{identifier} {return(IDENTIFIER);}
"\n" {return('\n');}
[ ]+ {/*do nothing*/}


%%
//Becasue yylex() is a defined routine in yyparse(), we can skip this part:)
```

calc.y

```
%{
    #include <stdio.h>
    void yyerror(char *str);
%}

%token IDENTIFIER TYPE SYMBOL

%%

declare:
    TYPE declare_loop SYMBOL '\n' { printf("Grammar Loop Example\n");}
    ;

declare_loop:
    id
    | declare_loop ',' id
    ;

id:
    IDENTIFIER
    ;


%%
int main(){
    yyparse();
    return 0;
}
```

Windows PowerShell

```
PS C:\TestFile\calc> flex calc.lex
PS C:\TestFile\calc> gcc lex.yy.c y.tab.c -ly -ll
y.tab.c: In function 'yyparse':
y.tab.c:1119:16: warning: implicit declaration of function 'yylex' [-Wimplic
it-function-declaration]
        yychar = yylex ();
                 ^~~~~
PS C:\TestFile\calc> .\a.exe
long a, b;
Grammar Loop Example
```

TA：林晉廷 2019.5.22

提醒

1. 無論會不會傳給parser都要print，包含註解
2. Symbol table要補齊，一樣會有評分(重複宣告、未宣告使用、錯誤宣告)
   - 與YACC有關聯，ex. --i 、 i- -3
3. Lex常見錯誤:
   - 負數、減號區分
   - 宣告類別未包含string (String)
   - 減負數
4. **Java常見的語法均為隱測可能出現的錯誤**

TA : 林晉廷 2019.5.20

隱測提示

1. compound如果只有一句可不用大括號
2. print裡不只有string，可符合java語法皆可
3. condition符合java語法皆可使用，且for、while不相同
4. if、else if、else要分清楚compound的區域
5. 陣列宣告 int [] i= new int [1];

Bonus:

1. object declare
2. id used

*因有些人期中分數較不理想故給隱測提示，希望大家都能拿高分

TA：林晉廷 2019.5.20

評分標準

評分的部分，不只是將4份測資通過，更要實作1-3項標準，配分如下:

1. Syntactic Definitions (Print error) 45%
2. Semantic Definitions 20%
3. Recovery 5%
4. 隱藏測資 20%
5. 口頭問答 10%

如有項目未做或未做完整都會被扣分， 以上配分為此項最多的扣分分數

TA : 林晉廷 2019.5.20