%{

#include<stdio.h>

#include<stdlib.h>

#include<ctype.h>

int yylex();

int lineno = 1;

int tokenval = 0 ;

extern int yyparse();

FILE\* yyin;

void yyerror(const char\* s);

int reg\_count = 0; // 虚拟的寄存器计数器

int new\_register() {

return reg\_count++;

}

%}

%token PLUS MINUS TIMES DIVIDE LPAREN RPAREN INTEGER

%left PLUS MINUS

%left TIMES DIVIDE

%right UMINUS

%%

lines : lines expr '\n' { free\_register($2); }

| lines '\n'

| ;

expr : expr PLUS expr { $$ = new\_register(); printf("add $t%d, $t%d, $t%d\n", $$, $1, $3); free\_register($1); free\_register($3); }

| expr MINUS expr { $$ = new\_register(); printf("sub $t%d, $t%d, $t%d\n", $$, $1, $3); free\_register($1); free\_register($3); }

| expr TIMES expr { $$ = new\_register(); printf("mul $t%d, $t%d, $t%d\n", $$, $1, $3); free\_register($1); free\_register($3); }

| expr DIVIDE expr { $$ = new\_register(); printf("div $t%d, $t%d\nmflo $t%d\n", $1, $3, $$); free\_register($1); free\_register($3); }

| LPAREN expr RPAREN { $$ = $2; }

| MINUS expr %prec UMINUS { $$ = new\_register(); printf("neg $t%d, $t%d\n", $$, $2); free\_register($2); }

| NUMBER { $$ = new\_register(); printf("li $t%d, %d\n", $$, tokenval); }

;

NUMBER : INTEGER { $$ = tokenval; }

;

%%

void free\_register(int reg) {

// 对于简化的实现，此函数不会实际释放寄存器

// 但在真实情况下，它将用于管理寄存器池

}

int yylex()

{

int t;

while (1) {

t = getchar();

if (t == ' ' || t == '\t')

;

else if (t == '\n')

return '\n';

else if (isdigit(t)) {

tokenval = 0;

while (isdigit(t)) {

tokenval = tokenval \* 10 + t - '0';

t = getchar();

}

ungetc(t, stdin);

return INTEGER;

}

else {

switch(t) {

case '+': return PLUS;

case '-': return MINUS;

case '\*': return TIMES;

case '/': return DIVIDE;

case '(': return LPAREN;

case ')': return RPAREN;

default:

tokenval = 0;

return t;

}

}

}

}

int main(void)

{

yyin=stdin;

do{

yyparse();

}while(!feof(yyin));

return 0;

}

void yyerror(const char\* s){

fprintf(stderr,"Parse error: %s\n",s);

exit(1);

}