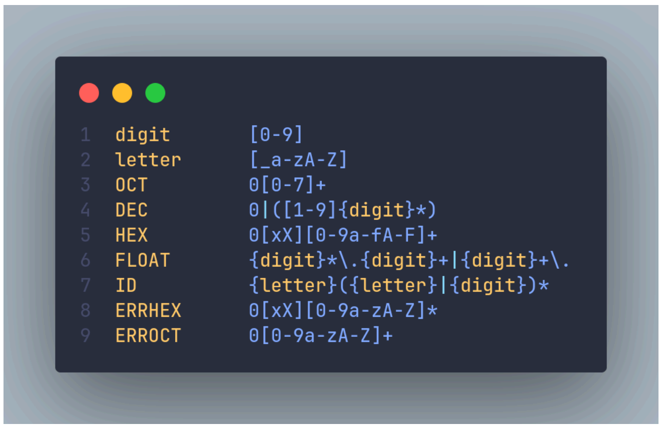
编译原理实验一：词法语法分析

词法分析

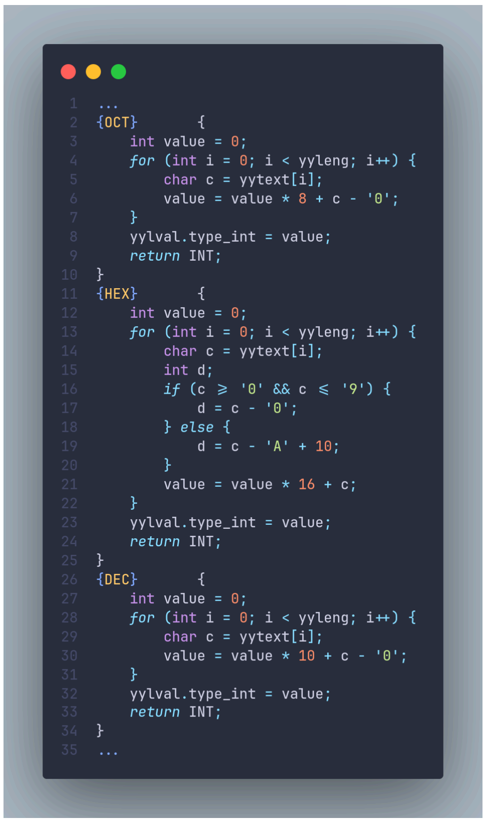
使用Flex进行词法分析代码的生成

编写词法单元的正则表达式



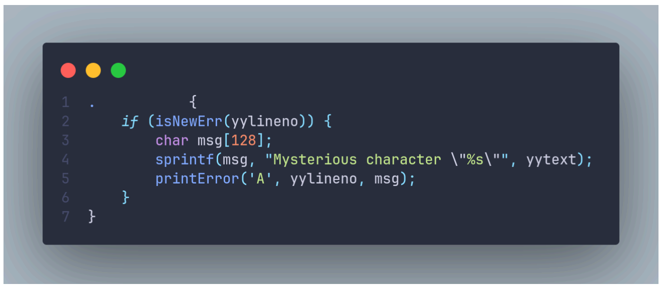
```
digit       [0-9]
letter      [_a-zA-Z]
OCT         0[0-7]+
DEC         0|([1-9]{digit}*)
HEX         0[xX][0-9a-fA-F]+
FLOAT       {digit}*\.{digit}+|{digit}+\.
ID          {letter}({letter}|{digit})*
ERRHEX      0[xX][0-9a-zA-Z]*
ERROCT      0[0-9a-zA-Z]+
```

编写检测到词法单元后进行的操作



```
...
{OCT}       {
    int value = 0;
    for (int i = 0; i < yyleng; i++) {
        char c = yytext[i];
        value = value * 8 + c - '0';
    }
    yylval.type_int = value;
    return INT;
}
{HEX}       {
    int value = 0;
    for (int i = 0; i < yyleng; i++) {
        char c = yytext[i];
        int d;
        if (c >= '0' && c <= '9') {
            d = c - '0';
        } else {
            d = c - 'A' + 10;
        }
        value = value * 16 + c;
    }
    yylval.type_int = value;
    return INT;
}
{DEC}       {
    int value = 0;
    for (int i = 0; i < yyleng; i++) {
        char c = yytext[i];
        value = value * 10 + c - '0';
    }
    yylval.type_int = value;
    return INT;
}
...
```

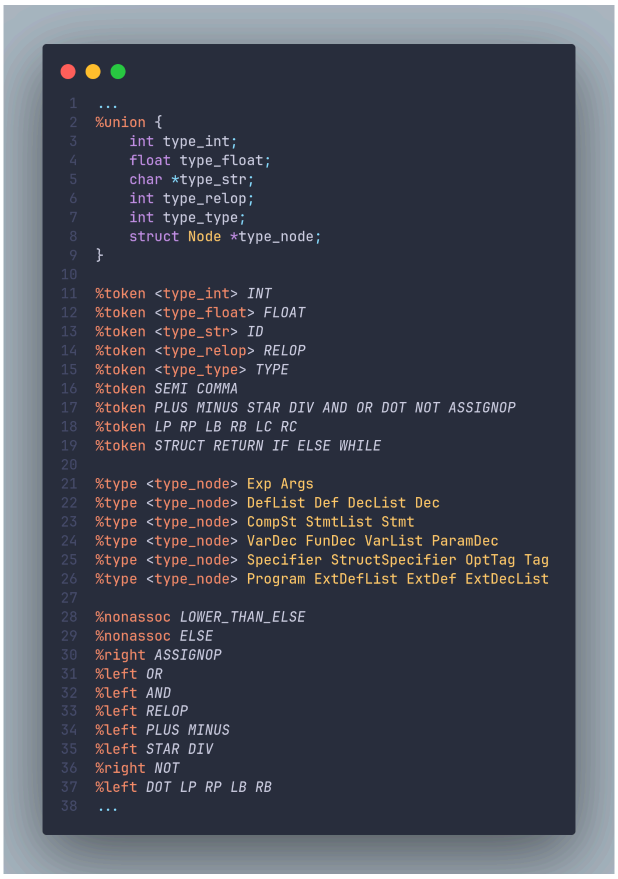
检测到未知词法单元后的错误处理



```c
.           {
    if (isNewErr(yylineno)) {
        char msg[128];
        sprintf(msg, "Mysterious character \"%s\"", yytext);
        printError('A', yylineno, msg);
    }
}
```

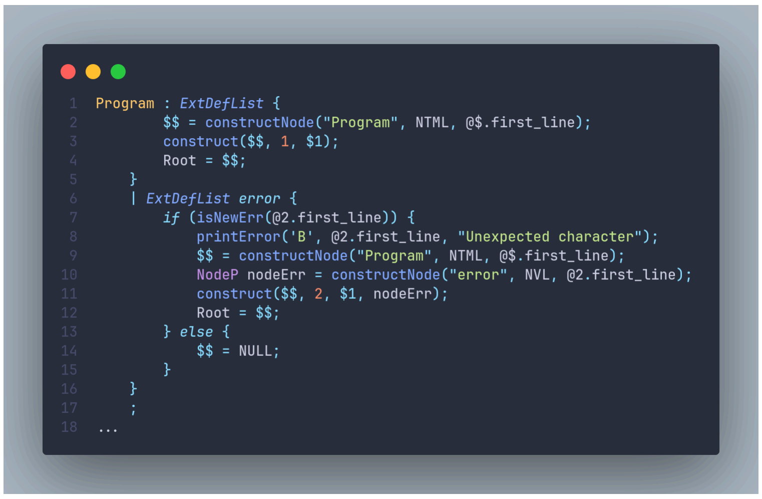
语法分析

定义终结符和非终结符，并指定类型和优先级



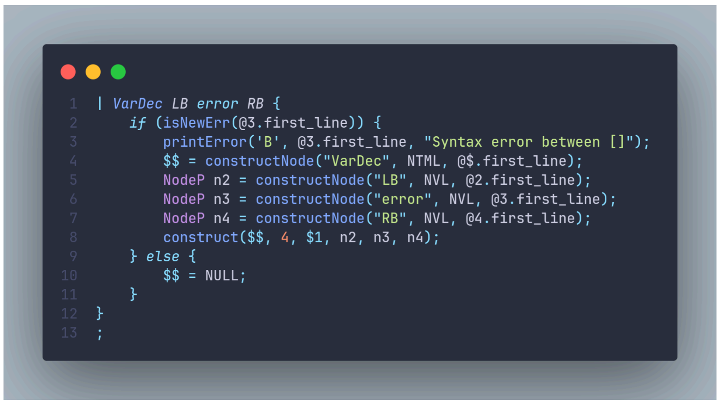
```
...
%union {
    int type_int;
    float type_float;
    char *type_str;
    int type_relop;
    int type_type;
    struct Node *type_node;
}

%token <type_int> INT
%token <type_float> FLOAT
%token <type_str> ID
%token <type_relop> RELOP
%token <type_type> TYPE
%token SEMI COMMA
%token PLUS MINUS STAR DIV AND OR DOT NOT ASSIGNOP
%token LP RP LB RB LC RC
%token STRUCT RETURN IF ELSE WHILE

%type <type_node> Exp Args
%type <type_node> DefList Def DecList Dec
%type <type_node> CompSt StmtList Stmt
%type <type_node> VarDec FunDec VarList ParamDec
%type <type_node> Specifier StructSpecifier OptTag Tag
%type <type_node> Program ExtDefList ExtDef ExtDecList

%nonassoc LOWER_THAN_ELSE
%nonassoc ELSE
%right ASSIGNOP
%left OR
%left AND
%left RELOP
%left PLUS MINUS
%left STAR DIV
%right NOT
%left DOT LP RP LB RB
...
```

定义语法规则，并构造语法树



```
Program : ExtDefList {
        $$ = constructNode("Program", NTML, @$.first_line);
        construct($$, 1, $1);
        Root = $$;
    }
    | ExtDefList error {
        if (isNewErr(@2.first_line)) {
            printError('B', @2.first_line, "Unexpected character");
            $$ = constructNode("Program", NTML, @$.first_line);
            NodeP nodeErr = constructNode("error", NVL, @2.first_line);
            construct($$, 2, $1, nodeErr);
            Root = $$;
        } else {
            $$ = NULL;
        }
    }
    ;
...
```

对于数组下标、分号缺失、多余字符等语法错误进行检查和报错



```
| VarDec LB error RB {
    if (isNewErr(@3.first_line)) {
        printError('B', @3.first_line, "Syntax error between []");
        $$ = constructNode("VarDec", NTML, @$.first_line);
        NodeP n2 = constructNode("LB", NVL, @2.first_line);
        NodeP n3 = constructNode("error", NVL, @3.first_line);
        NodeP n4 = constructNode("RB", NVL, @4.first_line);
        construct($$, 4, $1, n2, n3, n4);
    } else {
        $$ = NULL;
    }
}
;
```

遍历语法分析树，输出语法结构信息（详见代码）

有待改进

- 构造语法树部分的代码可以优化的更加简洁
- 有两个测试点始终无法通过，还未找出问题，可能还存在某些语法错误不能识别