电子科技大学信息与软件工程学院

**实 验 报 告**

学 号 2018091613009

姓 名 王明远

（实验） 课程名称 编译技术实验(3)

理论教师 周尔强

实验教师 周尔强

**电 子 科 技 大 学**

**实 验 报 告**

**学生姓名：王明远 学号：2018091613009 指导教师：周尔强**

**组队排序1 学生姓名：**丁志鹏 **学号：** 2018091606007

**组队排序2 学生姓名：**黄粤升 **学号：** 2018091606010

**组队排序3 学生姓名：**王明远 **学号：** 2018091613009

1. **实验名称：**编译技术实验(3)
2. **实验学时：**4
3. **实验目的：**

学习所提供的“表达式文法”的LR分析处理.

学习使用bison完成带语法树的语法分析程序。

会运用bison与flex构建编译器的前端部分。

学会makefile构造工具。

1. **实验原理：**

Bison语法分析与flex词法分析

BNF范式

正则表达式

cmake

1. **实验内容：**

1. 学习所提供的“表达式文法”的LR分析处理

理解 calc1.l, calc1.y, calc2.l, calc2.y的内容

在eclipse中建立工程,对calc3.l, calc3.y调试运行

2. 学习lrgram.txt所提供的文法

与递归下降分析所提供的文法作比较

1. 学习、理解 Makefile 文件的格式及写法（选做）
2. 编写lrgram所提供文法的LR语法分析程序

(1)编写生成“语法树”的相关程序，包括

bison源程序 lrparser.y

flex源程序 lrlex.l

语法树相关程序 ast.h 和 ast.c

(2)其它相关函数(如main函数)等 main.c,

(3)使编译得到的rdparser最终从命令行读取要分析的程序test.c,分析后调用showAst打印该程序的结构。

1. **实验器材：**

  个人PC机，包括bision

1. **实验步骤：**

先完成语法分析部分：

再在直接复制部分引入头文件

1. extern int yylex();
3. extern void showAst(Ast \*, int);

在定义部分，定义所有的终结符与注释。

1. /\* 引入bison生成头文件 \*/
2. #include "lrparser.h"
3. #include "ast.h"

在每个项目的匹配代码中，返回在bison中定义好的终结符码。

再完成语法分析部分：

声明几个外部函数

接着，定义好所有的终结符与非终结符。注意左右结合或非结合的区别。

1. %token VOID INT STR RETURN IF ELSE WHILE PRINT SCAN
2. %nonassoc CMP
3. %token NUMBER STRING
4. %token ID
5. %token EOL
7. %left '+' '-'
8. %left '\*' '/'
9. %left '='
11. %type program
12. %type external\_declaration
13. %type function\_definition
14. %type declaration
15. %type type
16. %type init\_declarator\_list
17. %type init\_declarator
18. %type declarator
19. %type add\_expr
20. %type intstr\_list
21. %type initializer
22. %type direct\_declarator
23. %type parameter\_list
24. %type parameter
25. %type statement
26. %type compound\_statement
27. %type expression\_statement
28. %type selection\_statement
29. %type iteration\_statement
30. %type jump\_statement
31. %type print\_statement
32. %type scan\_statement
33. %type expr
34. %type assign\_expr
35. %type cmp\_expr
36. %type mul\_expr
37. %type primary\_expr
38. %type expr\_list
39. %type id\_list
40. %type statement\_list

在每个BNF文法中，匹配好对应的代码。

例子：

思路仍然是实验二的。节点定义如下：

1. init\_declarator\_list: init\_declarator {
2. $$ = buildRootNode("init\_declarator\_list", 0);
3. $$->left = $1;
4. }
5. | init\_declarator\_list ',' init\_declarator {
6. $$ = buildRootNode("init\_declarator\_list", 1);
7. $$->left = $1;
8. $$->right = $3;
9. }
10. ;
11. typedef struct AST{
12. char name[20];
13. int path;
14. int nodetype;
15. unsigned int lineno;
16. unsigned int col;
17. char \*value;
18. struct AST \*left;
19. struct AST \*right;
20. struct AST \*next;
21. }Ast;

除此之外，使用cmake自动构造工具代替gnu-make。

1. /\* 取消yywrap使用、取消默认匹配行为、开启行数计数 \*/
2. %option noyywrap yylineno

Cmake配置代码如下：

在工程基目录CMakeLists.txt中写入：

1. cmake\_minimum\_required(VERSION 2.8)
3. project(rdparser)
5. add\_subdirectory(src)

第一行表示最低支持cmake级别

第二行表示工程名称

第三行表示添加一个子目录文件夹src

在src文件夹下同样放置一个CMakeLists.txt文件：

1. # bison
3. find\_package(BISON)
5. BISON\_TARGET(lrparser ${CMAKE\_CURRENT\_SOURCE\_DIR}/origin/lrparser.y ${PROJECT\_SOURCE\_DIR}/src/lrparser.c
6. DEFINES\_FILE ${PROJECT\_SOURCE\_DIR}/include/lrparser.h
7. )
9. # flex
11. find\_package(FLEX)
13. FLEX\_TARGET(lrlex ${CMAKE\_CURRENT\_SOURCE\_DIR}/origin/lrlex.l ${PROJECT\_SOURCE\_DIR}/src/lrlex.c)
15. ADD\_FLEX\_BISON\_DEPENDENCY(lrlex lrparser)
17. include\_directories(../include/)
19. add\_executable(rdparser ast.c lrlex.c lrparser.c main.c ${BISON\_lrparser\_OUTPUTS} ${FLEX\_lrlex\_OUTPUTS})
21. set (EXECUTABLE\_OUTPUT\_PATH ${PROJECT\_SOURCE\_DIR}/bin)