

编译Project

COMP130014.01

2018.09

简介

- Project分成两部分，评分分别占40%与60%
- 每组不超过4个人
- 小组成员名单发送至负责PJ的TA
- Project介绍及demo:
https://github.com/linchuming/Compiler_Project
- TA联系方式:
 - 林楚铭 17210240037@fudan.edu.cn (负责Project)
 - 马晨曦 17210240039@fudan.edu.cn (负责课程作业)

实验环境

- OS: linux, 推荐Ubuntu
- 依赖: gcc/g++ (版本不限), flex, bison
- flex与bison安装(以Ubuntu为例):
 - `sudo apt-get install flex`
 - `sudo apt-get install bison`
- 实验环境也可在MAC OS以及WINDOWS下配置, 具体配置请自行搜索
- 简单来说, 就是C/C++配合flex与bison两个工具

实验目的

- 通过flex与bison两种工具，分析目标PCAT语言，并生成目标语言的语法树
- PCAT语言可看作一种简化版的PASCAL语言

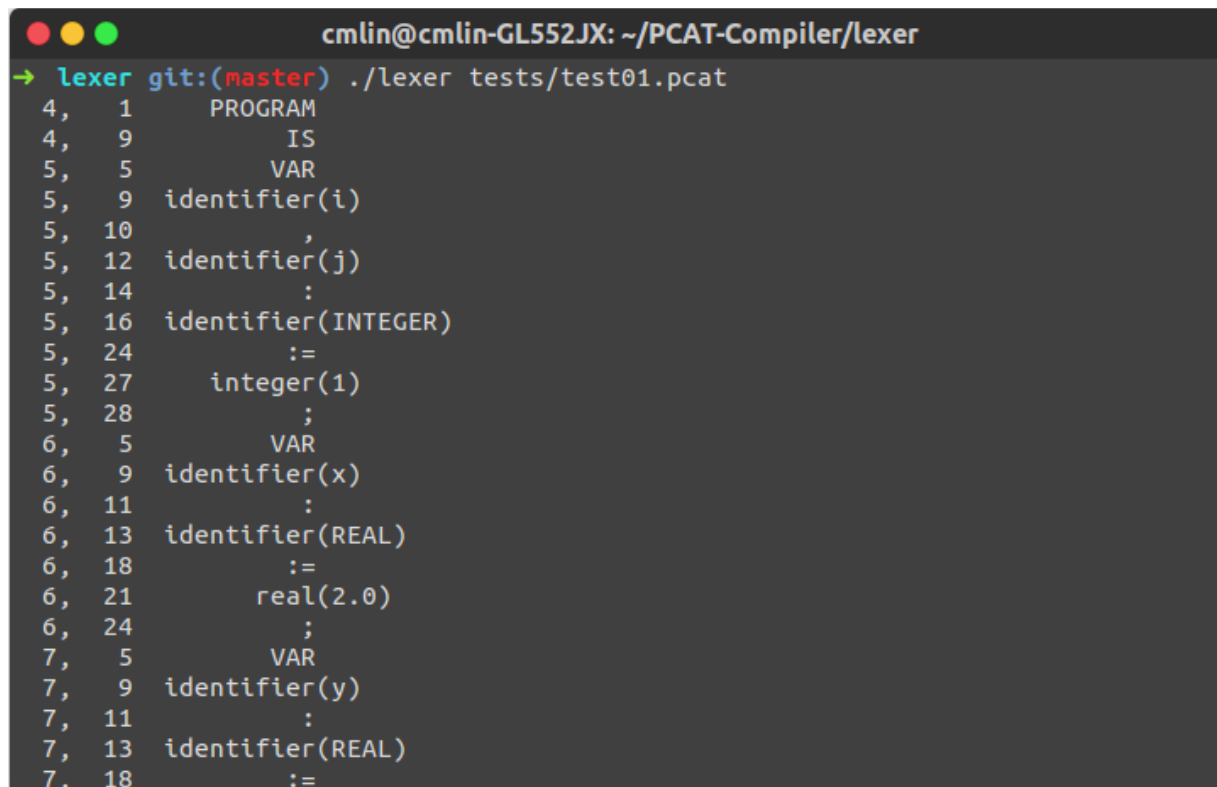
```
(* test01:      *)
(* test var decls. *)
(*              *)
PROGRAM IS
  VAR i, j : INTEGER := 1;
  VAR x : REAL := 2.0;
  VAR y : REAL := 3.0;
BEGIN
  WRITE ("i = ", i, ", j = ", j);
  WRITE ("x = ", x, ", y = ", y);
END;
```

Project 1 (40%)

- 使用flex，对于给定的PCAT语言的12个样例做词法分析，打印出所有的tokens
 - 完成基本功能（20%）
 - 输出每一个token的起始行号与列号（5%）
 - 对于一些基本的错误，如整型溢出，能提供报错信息（5%）
 - 编写文档，包含flex的使用用法，用到的正则表达式以及实现的细节等等，最后说明项目的成员分工（10%）
-
- 提交项目代码以及项目文档PDF
 - DDL: 2018年10月31日 11:59PM

Project 1 示例

- 输出格式参考，可以自定义，无需按照图片格式

A terminal window with a dark background and light-colored text. The title bar shows three colored circles (red, yellow, green) and the text 'cmlin@cmlin-GL552JX: ~/PCAT-Compiler/lexer'. The prompt is '→ lexer git:(master) ./lexer tests/test01.pcat'. The output consists of three lines of code, each with a line number and column number in the first two columns, followed by the code text. The first line is '4, 1 PROGRAM'. The second line is '4, 9 IS'. The third line is '5, 5 VAR'. The fourth line is '5, 9 identifier(i)'. The fifth line is '5, 10 ,'. The sixth line is '5, 12 identifier(j)'. The seventh line is '5, 14 :'. The eighth line is '5, 16 identifier(INTEGER)'. The ninth line is '5, 24 :='. The tenth line is '5, 27 integer(1)'. The eleventh line is '5, 28 ;'. The twelfth line is '6, 5 VAR'. The thirteenth line is '6, 9 identifier(x)'. The fourteenth line is '6, 11 :'. The fifteenth line is '6, 13 identifier-REAL)'. The sixteenth line is '6, 18 :='. The seventeenth line is '6, 21 real(2.0)'. The eighteenth line is '6, 24 ;'. The nineteenth line is '7, 5 VAR'. The twentieth line is '7, 9 identifier(y)'. The twenty-first line is '7, 11 :'. The twenty-second line is '7, 13 identifier-REAL)'. The twenty-third line is '7, 18 :='.

```
cmlin@cmlin-GL552JX: ~/PCAT-Compiler/lexer
→ lexer git:(master) ./lexer tests/test01.pcat
4, 1 PROGRAM
4, 9 IS
5, 5 VAR
5, 9 identifier(i)
5, 10 ,
5, 12 identifier(j)
5, 14 :
5, 16 identifier(INTEGER)
5, 24 :=
5, 27 integer(1)
5, 28 ;
6, 5 VAR
6, 9 identifier(x)
6, 11 :
6, 13 identifier-REAL)
6, 18 :=
6, 21 real(2.0)
6, 24 ;
7, 5 VAR
7, 9 identifier(y)
7, 11 :
7, 13 identifier-REAL)
7, 18 :=
```

Project 2 (60%)

- 结合之前完成的内容，使用flex&bison完成对PCAT语言的语法树建立，并打印出语法树
- 完成基本功能（40%）
- 具有语法报错功能并提示错误位置（10%）
- 编写文档，包括bison的使用方法，实现细节以及成员的分工（10%）
- 该项目需展示给TA，先完成可先展示，时间与TA预约即可
- 提交项目代码以及项目文档PDF
- DDL: 期末考试一周前

Project 2 示例

- 输出格式可自定义，能看出是树形结构即可

```
cmlin@cmlin-GL552JX: ~/PCAT-Compiler/parser
→ PCAT-Compiler git:(master) ls
documents lexer LICENSE parser README.md
→ PCAT-Compiler git:(master) cd parser
→ parser git:(master) ls
ast.c ast.o main.c parser pcat.lex pcat.output pcat.y run_test.rb
ast.h gc-7.2 Makefile pcat.c pcat.o pcat.tab.h pcat.yy.c test
→ parser git:(master) ./parser test/test01.pcat

(program
  (body
    (declaration_block
      (var_decl
        (id_list_node i j)
        (type INTEGER) 1)
      (var_decl
        (id_list_node x)
        (type INTEGER) 2)
      (var_decl
        (id_list_node y)
        (type_empty) 3))
    (statement_block
      (write_state "i = " i ", j = " j)
      (write_state "x = " x ", y = " y))))%
→ parser git:(master)
```


提交方式

- 如果文件太大，可先上传至百度云或者复旦云，再将网盘分享地址发送到TA邮箱；文件小则可直接发送到TA邮箱。
- TA邮箱： 17210240037@fudan.edu.cn
- 若对PJ有疑问，可与TA联系
- TA办公地址：计算机楼302

Flex简介

- 一种可使用正则表达式完成文本词法分析的工具
- 举例：提取出只有加法和减法的表达式的Token

Flex简介

lexer.lex

```
1  %{
2  #include "lexer.h"
3  %}
4  %option      nounput
5  %option      noyywrap
6
7  DIGIT        [0-9]
8  INTEGER      {DIGIT}+
9  REAL         {DIGIT}+"."{DIGIT}*
10 WS           [ \t]+
11
12 %%
13 {WS}          /* skip blanks and tabs */
14 <<EOF>>       return T_EOF;
15 "+"          return ADD;
16 "-"          return SUB;
17 {INTEGER}|{REAL} return NUMBER;
18 %%
19
```

定义区

规则区

用户代码区

lexer.h

```
1  #ifndef _LEXER_H_
2  #define _LEXER_H_
3
4  #define T_EOF      0
5  #define ADD        1
6  #define SUB        2
7  #define NUMBER     3
8
9  #endif
10
```

Flex简介

编译:

flex -o lexer.c lexer.lex

g++ -c lexer.c -o lexer.o

g++ lexer.o main.cpp -o main

```
cmlin@FudanT630:~/disk3/flex_demo$ ./main
1+2-3.3+2.2
1
+
2
-
3.3
+
2.2
```

```
1  #include <iostream>
2  #include <cstdio>
3  #include "lexer.h"
4  using namespace std;
5
6  int yylex();
7  extern "C" FILE *yyin;
8  extern "C" char *yytext;
9
10 int main(int argc, char **argv)
11 {
12     if (argc > 1) {
13         yyin = fopen(argv[1], "r");
14     } else {
15         yyin = stdin;
16     }
17
18     while (true) {
19         int n = yylex();
20         if (n == T_EOF) {
21             break;
22         }
23         cout << yytext << endl;
24     }
25
26     return 0;
27 }
```

参考资料

- Flex manual:
http://ranger.uta.edu/~fegaras/cse5317/flex/flex_toc.html
- Bison manual:
http://ranger.uta.edu/~fegaras/cse5317/bison/bison_toc.html

祝大家顺利毕业！

