**暨南大学本科实验报告专用纸**

课程名称 编译原理实验 成绩评定

实验项目名称 Bison工具的使用 指导教师 余芳

实验项目编号 实验2 实验项目类型 验证性 实验地点 N116

实验时间 2019 年 5 月 29 日 下 午～ 5 月29 日 下 午

**小组成员: 臧仲尧 2016051575 计算机科学与技术**

**邝庆璇 2016051598 计算机科学与技术**

**谢艳娇2016051389 软件工程**

**黄可炜2016053492 软件工程**

**(一)实验目的**

### 熟悉语法分析器自动生成程序Bison的使用。

**(二)实验内容**

**1.写出算术表达式的无二义文法，对比其与示例文法的区别**

**（设计输入使差别展现）。**

**2.自主设计任意文法，要求:**

**1)设计词法及语法规则；**

**2)写出构词规则序列、产生式序列及翻译规则；**

**3)给出输入、得到输出并分析结果。**

**(三)设计思想**

**1.算术表达式的无二义文法**

1. >E+E

E->E-E

1. >E\*E

E->E/E

E->(E)

E->a

无二义文法:

E->E\*T|T

T->T/F|F

F->F+M|M

M->M-N|N

N->(E)|a

假设有输入：a+a\*(a-a)/a

若使用示例文法有两种语法分析树

E

E

\*

E

a

E

+

E

E

a

E

/

E

-

E

)

E

(

a

a

a

E

E

/

E

E

-

a

a

E

E

\*

E(

a

+

(

)

E

E

E

a

a

对应的非二义文法则只有一种语法分析树（见下页）

E

E

\*

T

T

/

T

F

a

N

E

)

(\*

M

N

N

a

F

+

M

MN

N

F

F

M

a

T

F

N

-

M

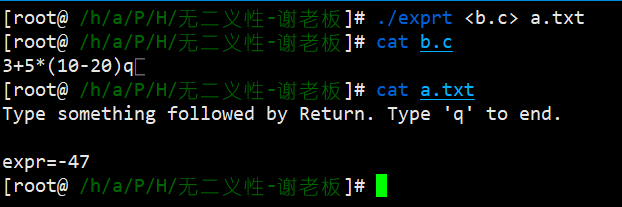
a

N

a

M

**实验结果：**



**2.编写语法分析器，进行JSON格式校验，输出其生成的语法树，**

**然后将校验内容进行格式化输出。**

①**JSON: JavaScript对象表示法(JavaScript Object Notation),**

是一种存储和交换文本信息的语法。

任何支持的类型都可以通过 JSON 来表示，例如字符串、数字、

对象、数组等。但是对象和数组是比较特殊且常用的两种类型。

**对象：**对象在 JS 中是使用花括号包裹 {} 起来的内容，数据结

构为 {key1：value1, key2：value2, ...} 的键值对结构。在面向

对象的语言中，key 为对象的属性，value 为对应的值。键名可以

使用整数和字符串来表示。值的类型可以是任意类型。

**数组：**数组在 JS 中是方括号 [] 包裹起来的内容，数据结构为

["java", "javascript", "vb", ...] 的索引结构。在 JS 中，数

组是一种比较特殊的数据类型，它也可以像对象那样使用键值对，

但还是索引使用得多。同样，值的类型可以是任意类型。

**②JSON实例**



（测试样例）



（这是阿里的一个机器学习云平台里头的json，通过chrome插件格式化显示了；为了显示方便，19到63行之间就收缩起来了）

③JSON词法规则。

定义如下token

* BEGIN\_OBJECT（{）
* END\_OBJECT（}）
* BEGIN\_ARRAY（[）
* END\_ARRAY（]）
* NULL（null）
* INT（整数）
* FLOAT （小数）
* STRING（字符串）
* BOOLEAN（true/false）
* SEP\_COLON（:）
* SEP\_COMMA（,）
* END\_DOCUMENT（#）

④JSON语法规则

1)start ->object END\_DOCUMENT| array END\_DOCUMENT

2)object -> BEGIN\_OBJECT END\_OBJECT

|BEGIN\_OBJECT members END\_OBJECT

3)members ->pair| members SEP\_COMMA pair

4)array -> BEGIN\_ARRAY END\_ARRAY

| BEGIN\_ARRAY elements END\_ARRAY

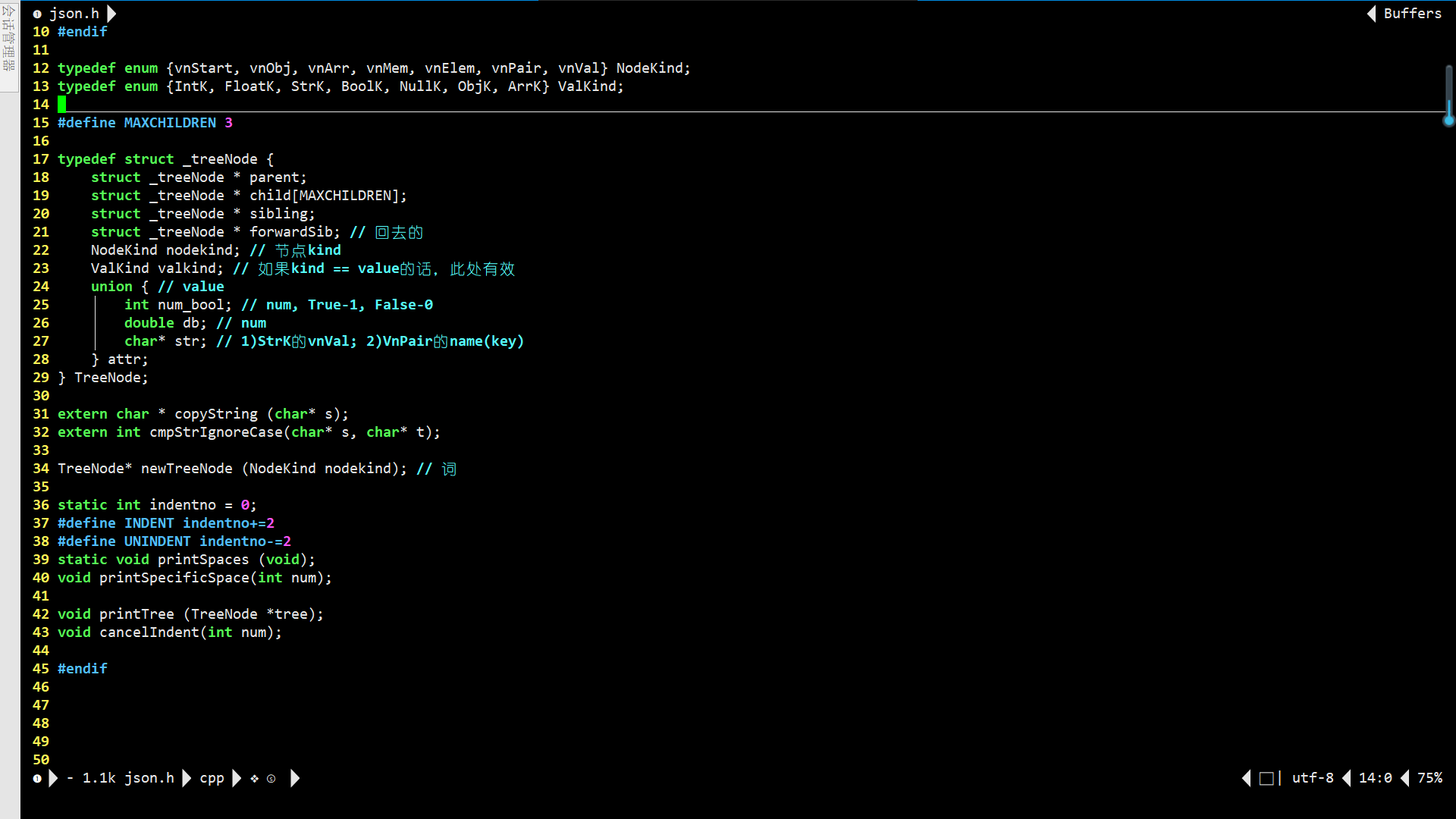
5)elements -> value| elements SEP\_COMMA value

6)pair -> STRING SEP\_COLON value

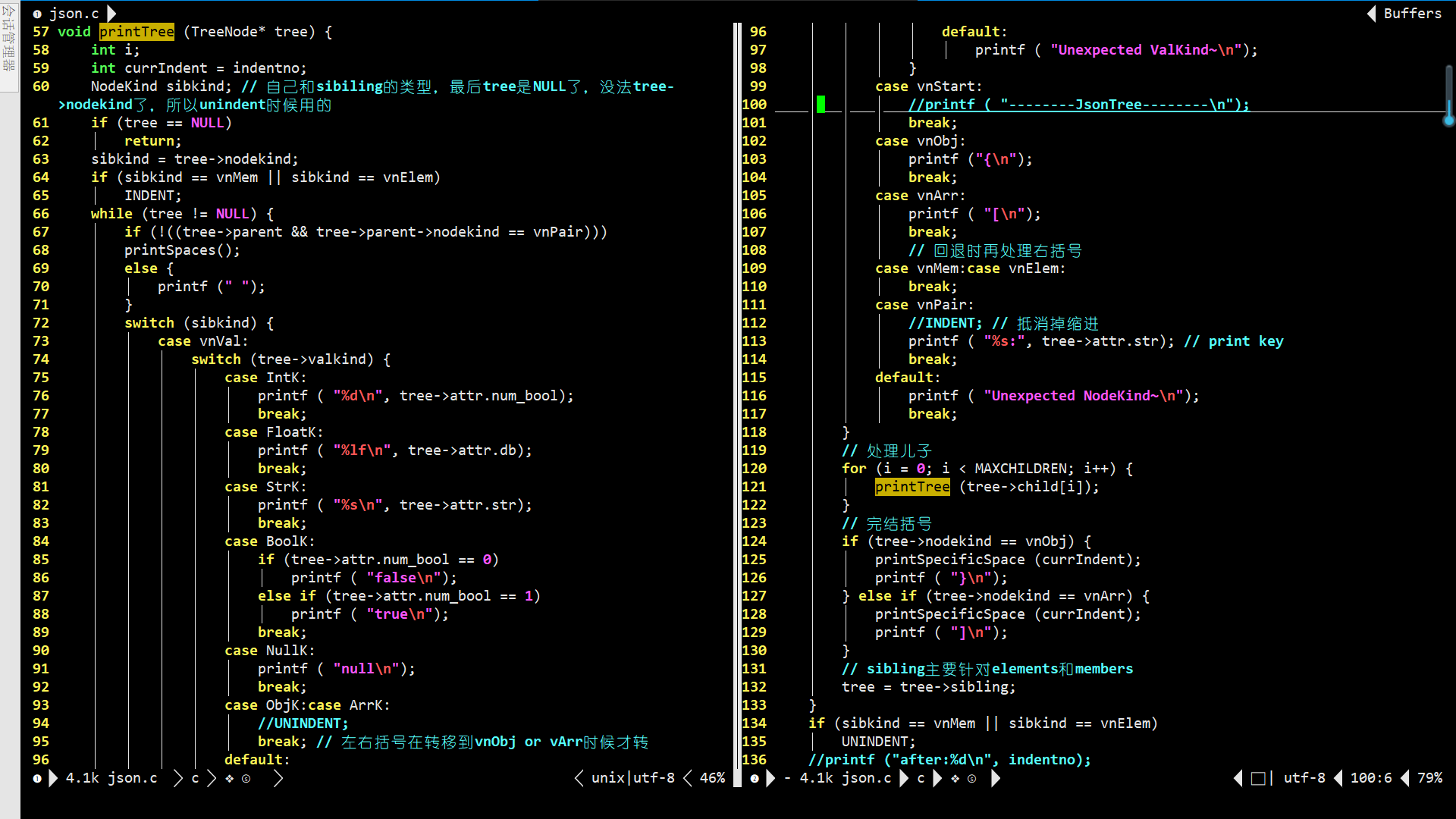
7)value -> STRING| object| BOOLEAN| NULL| array| INT| FLOAT

⑤为了生成并输出JSON语法树，仿照TinyC文件编写treenode结点结构体以及printTree函数。

1) treeNode结构体



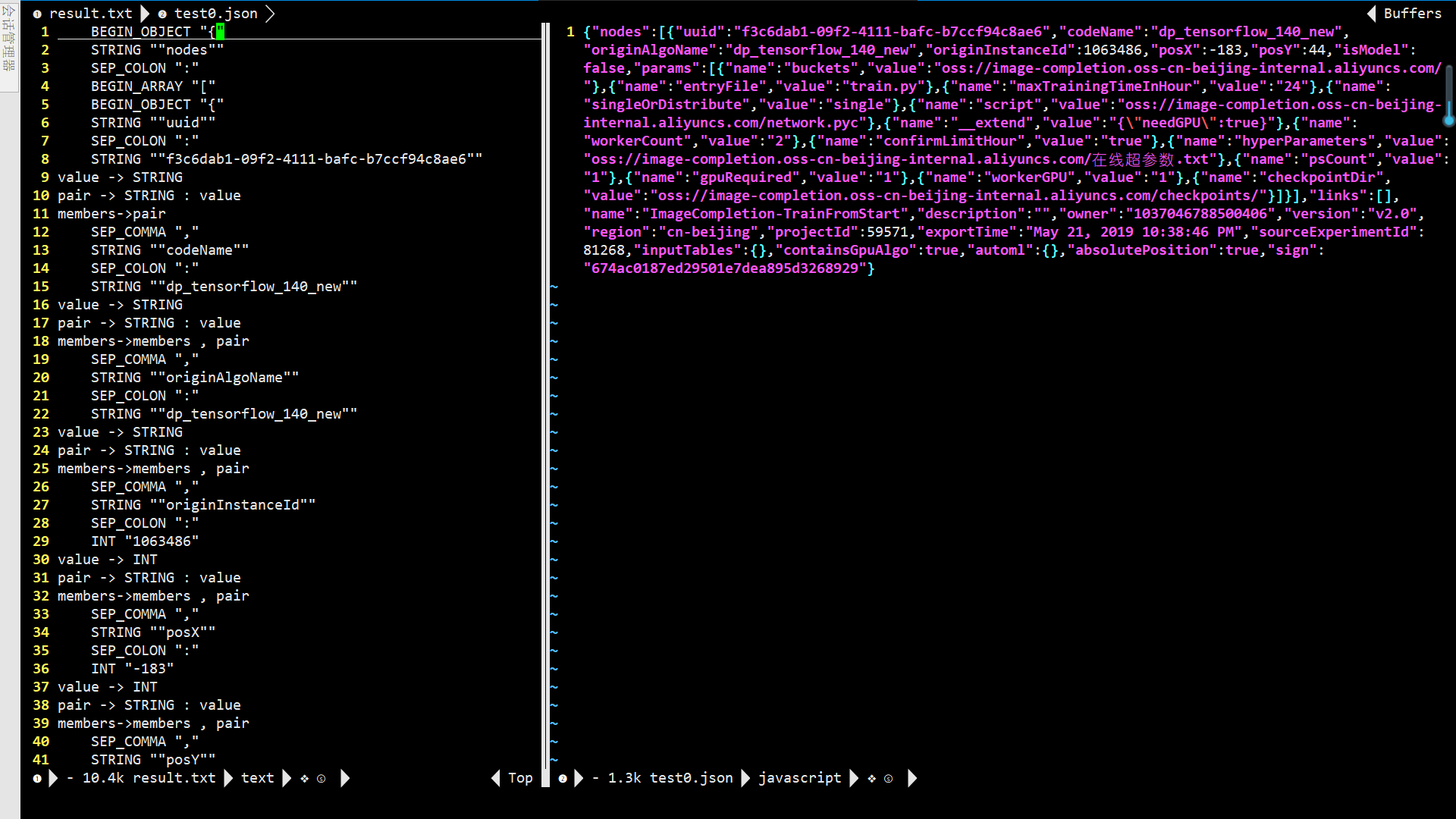
②printTree函数



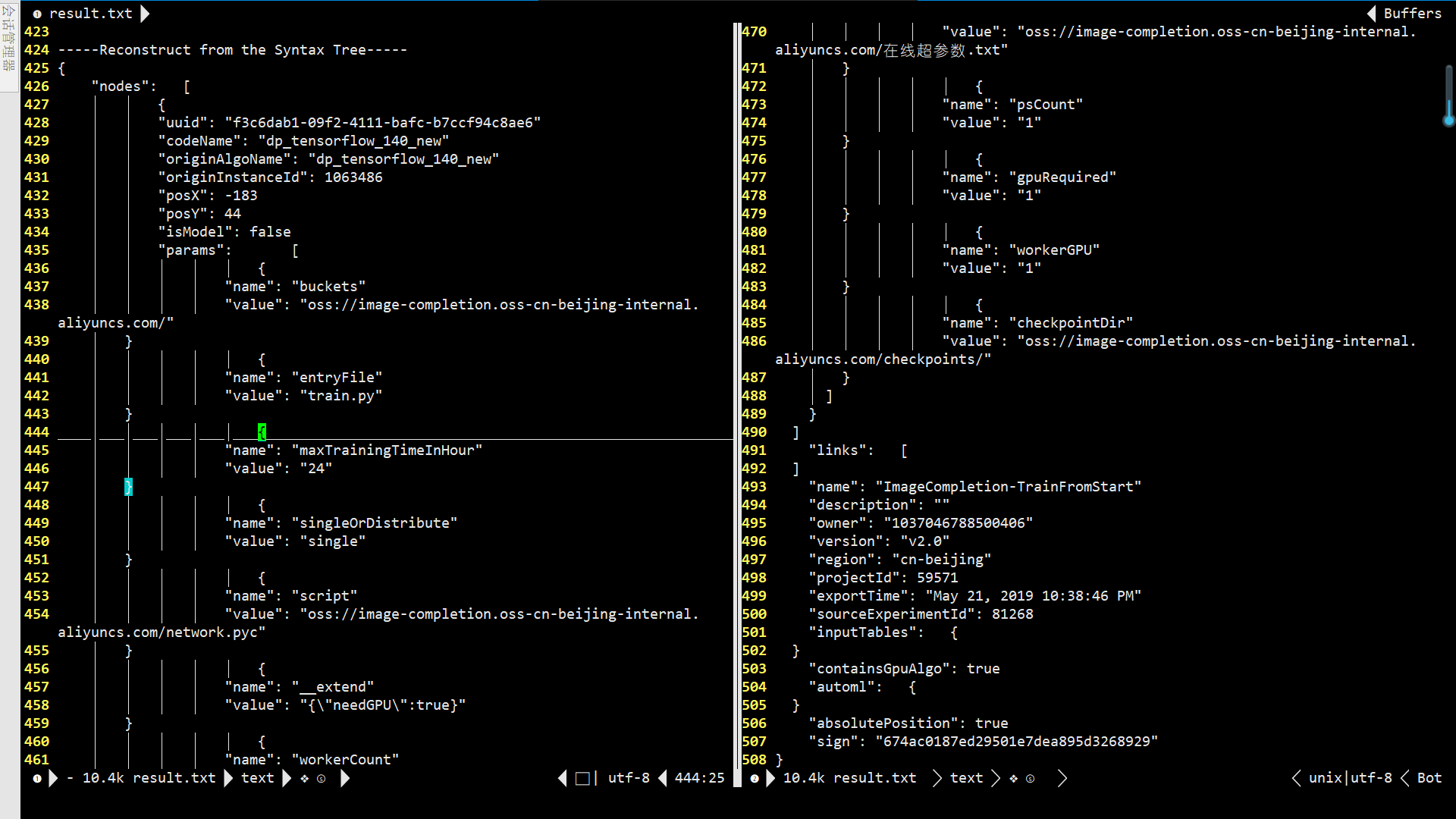
**(四)** **实验结果截图**

2. JSON测试文件以及实验结果展示。

①左边为语法分析器生成的结果文档，右边为测试文档。



②剩下的语法树以及对JSON文件的格式化输出



**(五)调试过程中遇到的问题**

#### （1）词法解析时，被处理的token没有返回值：

#### 在遍历过程中因为异常无法跳到下一状态。null在我的逻辑中应该归为NULL这类token，调试发现是在lex识别null时没有返回值。

（2）代码实现：

①参考了部分tinyc的代码及一点github上一个sql解释器的开源代码，但若要清楚理解前辈大佬们的代码和思路，还是要费点功夫的。

②makefile折腾了好一会儿。

③打印语法树部分，尽管已经在tinyc的基础上改了，但缩进一开始仍旧很有问题，后来才调好一点的。

**(六)实验总结**

JSON词法分析器的要点是遍历输入源，并尝试查找字符串，数字，布尔值，空值或JSON语法（如左圆括号和左括号）的模式，最终将每个元素作为列表返回。所以这里的目标是尝试匹配字符串，数字，布尔值和空值。如果这些都不匹配，检查字符是否为空白，如果是，则将其丢弃。若它是JSON语法的一部分（如左圆括号），就将其作为标记存储，若字符/字符串都不匹配，则最后抛出异常。

备注：JSON语法部分用了左递归，因为参考文档上说比较推荐使用，能提高效率。

**(七)小组分工**

·词法分析器:谢艳娇；

·语法规则:臧仲尧；

·代码部分:邝庆璇；

·测试补充等：黄可炜；

·实验报告撰写:前三人。

（链接：git@github.com:Susan9001/Assignment\_CompilersPrinciples.git）