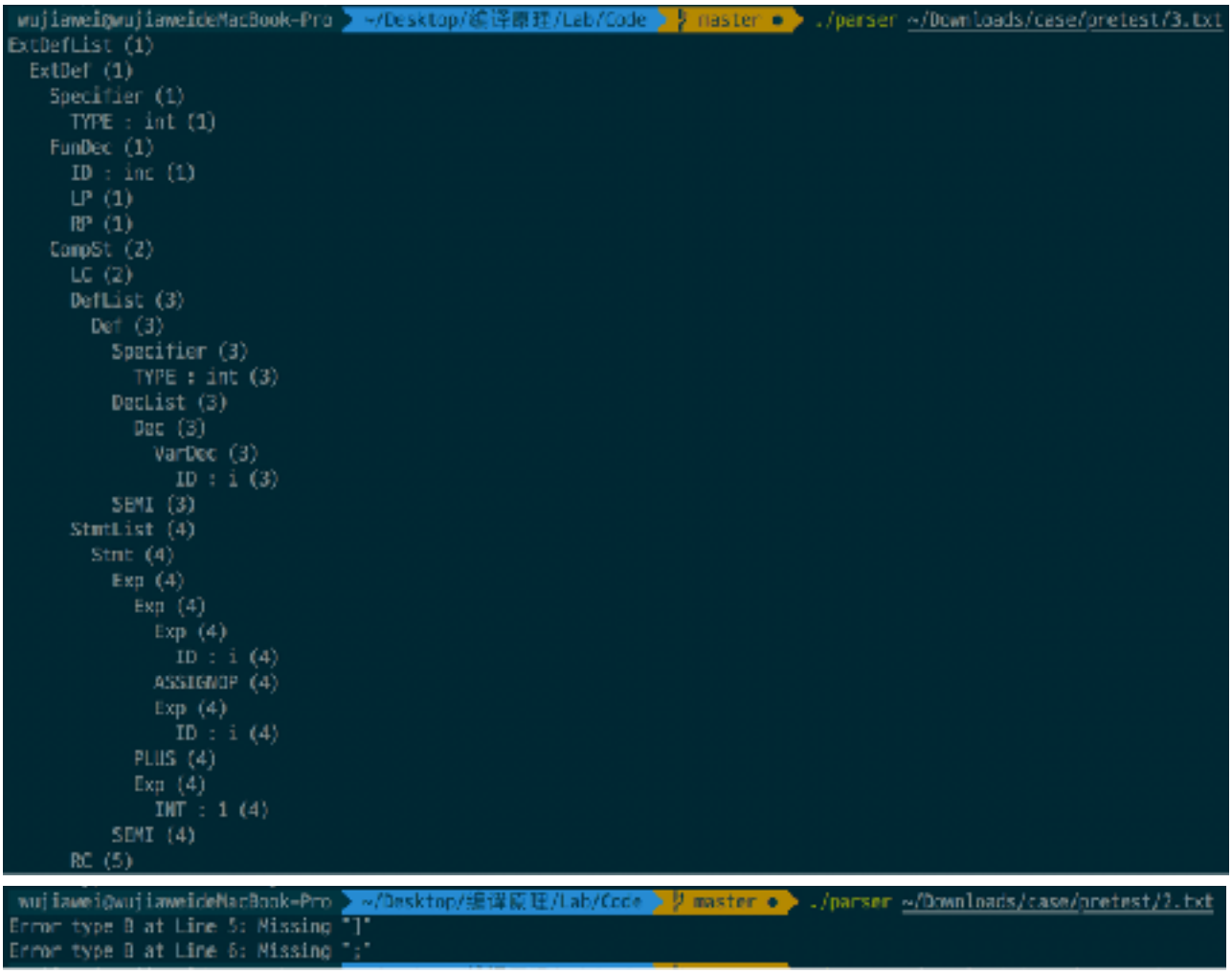
词法分析与语法分析实验报告

151220125 吴佳玮

邮箱：498279281@qq.com

一、 完成功能

实现了对C—语言的词法分析、语法分析以及简要的错误检测，并且支持输出语法树并标记行号。词法分析时完成1.1的要求，实现了对8进制和16进制的识别



二、 实验环境

实验编写在 MacOS 系统下，使用flex 2.5.35和bison 2.3编译通过并运行。提交之前在ubuntu 16.04环境下编译通过并测试成功

//macOS下编译需要-ll参数而非-lfl参数，在提交前已将Makefile恢复为-lfl

三、 实现解释

1、词法分析

每个词素的类型为NODE *,结构如下，每个节点记录了节点类型来方便打印语法树，同时记录行号。valFloat存储FLOAT类型的值，valString存储ID类型的值，valInt存储INT类型的值，同时在TYPE类型中，存储0(float),1(int)；在RELOP类型中，存储1(>),2(<),3(>=),4(<=),5(==),6(!=) //语法分析树中无要求，存储为实验二做准备

```
typedef struct node{
  enum NodeType type;
  int valInt;
  float valFloat;
  char valString[32];
  int length;
  struct node* childNodes[10];
  int linenum;
} Node;
```

2、 语法分析

使用结合性解决了if-else冲突，并按照文法规定了运算符号的结合性与优先级，每条语句动作都使用了可变参函数void insertNode(int num,…); 来插入子节点。

3、 语法树输出

使用void printTree(Node* start,int depth);函数输出语法树，在确定无错误之后，由根节点开始深度递归输出，并根据深度来缩进。利用枚举类型NodeType来确定节点类型，并根据字符串数组Nodename来输出类型名，这样避免了节点中存储字符串，节省了空间

4、 错误检测

重写yyerror函数将其置空，然后对每个“;””)””]””}”的产生式增加错误恢复语句，并且在每个错误恢复语句后输出错误信息且讲iserror变量置1，以此来判断程序是否出错。