|  |
| --- |
| 编译原理课程设计  实验报告  **指导教师：李荣**  **年 级： 2019**  **班 级： 28**  **小组编号： 1**  **组长学号姓名：21190824朱志宇**  **组员学号姓名：21191418张小北**  **组员学号姓名：21190823白羽帆**  **2022年 4 月 20 日**  **计算机科学与技术学院** |

|  |  |  |
| --- | --- | --- |
| **完成实验内容** | | |
| 从源文件读入SNL程序，进行词法分析，生成token序列，语法分析，生成语法树并可视化，语义分析 | | |
| **小组成员任务完成情况** | | |
| 姓名 | 具体完成任务 | 工作量  百分比 |
| 张小北 | 词法分析，程序UI设计 | 33% |
| 白羽帆 | 语法分析 | 33% |
| 朱志宇 | 语义分析 | 33% |
| **小组成员协作情况** | | |
| 张小北：完成词法分析，将token保存至文件  白羽帆：完成语法分析，建立语法树  朱志宇：根据语法树，完成语义分析 | | |
| **实验平台与编程语言** | | |
| **实验平台：Pycharm**  **编程语言：Python** | | |
| **实验方案设计** | | |
| 采用模块化开发，每人负责一部分功能，通过文件或者内存中变量传递模块间数据  张小北：从txt文件中读入程序，依照词法分析DFA将程序分为多个token，并将注释过滤，结果输出至指定文件。  白羽帆：从文件中读入104个产生式和对应的终极符，求出其first集，follow集，predict集。之后利用求出的predict集进行ll1分析和递归下降分析，输出语法检查信息，若语法无误则输出语法树。  朱志宇：接收语法树，根据语法树的建立过程（先根序列），即文法分析的匹配过程，通过进行的语法动作减少不必要的判断，避免不必要的分析（如推出空语句），如果推出多条同类型语句（xxxMore），利用循环分别处理每一条，分析每一处各种可能的语义错误，整体思路类似于递归下降。 | | |
| **程序界面及运行截图** | | |
| 1. 主界面：主界面左侧为选项框，可以选择展示程序，词法分析，语法分析和语义分析结果。上侧add按钮可从文件添加程序并编译，add右侧则可以选择展示哪个已被编译程序的结果。  1. 报错提示，语法，语义分析时若出现错误则通过弹窗提示。 | | |
| **源程序核心代码** | | |
| GUI：  GUI部分核心代码为词法，语法，语义分析调用逻辑  l = lex() #词法分析 l.run(directory)  flag=check\_grammar(0) #运行语法分析 if flag==1 : #若有错误flag为0  sem\_flag=sem\_run(directory) #语法分析无误，运行语义分析  if sem\_flag == False: #结果为False 语义分析无误  self.set\_program(directory) #编译完成，更新界面  else :   self.msg('sem’) #错误处理 else:  self.msg('parse')  词法分析：  def error(self,line): 词法出错的处理 def is\_reserved(self,ch): 是否为保留字 def scan(self,ch,line): 对一行代码进行词法分析  def run(self,path): 接口，读入文件，输出结果  语法分析：  def check\_grammar(mode=0, width="5000px", height="5000px")  def genFirstSet(key)  def findFirstSet(vl)  def genFollowSet()  def genPredictSet()  def ll1()  def begin(name, depth=1)  def visualization(width, height,gramTreeDic)  语义分析：  def analyze(self):  self.current = self.root.firstChild()  self.programHead()  self.declarePart()  self.programBody()  self.out.close()  def programHead(self):  self.stepInto("ProgramName")  self.stepInto("ID") def declarePart(self):  self.stepInto("DeclarePart")  self.typeDec()  self.varDec()  self.procDec() def programBody(self):  self.stepInto("BEGIN")  self.stmList()  self.stepInto("END") | | |
|  | | |