

1 引言

1.1 编写目的

本测试报告为编译原理课程项目的测试报告，目的在于总结测试阶段的测试及分析测试结果，描述系统是达到课程项目任务书中的要求，同时对软件质量进行相关的评估。

1.2 项目背景

详情见设计报告书

1.3 术语和缩略语

- 词法分析、NFA、DFA、最小化DFA
- 语法分析、文法化简、左递归、左公因子、LL(1)分析表
- RE文件：存有某个高级程序设计语言的所有单词的正则表达式的文本文件
- BNF文件：存有某个高级程序设计语言的BNF文法的文本文件

2 测试概要

2.1 测试环境与配置

客户端配置

机器名	CPU	内存	软件环境
Acer Nitro AN515-57	11th Gen Intel(R) Core(TM) i5-11400H @ 2.70GHz 2.69 GHz	16.0 GB (15.8 GB 可用)	Windows 10 家 庭中文版

IDE配置

IDE	平台工具集	Qt Installation
Microsoft Visual Studio Community 2022 17.4.2	Visual Studio 2022 (v143)	5.15.2_msvc2019_64

2.2 测试方法和工具

测试内容	测试方法	测试工具
功能	手工	Visual Studio 2022
可靠性	手工	Visual Studio 2022
安装和手册	手工	Visual Studio 2022

2.3 测试用例设计

编程语言	用例类型	正确性	数量
Tiny、Mini-C	源码文件	正确	3
		错误	1
	正则表达式文件	正确	1
		错误	0
	BNF文法文件	正确	1
		错误	0

详情见测试用例目录

3 功能测试

3.1 输入模块

对于文件的选择和读取并无问题，可点击或拖拽选择文件，并在浏览框中显示



3.2 输出模块

生成文件：

- 词法分析程序源码： `tiny-lex.c`, `minic-lex.c`
- 词法分析可执行程序： `tiny-lex.exe`, `minic-lex.exe`
- 单词编码文件： `tiny-token.txt`, `minic-token`
- 语法树文件： `tiny-tree.txt`, `minic-tree.txt`

存储位置

- 执行已打包程序：内嵌
- IDE中运行： "项目可执行程序目录"/`save_tiny/`、 "项目可执行程序目录"/`save_minic/`

测试中出现问题：

项目绝对路径中出现中文或空格，导致文件读写失败、执行外部可执行程序失败。

解决：

- 取文件绝对路径时进行编码转换
- 使用可识别空格的文件读写函数打开文件
- 采用 QProcess 执行外部程序，避免将可执行程序 and 参数手动拼接成字符串，造成调用失败

3.3 词法分析模块

正确源码得出正确结果



3.4 语法分析模块

正确源码得出正确结果



4 性能测试

进程内存占用

minic

初始：29MB

最高：36MB

tiny

初始：29MB

最高：40MB

5 易用性测试

- 导航和结构：导航和结构清晰，用户能快速找到需要的功能或信息
- 可用性：

- 页面布局：根据功能模块进行分区
- 字体大小：适中
- 颜色搭配：白底黑字
- 可理解性：系统的控件文本和使用说明比较易于理解，较为符合用户的习惯和期望
- 反馈和确认：功能运行成功后自动跳转相关结果界面
- 多设备兼容性：未测试，建议在安装Windows 10以上系统的较新PC机型运行

6 整体测试结果分析

6.1 测试总结分析

对于tiny和minic的正确测试用例，能进行语法和词法分析，生成正确的NFA、DFA、最小化DFA、词法分析程序源程、可执行词法分析程序；对BNF文法能正确化简、消除左递归、提取左公因子、生成LL(1)分析表和语法树。

对于含有词法错误或语法错误的源代码，程序不会中断，能给出错误的单词和语法错误所在处。

对于格式或内容错误的正则表达式文件和BNF文法文件，未做限制，可能造成程序功能失效或崩溃。

6.2 测试缺陷和限制

- 采用手工方法测试，效率较低
- 没有批量输入文件功能
- 采用了图形界面，交互全靠点击、拖动

6.3 建议

- 采用批量选择的方式输入文件
- 检查用户输入的文件类型，是否与当前要分析的源码类型匹配，或禁止用户输入不匹配的文件
- 优化数据结构，使得用户不重新打开软件，就能切换分析的语言
- 增加对于输入正则表达式和BNF文法的详细描述
- 采用图形化方法描述生成的NFA、DFA、最小化DFA状态转换图和语法树