南开大学 计算机大类

姓名 仇科文

学号 2312237

班级 网安代管2.1班

2024年5月13日

高级语言程序设计

实验报告

目录

[一. 作业题目 2](#_Toc166517542)

[1． HashMark——使用模板与数据生成目标文本。基于flex & bison的井号表达式计算与基于Qt的GUI界面。 2](#_Toc166517543)

[二. 开发软件 2](#_Toc166517544)

[1． Visual Studio Code 2](#_Toc166517545)

[2． Qt 2](#_Toc166517546)

[3． Github Desktop 2](#_Toc166517547)

[三. 课题要求 2](#_Toc166517548)

[1． 面向对象。 2](#_Toc166517549)

[2． 单元测试。 2](#_Toc166517550)

[四. 主要流程 2](#_Toc166517551)

[1． 整体流程 2](#_Toc166517552)

[2． 实现思路 3](#_Toc166517553)

[3． 算法 4](#_Toc166517554)

[4． 单元测试 5](#_Toc166517555)

[五. 收获 9](#_Toc166517556)

高级语言程序设计大作业实验报告

# 作业题目

## HashMark——使用模板与数据生成目标文本。基于flex & bison的井号表达式计算与基于Qt的GUI界面。

# 开发软件

## Visual Studio Code

## Qt

## Github Desktop

# 课题要求

## 面向对象。

## 单元测试。

# 主要流程

## 整体流程

* + 1. 设计目标代码格式：使用“#{”和“}#”包裹不被显示的头部文本，使用“##”与“##”包裹需要被计算的部分（即“井号表达式”），其余部分当做文本处理。
    2. 设计Qt界面：将窗口分为三部分，分别展示头部文本、变量表与生成文本。
    3. 使用Qt设计井号表达式计算方法。
    4. 使用Flex & Bison重构了Qt表达式中的井号表达式计算部分。
    5. 编写接口文件，将由Flex & Bison生成的compiler代码及其附属代码中的方法封装进头文件中。
    6. 重构了使用Qt的原始代码拆分部分。
    7. 实现了细碎的小细节，例如拖动条。

## 实现思路

* + 1. Qt部分：

使用一系列类的单例对用户打开的代码进行处理，并在计算的步骤调用由Flex & Bison生成的compiler代码对井号表达式进行计算。

这里列举使用到的类及其功能：

TextReaded: \* 用于读取文件并存储。

TextSeperated: \* 用于拆分文件为表达式，并存储拆分后的部分。

HsExprCalculator: \* 用于计算井号表达式并存储计算结果。

HsExprComponents: \* 用于定义并存放井号表达式的组件（变量与函数）

TextGenerated: \* 用于拼合并存储生成好的文本。

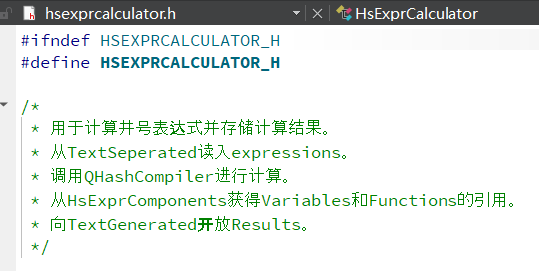
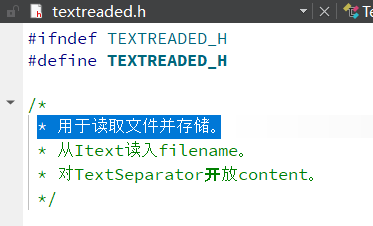
HsVarEditWidget: \* 专用于显示变量的组件类。

HsCodeGenerated: \* 用于生成用来保存或另存为的代码。

IQHashCompiler: \* Qt图形界面和F&B编译器的桥梁.

IText: \*文本处理部分与图形化界面的接口。

具体使用的类及其之间的调用关系可以前往项目源码中的各个头文件中进行查看，那里写得很详细，例如：



## 算法

* + 1. 1. Flex 中的算法
       1. 正则表达式转换成NFA（非确定有限自动机）：Flex 首先将定义的正则表达式转换为非确定性有限自动机（NFA）。这个过程涉及到正则表达式操作（如连接、选择和闭包）的图形化表示。
       2. NFA转换成DFA（确定有限自动机）：为了更高效地匹配输入文本，NFA 被进一步转换为确定性有限自动机（DFA）。DFA 确保每个输入字符都只有一个可能的状态转移，从而提高匹配的速度。
       3. DFA的最小化：DFA 还可以通过合并等效状态来进一步优化，减少状态数，提高运行效率。
    2. 2. Bison 中的算法
       1. LALR(1) 分析：Bison 主要使用 LALR(1) （向前看一个符号的LR分析）算法来生成语法分析器。LALR(1)用于处理大多数编程语言的语法。

## 单元测试

此处使用安装HashMark自带的demo.txt进行测试。

安装并打开HashMark，打开demo.txt如下：



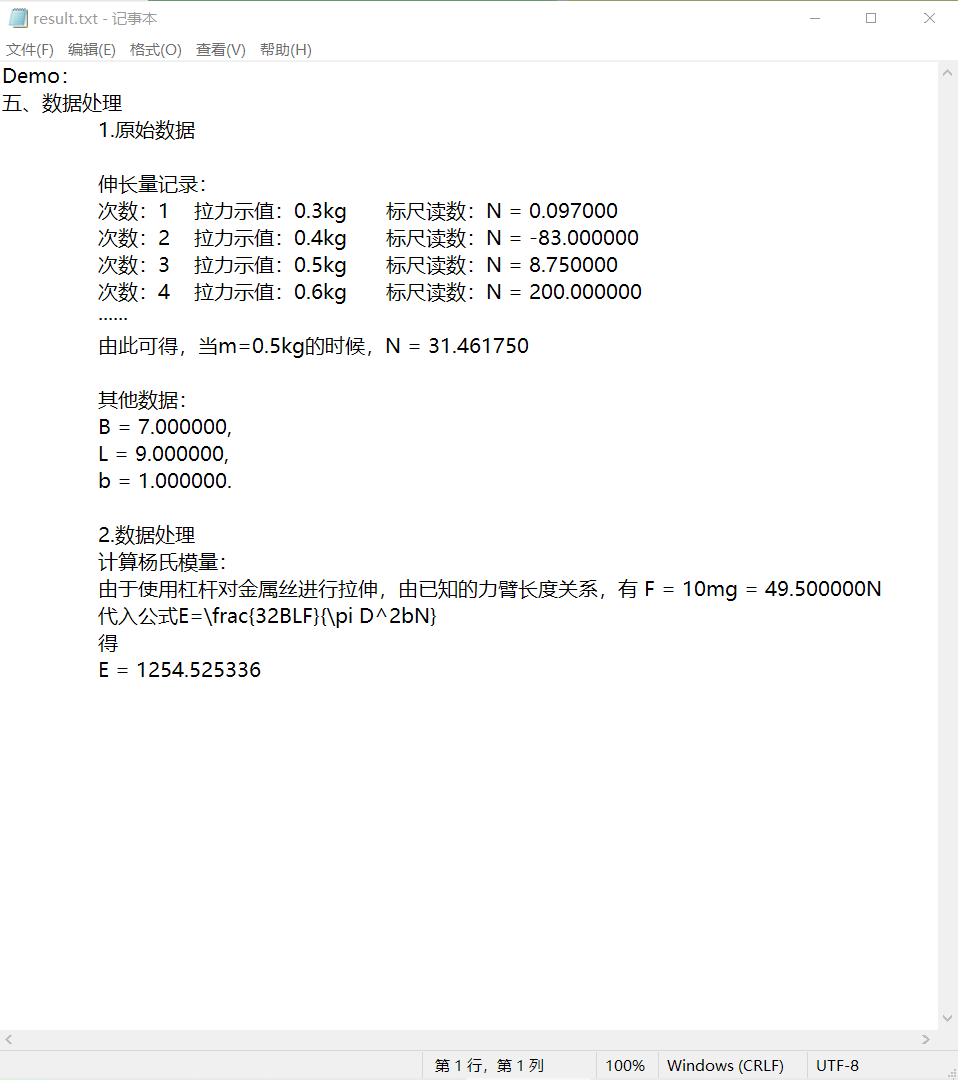
可以看到正确识别了demo.txt中的内容：



尝试更改数据：



选择导出，可得到导出文本：



# 收获

## 学会了Qt的基本使用方式。

## 学会了多文件项目的组织方法。

* + 1. 积累了组织多文件项目的经验。
    2. 感悟到模块化、降耦合化的重要性。
    3. 积累了重构代码的经验。
    4. 学会了编写合格的调试信息。

## 学习了编译器工程以及离散数学的部分知识。

## 学会了使用Git管理代码、使用GitHub Desktop提交代码到远程仓库。

## 学会了编写接口文件、构建模块化代码。

## 学会了C语言与C++语言的协作方式。

* + 1. 学会了编写从C到C++的接口文件，使用extern “C”。
    2. 学习了C语言的特性及其与C++的不同。

## 学习了变量、函数、文件命名规则。

## 学习了构建安装包。

## 学习了GitHub的相关用法。