2019年7月

学号: 1653298

姓名: 黄煜萧

专业: 计算机科学与技术

《数据结构》课程设计总结

目录

[第一部分 算法实现设计说明 3](#_Toc17882921)

[1.1题目 3](#_Toc17882922)

[1.2软件功能 3](#_Toc17882923)

[1.3设计思想 3](#_Toc17882924)

[1.4 逻辑结构与物理结构 4](#_Toc17882925)

[1.5 开发平台 4](#_Toc17882926)

[1.6 系统运行结果与分析 5](#_Toc17882927)

[1.7 操作说明 6](#_Toc17882928)

[第二部分 综合应用设计说明 8](#_Toc17882929)

[2.1 题目 8](#_Toc17882930)

[2.2 软件功能 8](#_Toc17882931)

[2.3 设计思想 8](#_Toc17882932)

[2.4 逻辑结构与物理结构 9](#_Toc17882933)

[2.5 开发平台 9](#_Toc17882934)

[2.6 系统运行结果分析说明 10](#_Toc17882935)

[2.7 操作说明 11](#_Toc17882936)

[第三部分 实践总结13](#_Toc17882937)

[3.1 所做的工作 13](#_Toc17882938)

[3.2总结与收获 14](#_Toc17882939)

[第四部分 参考文献 14](#_Toc17882940)

第一部分 算法实现设计说明

# 1.1题目

8. 堆的建立和筛选

输入一组关键值，用堆排序的方法进行从小到大的排序。

要求： (1)可以实现从小到大的排序，输出并显示该结果；

(2)随时显示输出堆顶元素，进行重新筛选后的堆。

# 1.2软件功能

软件所需实现的功能为堆排序。

堆排序算法分建立堆与调整堆两个步骤： 建立堆为在堆排序算法初始化时将原始数据建立为一个堆；调整堆为每次输出堆顶元素后，对余下数据进行调整使之仍为堆。当所有数据输出完毕后，算法结束。堆排序算法是以建立堆的算法初始化，并循环调用堆调整算法而形成的一套整体的排序算法。

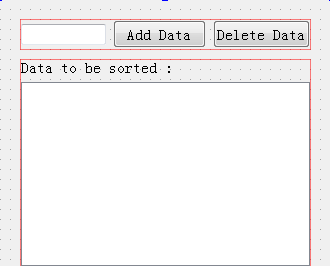
由于软件要求能随时显示进行重新筛选后的堆，所以仅需实现建立堆和堆调整算法，通过在软件内调用堆建立算法进行排序初始化，之后逐次调用堆调整算法来输出每步的排序结果及相应堆，可完成软件所需的基本功能。

# 1.3设计思想

软件按以下步骤设计：

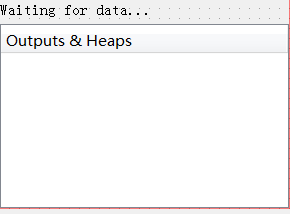
a.构建用于堆排序的HeapBuild()及HeapAdjust()算法，采用大顶堆进行排序，详见源代码heapsort.h文件；

b.构建输入界面及逻辑，用于输入数据，数据将存储于一个数组中，提供排序的输入；



c.构建输出显示界面，界面应使用树形控件构建，树未展开时只显示根元素(堆顶)，作为排序

的输出结果，树展开时作为当前输出结果对应的堆；



d.构建输出显示逻辑，将存储堆的数组转化为树形结构显示，详见源代码treebuild.h文件；

e.构建程序主循环，使用HeapBuild()初始化，调用HeapAdjust()输出每一步的结果，详见源代码mainwindow.cpp文件；

f.通过Debug模式详细设计用户交互，考虑各种可能出现的情况，完善软件。

# 1.4 逻辑结构与物理结构

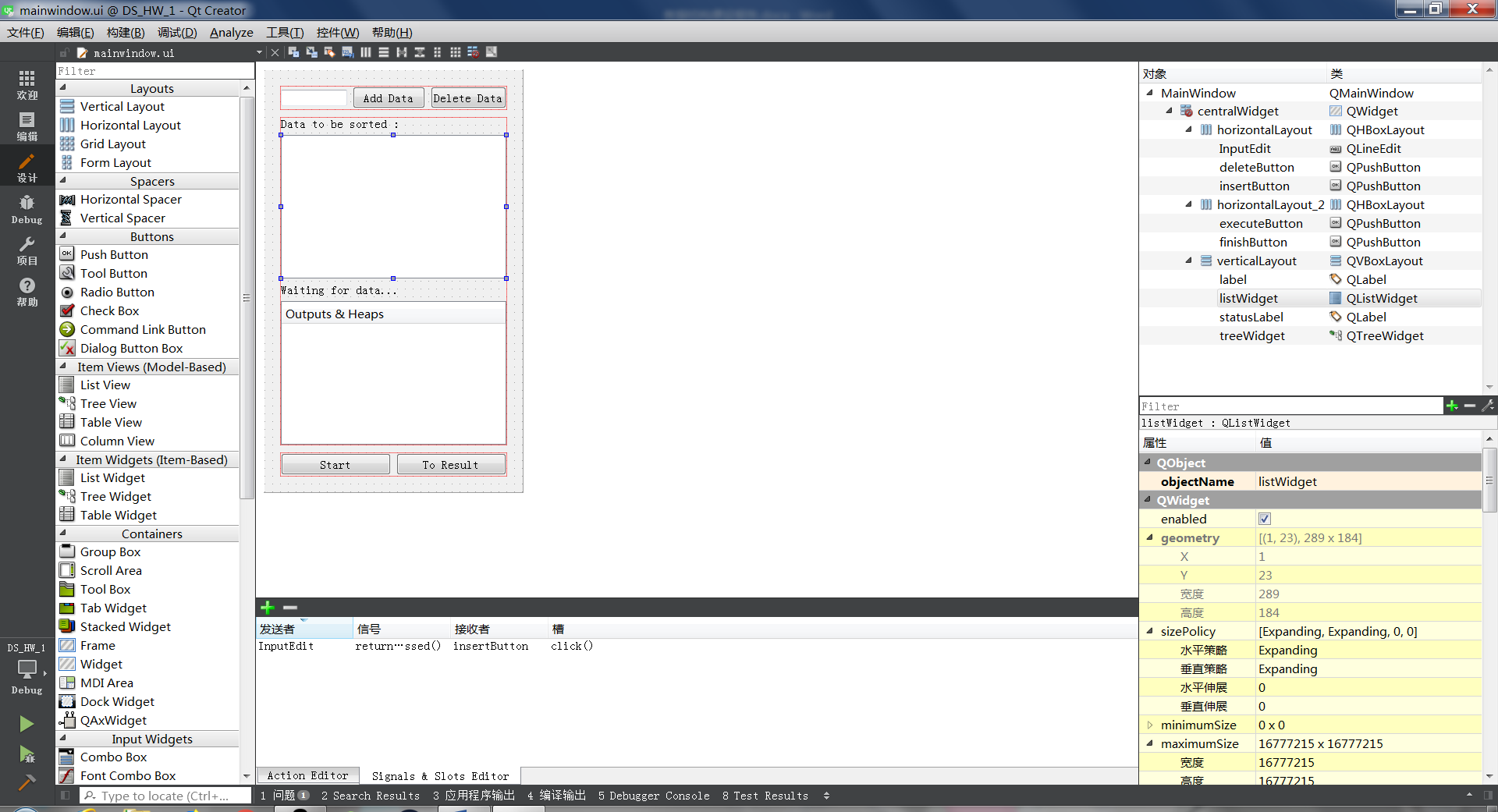
软件的主要功能侧重点在算法，所使用到的数据结构只有线性表。

由于软件对算法输入数据表的操作全由QListWidget控件托管，因此线性表只需要用于存储，不需要含有插入、删除等功能。

综上，软件所使用的数据结构的逻辑结构为简单线性表，物理结构采用顺序表，及C++数组。

# 1.5 开发平台

堆排序算法部分采用VisualStudio2017开发，利用控制台进行Debug。

界面及整体功能使用Qt 5.9.0进行开发，界面设计在Qt creator的界面设计器中进行。

界面的控制逻辑使用Qt的事件-插槽机制进行开发实现。

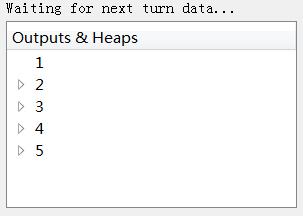
“执行程序”目录内的.exe程序在目录下(因为调用了dll文件)可直接运行，源代码文件最好安装Qt 5.9.0，使用Qt creator打开.pro文件进行浏览，VS2017也可进行源代码的浏览，不过会产生找不到Qt库的错误。

# 1.6 系统运行结果与分析

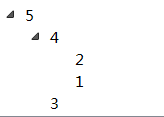
系统支持对32位整数的排序，界面信息采用英文，这是因为Qt 5.9.0对中文的兼容性存在一定问题。

软件运行正确并稳定，由于对用户输入有相应的规范及提示，软件也具有良好的容错性。

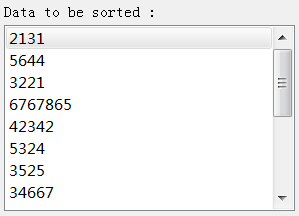
案例1：对5 4 3 2 1的排序：



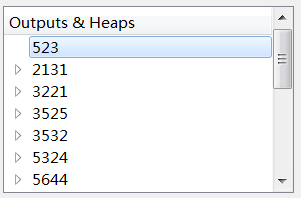
输出为5时对应的堆：

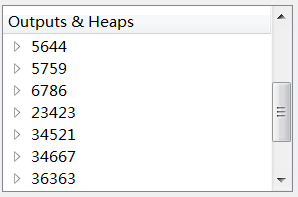


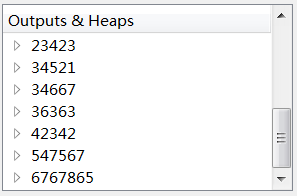
案例2：对稍大的混乱数据进行排序：



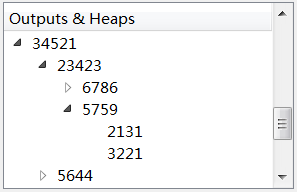
结果：





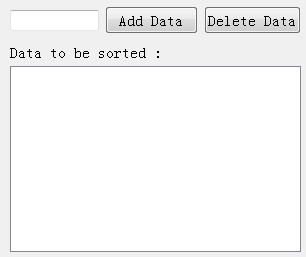


堆顶为34521的堆(只展开一部分)：

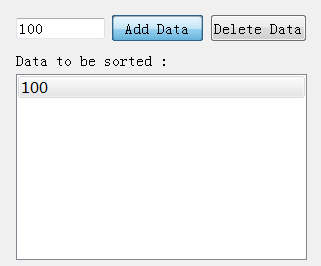


# 1.7 操作说明

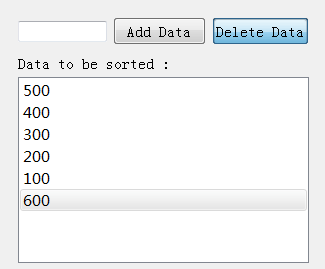
在数据输入部分录入待排序的数据：



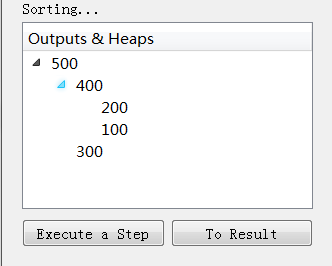
在文本框中输入待录入的数据，一次一个，点击Add Data按钮或按下回车添加



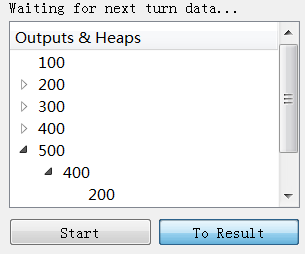
点击Delete Data按钮删除选中的数据



点击Start按钮开始排序，这时Start按钮变为Execute a Step，点击执行一步堆整理，输出相应的堆，堆顶作为当前排序输出的元素，点击三角符号可逐步展开堆对应的二叉树。



点击To Result按钮可直接执行至排序完成，排序完成后，可重复以上步骤进行下一组数据的排序。



第二部分 综合应用设计说明

# 2.1 题目

8.现需开发一个简单的文本编辑器（支持200字以内的输入即可），能支持的输入数据的形式和范围包括大写、小写英文字母、任何数字及标点符号。该编辑器需要实现以下功能：

(1)对输入的文字进行统计，统计出文字、数字、空格个数；

(2)统计某一字符串在文章中出现的次数，并输出该次数；

(3)删除某一子串，并将后面的字符前移；

(4)保存文本修改内容，可以撤销修改和恢复修改。

# 2.2 软件功能

软件所需实现的功能为载入、编辑、保存文本。

在编辑文本的过程中，应能实时显示文本中的字母、数字、标点个数，可对所作的修改进行一定步数的撤销、恢复。

软件还可实现在当前文本中对某子串出现次数的统计及对某子串的删除操作。

对文本的载入、保存通过C++的<fstream>标准库即可实现，编辑功能则由相应的文本域控件托管。软件的算法部分主要用于实现文本域控件内容变更时的数据控制、进行子串统计、删除的功能实现。

文本统计使用文本字符遍历即可实现，统计信息可通过一个标签控件在文本变更时输出

以实时显示。

字符串的查找可使用最为高效的KMP算法；字符串删除则可结合KMP算法所找到的位置，采用一般顺序表的基于索引删除方法进行删除。

撤销、恢复功能可考虑使用以文本串为元素的栈来实现。考虑到既要能向后撤销、又要能向前恢复，应使用两个栈：撤销栈在文本变更时将整个文本串入栈，用户提交撤销指令后出栈；恢复栈在撤销命令执行时入栈，用户提交恢复命令时出栈。

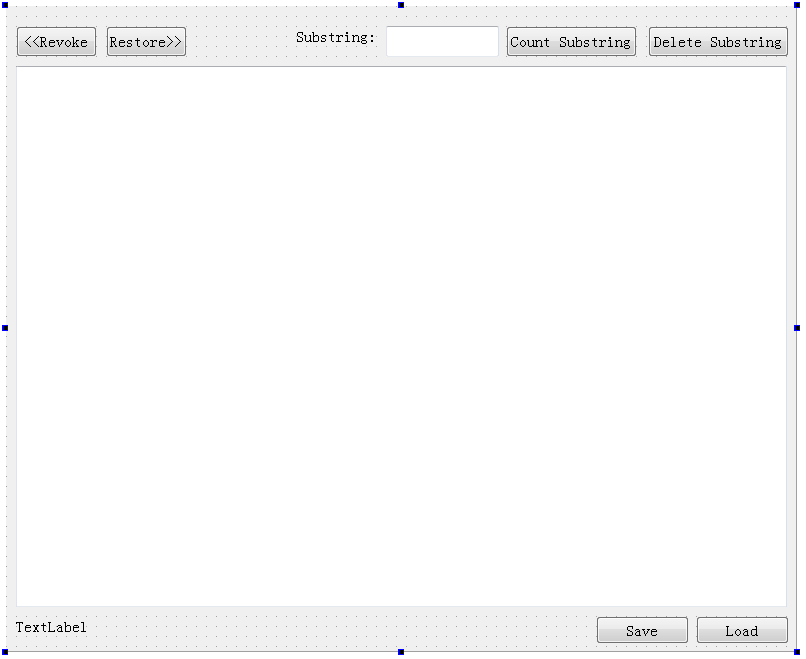
# 2.3 设计思想

软件按如下步骤设计：

a.编写KMP算法、子串删除、串信息统计算法函数，分别命名为KMPSearch()、DeleteRange()、Statistic()，实现相应功能，具体参见源代码textprocess.h文件；

b.编写用于存放撤销、恢复内容的栈类，类名为ContentStack，栈内部采用链表结构，具体参见源代码contentstack.h文件；

c.绘制界面，合理放置主要的功能控件;



d.在文本域控件QTextEdit发出文本改变信号，在信号的槽中调用Statistic()统计文本字符信息，输出至TextLabel标签上；

e.通过撤销栈(Bstack)、恢复栈(Fstack)实现撤销、恢复按钮的功能；

f.通过KMPSearch()函数、DeleteRange()函数以及弹窗控件，实现子串计数、子串删除功能；

g.通过<fstream>库实现载入、保存功能。

# 2.4 逻辑结构与物理结构

软件所使用到的数据结构有串和栈。

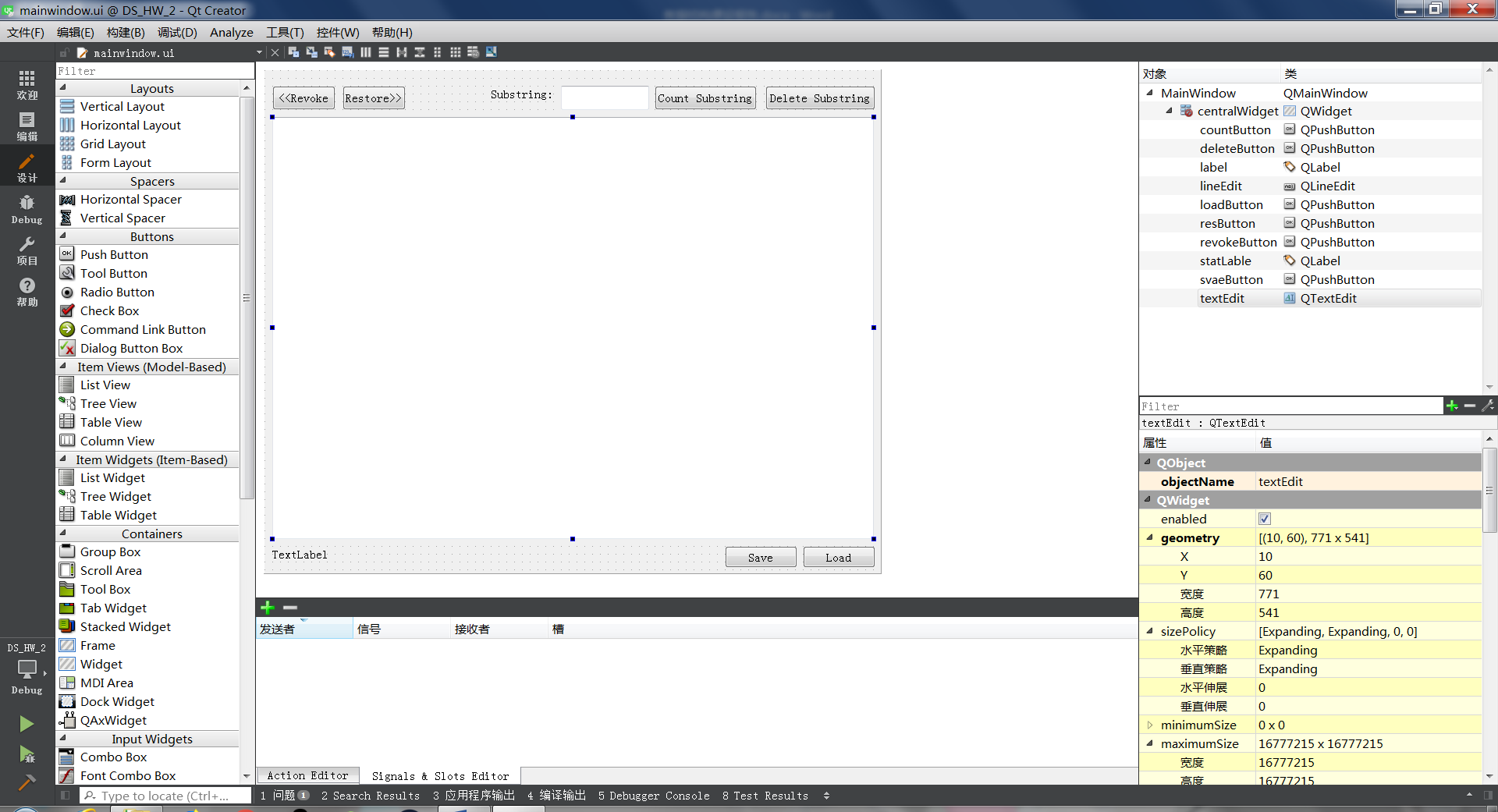
串用于文本处理，逻辑结构为存储ASCII码的线性表，物理结构为顺序表，使用的实体为C++的char数组。

栈用于撤销和恢复修改，栈内的元素为大容量字符串，用于跟踪记录文本内容。栈的逻辑结构采用线性表，物理结构采用单链表，单链表内定义了头指针、栈顶指针，以及相应的出、入栈操作。考虑到所使用的栈的特种性，栈初始化时会规定一个大小，大小表示允许撤销/恢复的最大步数，当入栈后栈大小超界时，自动丢弃栈底元素。

# 2.5 开发平台

字符串处理算法及栈类部分采用VisualStudio2017开发，利用控制台进行Debug。

界面及整体功能使用Qt 5.9.0进行开发，界面设计在Qt creator的界面设计器中进行。

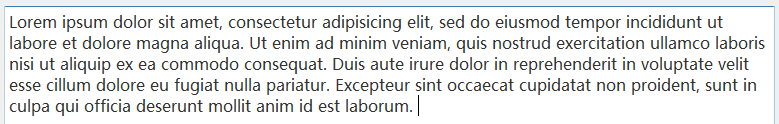


界面的控制逻辑使用Qt的事件-插槽机制进行开发实现。

“执行程序”目录内的.exe程序在目录下(因为调用了dll文件)可直接运行，源代码文件最好安装Qt 5.9.0，使用Qt creator打开.pro文件进行浏览，VS2017也可进行源代码的浏览，不过会产生找不到Qt库的错误。

# 2.6 系统运行结果分析说明

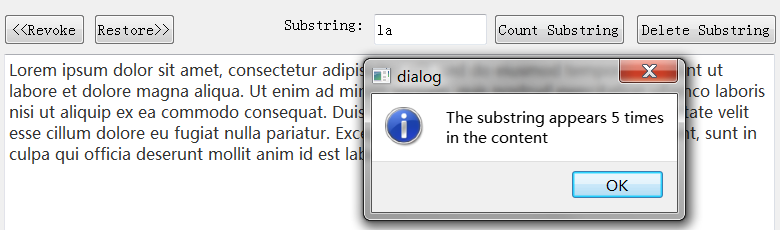
文本统计：





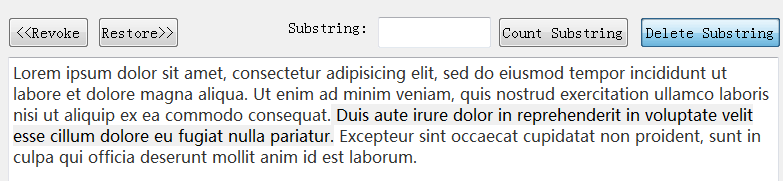
本软件支持最高2000字的英文文本。

统计子串出现次数：

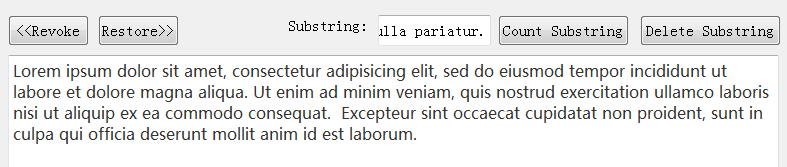


删除子串“Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur.”：

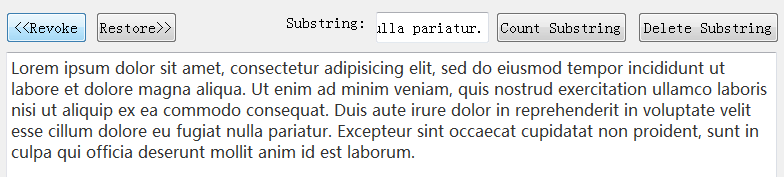
删除前：



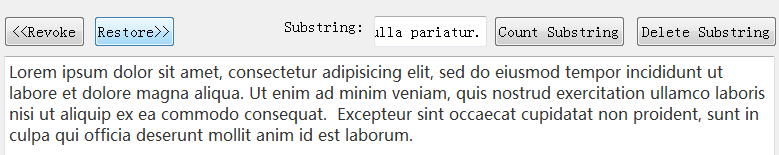
删除后：



撤销删除：



恢复删除：



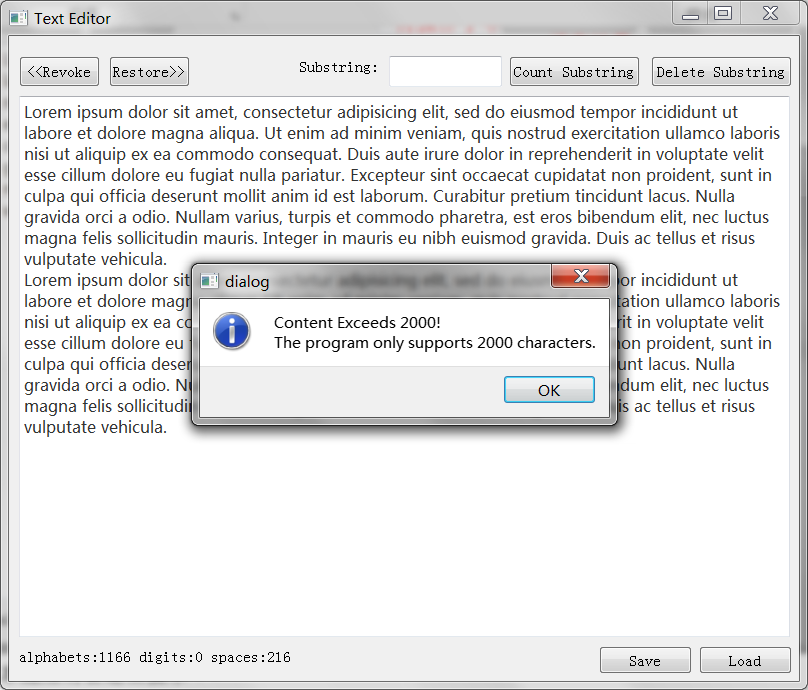
撤销/删除操作支持步数高达64次。

# 2.7 操作说明

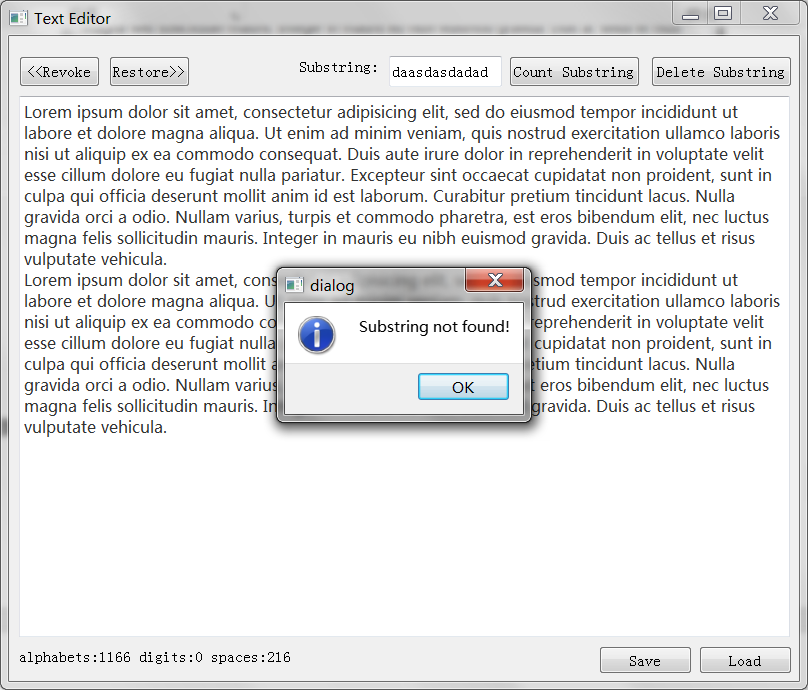
根据以下图片即可进行操作：



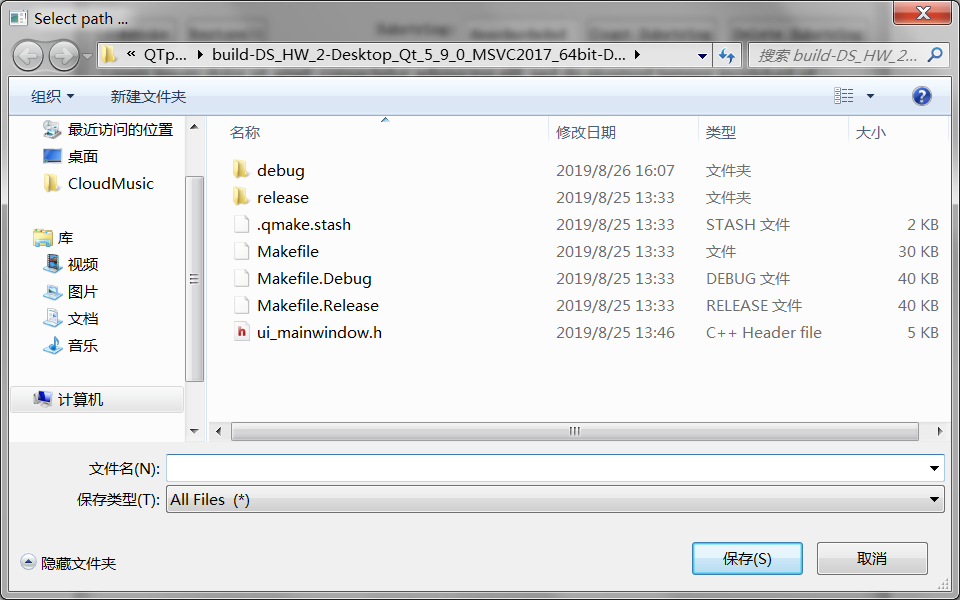
文本超限将会给出提示：



删除子串不存在将会给出提示：



点击保存、载入按钮将会弹出路径选择框：



第三部分 实践总结

# 3.1 所做的工作

本次课程设计主要完成了以下内容：

a.堆排序的算法实现即可视化；

b.简单文本编辑器的设计和编写。

本次项目使用了Qt 5.9.0 进行界面编写，Qt 是一个1991年由Qt Company开发的跨平台C++图形用户界面应用程序开发框架。它既可以开发GUI程序，也可用于开发非GUI程序，比如控制台工具和服务器。Qt是面向对象的框架，使用特殊的代码生成扩展以及一些宏，Qt很容易扩展，并且允许真正地组件编程。

在使用Qt进行GUI编写的过程中，我体会到了Qt的精妙简洁之处，也发现了它在兼容性上的一些不足。

在完成a的过程中，我完成了对堆排序算法的编写和理解。由于项目要求实时展示堆排序的过程，算法函数实际只完成了堆排序的一些步骤，与GUI的交互结合起来才是完整的堆排序。在进行GUI的设计过程中，我合理地选取了控件，利用QListWidget进行待排序数据管理，利用QTreeWidget进行排序结果输出以及堆的显示。其中对QTreeWidget的选择十分恰当，因为堆的标准可视化本身就是一棵二叉树，而Tree类控件的可折叠/展开性质使它既可以作为排序的输出列表(列表元素既为输出又为堆顶)，又可作为每一步的堆可视化(将输出的堆顶展开即可看见堆二叉树)，避免了重复造轮子。

b的完成实际比a简单，通过Qt的QTextEdit控件已可实现文本编辑器的大部分功能，项目的中心放在对子串查找算法的编写、撤销/恢复所用数据结构的构建中。b的完成使用到了KMP算法，KMP本只作为课程的选学内容，通过对b的完成，我深入理解了KMP算法的思想，以及它采用的子串前后缀差异度特征函数(next[])的精妙之处。

# 3.2总结与收获

本次课设的重点不但在于课程所学数据结构、算法的应用，还在于GUI编程、数据可视化。在GUI设计方面，老师推荐的是CLR，但我使用的是Qt。Qt作为一款跨平台的C++ GUI库，可以创建绚丽的界面，这也是我选择它的原因之一。

但在实际操作过程中，我发现Qt的学习成本还是比较大的。虽然它很简洁，且集成了Qt creator的IDE供我们使用，但它的兼容性很差，会不时出现莫名其妙的编译问题，我在处理编译器问题上耗费了大量时间和精力。同时，Qt的控件机制为信号与槽，这与C#、CLR、WPF等的消息处理机制存在差异，也增大了我学习的成本。

后来我通过网上资料了解到，Qt多用于嵌入式开发，实际上不是很适合用于此次课设项目(windows 环境)，但它跨平台的强大特性和速度是毋庸置疑的。

在进行GUI组件编程的过程中，我对GUI编程的难点也有了一定的认识。我认为界面的设计、布局以及处理逻辑的设计其实并不难， GUI编程的难处在于数据间的类型转换、以及对控件使用规则的学习。算法所生成的原始数据类型是要经过转换才能化为控件可展示的数据的。对控件使用规则的掌握则只能凭借资料及经验，是需要长期学习、实践的。

最后，通过本次项目，我深入地理解及掌握了堆排序算法、KMP算法，了解到了它们的精妙之处；我也领悟到了一个提高编程速度和质量的秘诀：不重复造轮子。算法和数据结构是课设所要培养能力的核心，当然是资料仅作辅助，自己全盘编写的；但对数据的细枝末节操作完全可由控件托管，控件功能的实现、数据转换完全可以复制、修改网上代码，这样大大加快了编程速度，并提高了编程质量，减少了BUG。

此次数据结构课程设计将作为我以后学习的宝贵经验。

第四部分 参考文献

[1] CoolMan\_1995.Qt系列:用MSVC2015编译常见编译错误及解决方案[EB/OL]. https://blog.csdn.net/CoolMan\_1995/article/details/80319559, 08/28/2019 10:49:54

[2] C语言中文网. Qt教程[EB/OL]. http://m.biancheng.net/qt/, 08/27/2019 21:02:03

[3]li-peng. Qt学习笔记 ListWidget的增删改[EB/OL]. https://www.cnblogs.com/li-peng/p/3651945.html, 08/28/2019 10:53:04

[4] 笑着後悔. QT 发布Debug和Release的文件夹中缺少\*.dll文件[EB/OL]. https://blog.csdn.net/haha\_dabai/article/details/78120674, 08/28/2019 10:54:35

[5] Hengzuzong. KMP ~~从入门到理解到彻底理解[EB/OL]. https://blog.csdn.net/gao506440410/article/details/81812163, 08/28/2019 10:55:54

[6]百度经验. 堆排序算法解析[EB/OL]. https://jingyan.baidu.com/article/5225f26b057d5de6fa0908f3.html, 08/28/2019 10:57:32