

# 《数据结构》课程设计总结

学号： 2151127

姓名： 华洲琦

专业： 计算机科学与技术

2023 年 8 月

# 目 录

第一部分 算法实现设计说明.....	3
1.1 题目 .....	3
1.2 软件功能 .....	3
1.3 设计思想 .....	3
1.4 逻辑结构与物理结构 .....	5
1.5 开发平台 .....	5
1.6 系统的运行结果分析说明 .....	6
1.7 操作说明 .....	7
第二部分 综合应用设计说明.....	11
2.1 题目 .....	11
2.2 软件功能 .....	11
2.3 设计思想 .....	12
2.4 逻辑结构与物理结构 .....	14
2.5 开发平台 .....	16
2.6 系统的运行结果分析说明 .....	16
2.7 操作说明 .....	17
第三部分 实践总结.....	20
3.1. 所做的工作.....	20
3.2. 总结与收获.....	20
第四部分 参考文献.....	22

# 第一部分 算法实现设计说明

## 1.1 题目

几种排序：要求随机输入一组数据，随时给出某一趟排序的变化情况：

(1)直接插入排序、折半插入排序、希尔排序；

(2)冒泡排序、快速排序；

(3)简单选择排序

## 1.2 软件功能

由于需要将每一趟排序过程（包括比较、交换或插入等操作）以动画的形式便捷地展示，本人设计的软件基于**微信小程序平台**，主要功能如下：

- 选择排序方式（共 6 种）
- 输入待排序的数列，并对输入做错误处理
- 动画展示每一趟排序，排序结束后弹窗提示交换/插入总次数

此外，还在设计了程序的基本信息页面，可以从该页面联系开发者并给出错误报告/更新建议等。



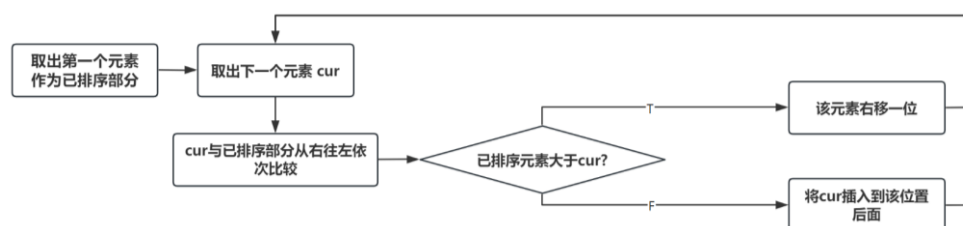
## 1.3 设计思想

根据题目要求，软件的设计主要聚焦两个方面：“**排序**”和“**动画**”：

出于微信小程序平台的开发要求，本软件的**排序算法**使用 JavaScript 语法编写，存储在每个排序页面对应的 .js 文件中，排序的数组来源于数据输入页面传送的数据。每种算法对应的设计思路和算法流程图如下：

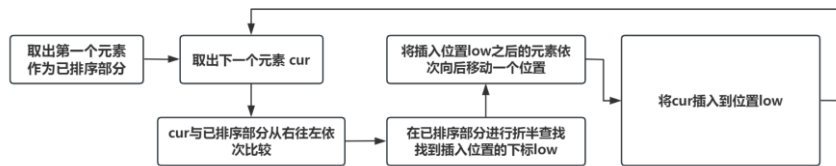
### 直接插入排序

1. 从第一个元素开始，将其视为已排序部分。
2. 取出下一个元素，将其插入到已排序部分的适当位置，使得插入后仍然保持有序。
3. 重复步骤 2，直到所有元素都插入到适当位置。



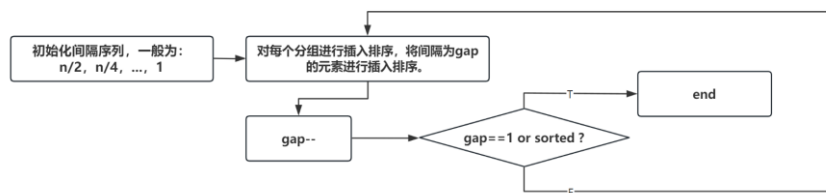
### 折半插入排序

和插入排序类似，但是寻找插入位置并不是遍历对调，而是使用折半查找（二分查找）的方式。



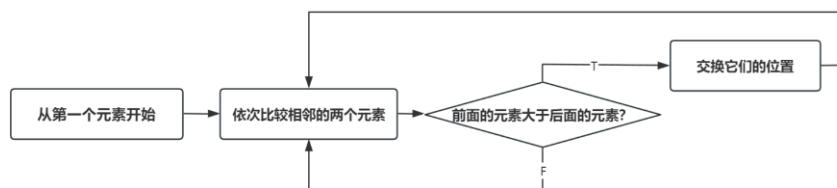
## 希尔排序

希尔排序是一种插入排序的改进算法，它通过定义一个间隔序列来分组进行插入排序，通过逐渐缩小间隔的方式，将较小的元素快速移到前面，从而减少后续插入排序的次数。



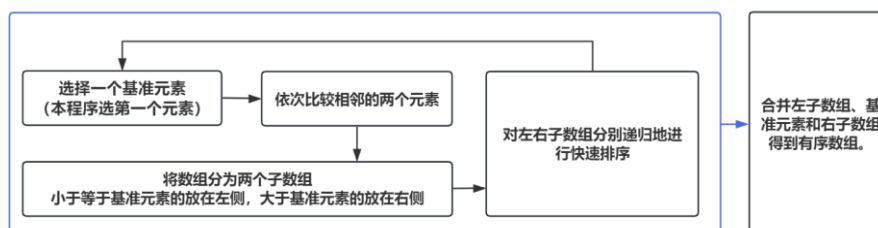
## 冒泡排序

主要思想为重复地比较相邻的两个元素，并根据大小关系交换位置，从而将较大（或较小）的元素逐渐“冒泡”到数组的一端。



## 快速排序

选择一个基准元素，将数组分成两个子数组，其中一个子数组的元素都小于基准元素，另一个子数组的元素都大于基准元素，然后对这两个子数组递归地进行快速排序，直到子数组只包含一个元素或为空，最终将所有子数组合并起来得到有序数组。



## 选择排序

每一轮选择都会确定一个最小（或最大）元素的位置，然后将其放到已排序的部分的末尾（或开头）。选择排序的时间复杂度为  $O(n^2)$ ，其中  $n$  是数组的长度。它是一种原地排序算法，不需要额外的空间。



除了排序算法的设计，更为重要的是**动画演示**的设计。动画设计的核心在于**延时动作**的

设计。在 JavaScript 中没有 C++ 中类似的 Sleep 函数，而是使用 setTimeout() 函数：

```
setTimeout(function() {  
    // 在延时后执行的代码  
}, 延时时间);
```

但是由于需要在 while 等循环中使用延时，将“延时后执行的代码”放在该函数执行内部较为不便。故本程序中设计封装了一个 sleep 函数：

```
43 sleep:function(time){  
44     var timeStamp = new Date().getTime();  
45     var endTime = timeStamp + time;  
46     while(true){  
47         if (new Date().getTime() > endTime){  
48             return;  
49         }  
50     }  
51 },
```

使用该函数可以执行和 C++ 中相同的 Sleep() 效果，单位为毫秒。

## 1.4 逻辑结构与物理结构

本程序中基于 JavaScript 实现算法，由于需要存储的数据为一组数据，故使用数组进行数据存储和处理。

```
输入框的值: (7) ["13", "2", "16", "5", "3", "9", "11"]  
0: "13"  
1: "2"  
2: "16"  
3: "5"  
4: "3"  
5: "9"  
6: "11"  
length: 7  
__proto__: Array(0)
```

### 逻辑结构：

- 数组是一种线性结构，其中的元素按照顺序排列，并且每个元素都有一个对应的索引。
- 数组逻辑结构是有序的，即元素的顺序是固定的，可以通过索引来访问和操作特定位置的元素。
- 数组逻辑结构可以表示为一个具有固定长度的有序集合，每个元素都与一个唯一的索引相关联。

### 物理结构：

- 在 js 中数组的物理结构通常是基于连续的内存空间，数组的元素在内存中存在连续存储位置。
- 数组的物理结构支持通过索引进行随机访问，因为可以根据元素的索引和内存地址计算出元素在内存中的位置。
- 数组的物理结构使得在已知索引的情况下，可以在常量时间复杂度内访问和修改元素。

## 1.5 开发平台

编程语言	WXML / WXSS / JavaScript
开发环境	微信开发者工具 Stable 1.06.2307250
版本控制系统	Git
算法测试工具	Jest（用于 js 算法代码测试）
应用测试工具	Selenium（用于小程序自动化测试）
性能测试工具	WebPageTest
开发设备硬件	MECHREV 极光 pro 2023 处理器 12th Gen Intel(R) Core(TM) i7-12650H 2.30 GHz 机带 RAM 16.0 GB 显卡 NVIDIA GeForce RTX 4050
运行环境	微信 Version $\geq$ 5.3.1

### 1.6 系统的运行结果分析说明

系统整体开发通过[微信开发者工具](#)进行，JavaScript 算法测试尝试使用 Jest 工具，程序整体自动化测试尝试使用 Selenium。

经过测试，程序在各种数据（负数/大数）下运行结果均正确。此外，程序具有健壮性，对输入数据进行纠错处理：

#### 1) 输入非数字

如下图（左 2），输入内容中有[任意符号](#)会导致弹窗**报错**，并拒绝跳转。但是如果输入的是[数字的不同形式](#)（如右图，小数、指数 E 形式等）则可以**通过**。



2) 输入无法绘制

由于输入的数据和后续绘图的 chart 柱长短有关，故考虑到美观性，要求最大数和最小数**差距比例**不能过大，否则会导致整体高度超过屏幕，而数字较小的柱子高度不明显。这里的“差距比例”通过以下公式计算：

$$(maxNum - minNum)/minNum$$

如下图，不符合要求会弹窗要求重新输入，并清空原有输入内容：






1.7 操作说明

由于工信部要求 2023/9/1 起小程序上线需备案，整体流程需要 3 个月时间，故本程序暂未上线，此处仅展示操作说明表：

首页	程序首页。可以跳转至任意一个排序页面和 About me 页面。	
----	----------------------------------	--

<p>数据输入 页面</p>	<p>从首页点击任意一个排序按钮,首先都会进入数据输入界面进行参数录入。相关错误处理上一段落已写明。</p>	
<p>直接插入 排序界面</p>	<p>点击“开始排序”即开启排序动画,红色标记表示当前处理位置,两个灰色标不断前移,表示每一次向前遍历比较和交换的过程。最后统计交换次数</p>	
<p>折半插入 排序界面</p>	<p>红色标记表示当前处理的位置,灰色标志表示折半寻找到的插入位置。最后会统计插入次数。</p>	



<p>希尔排序 界面</p>	<p>灰色标记表示 gap 左右的两个比较/交换位置，gap 会随着遍历不断缩小。最后展示交换总次数。</p>	
<p>冒泡排序 界面</p>	<p>两个相邻的灰色标记表示当前正在比较的两个相邻数。每一趟排序完成后点击“下一趟”会进入下一趟遍历，直到检测到排序完成。</p>	
<p>快速排序 界面</p>	<p>本程序中选用的快速排序的哨兵位为待排序序列的首元素，界面中用红色标记表示。灰色标记分别表示移动的左指针和右指针，条件满足即可交换，并进行下一次递归。</p>	

<p>选择排序 界面</p>	<p>灰色标记为当前遍历到的元素,红色标记表示本趟中找到的最小值,黑色表示基位置(待与最小值交换)。最后展示交换次数。</p>	
<p>程序信息 介绍页面</p>	<p>如有程序错误/异常,提供邮件联系方式。 助力每一个梦想 🤖🚀🎉</p>	

## 第二部分 综合应用设计说明

### 2.1 题目

★★ 参加奥运会有  $n$  个国家，各国编号为  $1\cdots n$ 。比赛分成  $m$  个男子项目，和  $w$  个女子项目。项目编号为男子  $1\cdots m$ ，女子  $m+1\cdots m+w$ 。不同的项目取前五名或前三名积分；取前五名的积分分别为：7、5、3、2、1，前三名的积分分别为：5、3、2；哪些取前五名或前三名由学生自己设定。

- (1) 可以输入各个项目的前三名或前五名的成绩；
- (2) 能统计各国总分，
- (3) 可以按各国编号、各国总分、男女团体总分排序输出；
- (4) 可以按各国编号查询国家某个项目的情况；可以按项目编号查询取得前三或前五名的国家。

### 2.2 软件功能

该软件主要功能奥运会成绩追踪和分析，主要涉及数据的输入、输出和排序，不需要过多的动画等图形化效果，因此采用的技术实现方法是基于 **python** 开发的 **PyQt5** 窗体程序。



以下列表展示具体的功能项：

- a) 输入国家总数 / 男子项目数 / 女子项目数
- b) 设定项目选取前三名积分/前五名积分方式
- c) 输入成绩（按照“国家编号 项目编号 名次 积分”）
- d) 统计各国积分
- e) 按总分排序输出
- f) 按国家编号排序输出
- g) 按男子团体总分排序输出
- h) 按女子团体总分排序输出
- i) 查询国家项目情况
- j) 查询项目前三/五名的国家

下表展示具体**功能组件**和实现方式：

创建窗口、布局、按钮、标签、输入字段和其他 GUI 组件的类和函数（ <b>菜单</b> 、 <b>工具栏</b> 和 <b>中央窗口部件</b> 区域）	QMainWindow
创建包含所有其他 GUI 组件的 <b>主窗口部件</b>	QWidget
垂直和水平布局的类（GUI 组件按照 <b>垂直</b> 或 <b>水平</b> 方向进行排列）	QVBoxLayout QHBoxLayout
<b>按钮</b> 组件（触发特定操作或事件的响应）	QPushButton
<b>标签</b> 组件（显示文本或图像）	QLabel
<b>文本输入框</b> 组件（接收用户的文本输入）	QLineEdit
<b>列表</b> 组件（显示列表形式的数据）	QListWidget
<b>消息框</b> 组件（显示消息、警告或错误信息）	QMessageBox
<b>对话框</b> 组件（接收用户的输入）	QInputDialog

2.3 设计思想

根据题目要求，软件是一个使用 **PyQt5** 设计的具有 **GUI** 的应用程序窗口，使用 **QListWidget** 存储和管理成绩信息，并实现添加、删除、查询、排序和统计等操作。以下是具体的设计项：

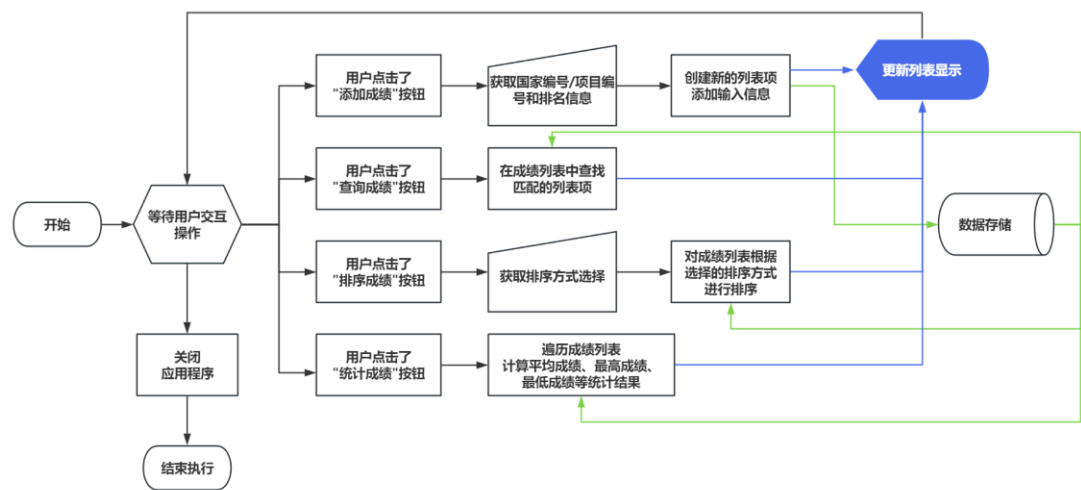
1) 数据结构选择

数据结构的选择主要围绕着存储和管理奥运会成绩信息展开，本程序选用

QListWidget 组件作为显示成绩信息的列表。

每个列表项可以包含一个成绩条目，其中包括国家编号和积分成绩等。而 QListWidget 提供 api 方法实现添加、删除和修改列表项，同时可以直接在列表项中显示文本。

2) 程序逻辑



3) 算法设计流程

导入必要的模块和库	导入 sys 和 PyQt5.QtWidgets 模块，这些模块包含了实现图形用户界面(GUI)所需的类和函数。
创建 OlympicsApp 类	定义了一个名为 OlympicsApp 的类，继承自 QMainWindow。该类用于创建奥运会成绩统计的应用程序窗口。
初始化函数	OlympicsApp 类中定义了一个 __init__ 初始化函数。该函数接受 n、m 和 w 作为参数，这些参数分别表示国家数量、男子项目数量和女子项目数量。在初始化函数中，设置了字典和列表，用于存储数据和计算结果。
初始化用户界面	initUI 方法中，设置了应用程序窗口的标题、大小和布局。使用 QVBoxLayout 布局管理器创建了垂直布局，并添加了各种 GUI 元素，如标签、按钮、文本框和列表框。
功能按钮的点击事件处理函数	各个按钮（如输入成绩、设定项目、统计各国总分等）连接了相应的点击事件处理函数，以实现相关功能。
输入成绩	enter_scores 函数处理输入成绩按钮的点击事件。它从输入框中获取输入的成绩信息，并进行格式验证和范围检查，然后将成绩数据存储到相应的数据结构中。
设定项目前三/五名	setup_project 函数处理设定项目按钮的点击事件。它弹出一个对话框，让用户输入项目编号，并选择计分规则（前三名或前五名）。然后将项目编号和计分规则存储到 project_setup 字典中。
统计各国总分	calculate_scores 函数处理统计各国总分按钮的点击事件。它遍历存

	储的成绩数据，根据设定的计分规则 <b>计算</b> 每个国家的总分，并将结果存储到 <code>country_scores</code> 字典中。
按总分排序输出	<code>sort_countries</code> 函数处理按总分排序输出按钮的点击事件。它将各国总分按 <b>降序排序</b> ，并将结果显示在列表框中。
按国家编号排序输出	<code>sort_countries_by_id</code> 函数处理按国家编号排序输出按钮的点击事件。它按国家编号对数据进行排序，并将结果显示在列表框中。
按男子团体总分排序输出	<code>sort_countries_by_male_scores</code> 函数处理按男子团体总分排序输出按钮的点击事件。它按男子团体总分对数据进行排序，并将结果显示在列表框中。
按女子团体总分排序输出	<code>sort_countries_by_female_scores</code> 函数处理按女子团体总分排序输出按钮的点击事件。它按女子团体总分对数据进行排序，并将结果显示在列表框中。
查询国家项目情况	<code>query_country</code> 函数处理查询国家项目情况按钮的点击事件。它从输入框中 <b>获取国家编号</b> ，并 <b>查询</b> 该国家在各个项目中的成绩情况，并将结果显示在列表框中。
查询项目前三/五名国家	<code>query_project</code> 函数处理查询项目前三/五名国家按钮的点击事件。它从输入框中 <b>获取项目编号</b> ，并 <b>查询</b> 在该项目中获得前三/五名的国家，并将结果显示在列表框中。
应用程序启动	在 <code>__main__</code> 函数中，创建了一个 <code>QApplication</code> 实例和 <code>OlympicsApp</code> 实例，并启动了应用程序的 <b>事件循环</b> 。

#### 4) GUI 布局和组件设计

使用 **PyQt5** 库创建了一个图形用户界面 (GUI)，主要使用 `QMainWindow`、`QWidget`、`QVBoxLayout`、`QHBoxLayout`、`QPushButton`、`QLabel`、`QLineEdit`、`QListWidget`、`QMessageBox` 和 `QInputDialog` 等组件来构建界面。

## 2.4 逻辑结构与物理结构

本软件基于 python 开发，在主类 `OlympicsApp` 中使用了**字典/列表**两种数据结构。

```

6 class OlympicsApp(QMainWindow):
7     def __init__(self, n, m, w):
8         super().__init__()
9
10        self.n = n
11        self.m = m
12        self.w = w
13        # Initialize the dictionaries first
14        self.data = {}
15        self.country_scores = {}
16        self.country_male_scores = {}
17        self.country_female_scores = {}
18        self.sorted_data = []
19        self.project_setup = {}
20
21        # 计算国家得分字典
22        for i in range(1, n + 1):
23            self.country_male_scores[i] = 0
24            self.country_female_scores[i] = 0
25
26        self.initUI()

```

## a) 字典

通过散列表和哈希函数的组合，Python 的字典能够以平均常数时间复杂度  $O(1)$

进行键的查找、插入和删除操作，使得字典在处理大量数据时具有高效的性能。

逻辑结构	<ul style="list-style-type: none"><li>● <b>键 (Key)</b>: 字典中的键是<u>唯一</u>的，并且用于标识和访问对应的值。键可以是<u>不可变</u>的数据类型，例如字符串、数字或元组，但不能是可变的数据类型，例如列表或字典本身。</li><li>● <b>值 (Value)</b>: 字典中的值与键相关联，可以是<u>任意</u>的数据类型，包括基本类型（如整数、字符串、布尔值）和复杂类型（如列表、字典、对象等）。</li><li>● <b>键值对 (Key-Value Pairs)</b>: 字典中的每个元素都由键和对应的值组成，形成了键值对的<u>映射</u>关系。</li></ul>
物理结构	<ul style="list-style-type: none"><li>● <b>散列表 (Hash Table)</b>: 字典的物理结构基于散列表实现。散列表是一种使用哈希函数将键映射到存储位置的数据结构。Python 的字典使用散列表来实现快速的键查找和插入操作。</li><li>● <b>桶 (Bucket)</b>: 散列表中的<u>存储位置</u>被称为桶。每个桶可以存储一个或多个<u>键值对</u>。当发生哈希碰撞（两个不同的键映射到相同的桶）时，Python 的字典使用<u>链表</u>（或其他方法，如红黑树）来解决冲突，将多个键值对链接在同一个桶中。</li><li>● <b>哈希函数 (Hash Function)</b>: 哈希函数用于将键映射到散列表中的桶。Python 使用内置的哈希函数来计算键的哈希值，哈希值决定了键在散列表中的存储位置。</li></ul>

## b) 列表

通过数组和指针的组合，Python 的列表能够以常数时间复杂度  $O(1)$  进行索引

访问、插入和删除操作，使得列表在处理元素集合时具有高效的性能。同时，由于

数组的连续内存分配，列表还支持通过索引进行快速的[随机访问](#)。

逻辑结构	<ul style="list-style-type: none"><li>● <b>元素 (Elements)</b>: 列表由多个元素组成，每个元素可以是任意的数据类型。元素在列表中的位置是<u>有序</u>的，可以通过<u>索引</u>访问和操作。</li><li>● <b>索引 (Index)</b>: 列表中的每个元素都有<u>一个</u>对应的索引，用于标识元素在列表中的位置。索引从 <math>0</math> 开始，依次递增，可以用于访问和修改列表中的元素。</li></ul>
物理结构	<ul style="list-style-type: none"><li>● <b>数组 (Array)</b>: 列表的物理结构基于数组实现。数组是一块<u>连续</u>的内存空间，用于存储<u>相同类型</u>的元素。Python 的列表通过数组来存储元素，并使用<u>动态数组</u>技术来实现可变长度。</li><li>● <b>大小 (Size)</b>: 列表的大小指的是列表中存储的<u>元素个数</u>。列表的大小可以根据需要动态调整，当元素数量超过数组的容量时，Python 会自动分配更大的内存空间，并将原有的元素复制到新的内存中。</li><li>● <b>指针 (Pointers)</b>: 为了支持动态长度和快速的元素访问，Python 的列表使用指针来<u>追踪</u>数组的起始位置和元素的位置。指针指向数组中的实际元素，使得可以通过索引快速定位元素。</li></ul>

## 2.5 开发平台

编程语言	<b>Python 3.11.2 虚拟环境</b> 
开发环境	Visual Studio Code 1.59.1
依赖库	<b>sys / PyQt5</b>
版本控制系统	Git
算法测试工具	pytest（用于 py 算法代码测试）
开发设备硬件	<b>MECHREV 极光 pro 2023</b> 处理器 12th Gen Intel(R) Core(TM) i7-12650H 2.30 GHz 机带 RAM 16.0 GB 显卡 NVIDIA GeForce RTX 4050
运行环境	<b>Windows 操作系统</b> （已将 Python 解释器和依赖库等一起打包，创建自包含运行环境的 .exe 可执行程序）

注：

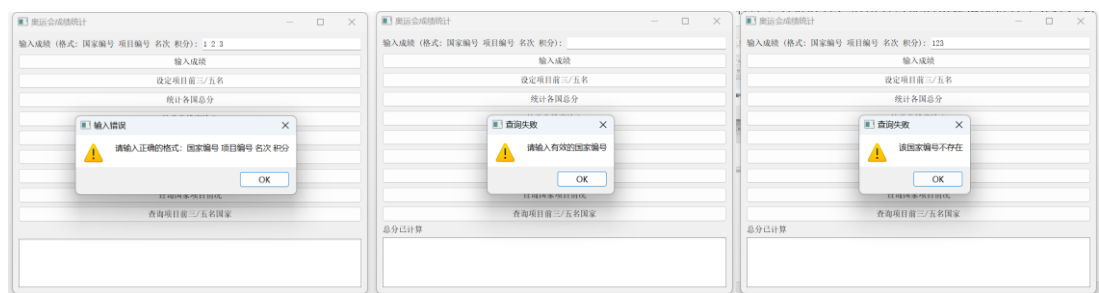
本程序使用的 PyQt5 库并非 Qt 官网主页的 Qt for Python，两者都基于 Qt 框架，区别在于 PyQt5 使用了商业许可和 GPL 许可的双重许可模型，而 Qt for Python(PySide2) 使用了 LGPL 许可。由于 PyQt5 有更丰富成熟文档和学习资源，本程序选择使用该库进行开发。

## 2.6 系统的运行结果分析说明

系统整体开发通过 Visual Studio Code (1.59.1) 进行，Python 算法测试尝试使用 pytest 工具。

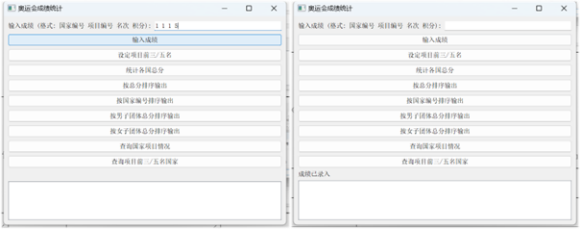
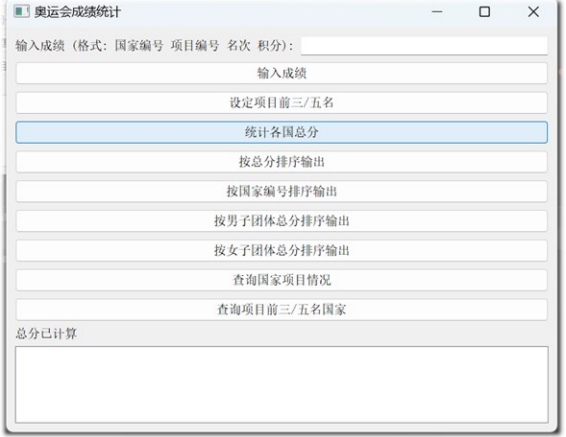


经过测试，程序具有健壮性，对所有可能的错误情况进行了处理，下图展示部分报错：

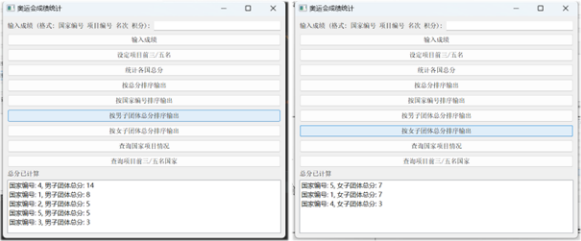
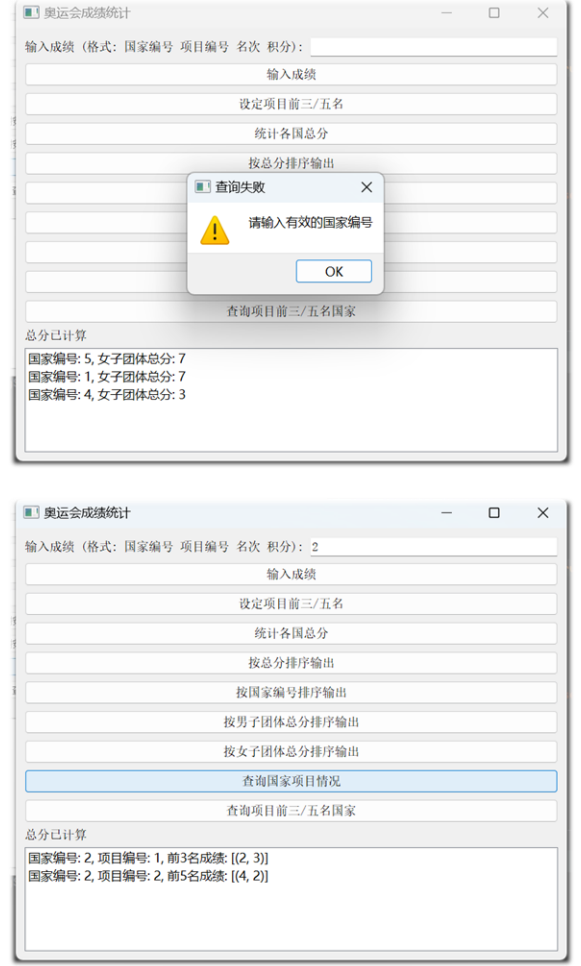
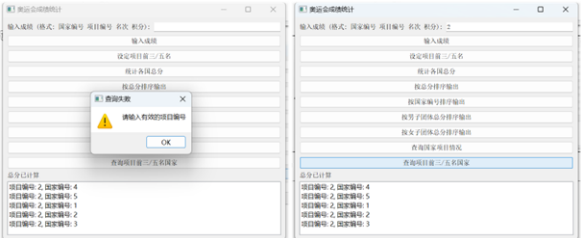




## 2.7 操作说明

1	输入参加奥运会的国家 (n)	
2	输入男子项目数量 (m)	
3	输入女子项目数量 (w)	
4	进入程序主页	
5	设定项目积分制度	

6	成绩录入	
7	统计各国总分	
8	按总分排序输出	
9	按国家编号排序输出	

10	按男子/女子团体总分 排序输出	
11	查询国家项目情况	
12	查询项目前三/五的国家	

## 第三部分 实践总结

### 3.1. 所做的工作

- 1) 复习数据结构的基础知识，虽然后续没有用到，但还是重点复习了树、图等章节的知识点。
- 2) 回顾微信小程序开发，包括 WXML/WXSS/JavaScript 的使用。
- 3) 加强 JavaScript 算法能力，尝试使用 js 编写各种排序算法。
- 4) 学习封装 js 中的延时操作，并和 C++ 中的延时函数底层原理进行比较。
- 5) 尝试微信小程序的上传、部署和上线。
- 6) 了解 Qt 基础知识，比较 PyQt5 和 Qt for Python 的区别。
- 7) 加强 Python 算法能力，尝试使用 py 编写存储和分析的排序算法
- 8) 学习 python 应用程序打包成可执行文件。
- 9) 学习并使用基本测试工具，如 Jest/ Selenium/ WebPageTest/ pytest。
- 10) 学习并使用版本控制系统 Git，虽然是独立开发但还是尝试使用其功能。
- 11) 课设文档的撰写。

### 3.2. 总结与收获

自我学习能力方面，不断尝试了多种业界常用的技术方向，如微信小程序开发、软件测试、算法单元测试、应用程序打包和使用版本控制系统等。此外对于 js 有了更深的了解，对于 PyQt5 库也能熟练掌握。总结下来，对于一门新技术的学习，首先要寻找其与已掌握技术的相通之处，如 PyQt 库就是基于 py 的窗体实现和 GUI 展示库，在掌握 py 语法的基础上可以跳过一些基础内容的学习，而转向更高阶内容的学习。此外，对于短学习周期来说，直接看 GitHub 已有项目的源代码可以起到更快的学习效果，此外 Chatgpt 虽然在编写整段项目代码时会错误百出，但在答疑解惑方面也有不错的能力，可以用来辅助学习，也可以用于规划学习路线等。总之，对于短周期技术学习来说，从头到尾看完整的视频、看完整的书籍的方式会导致极低的学习效率，而项目实践+源代码学习+Gpt 辅助的学习方法能更快掌握所需知识。

课程设计的收获方面，数据结构课程是后续所有计算机专业课的基础，“数据结构”是对数据的操作，主要涉及组织和管理数据（什么情况下使用什么数据结构来存储和管理）/算法设计和分析（如何使用更高效的算法处理数据）/提高程序性能（学习了复杂度的概念

和分析方法)等。而课程设计就是以上作用的实现,无论是排序算法的动态模拟,还是奥运会积分管理系统,本质上都是对数据的存储、处理,课程设计的收获就在于选择使用更好的存储方式和复杂度更低的处理算法。

个人收获方面,除了知识的学习,还尝试了和同学一起学习的学习方式,通过互相补充,可以提高学习效率。此外,通过实践真正让 `chatgpt` 成为学习和开发的助手——`gpt` 暂时不能替代开发,但是可以在答疑解惑、规划学习路线方面起到很好的作用。

## 第四部分 参考文献

- [1] 任路顺作.PyQt 编程快速上手 Python GUI 开发从入门到实践[M].北京：人民邮电出版社,2023
- [2] 明日科技编著.PyQt 从入门到精通[M]. 北京：清华大学出版社, 2020
- [3] 沈顺天. 微信小程序项目开发实战[M]. 北京：机械工业出版社, 2020
- [4] 大卫·弗拉纳根. JavaScript 指南 原书第 7 版[M]. 北京：机械工业出版社, 2021
- [5] Cherny, Boris. Programming Typescript: Making Your JavaScript Applications Scale[M]. O'Reilly Media, 2019
- [6] Yang Liu;Yang Ou;Wenhan Chen;Zhiguang Chen;Nong Xiao. LazySort: A customized sorting algorithm for non-volatile memory[J]. Information Sciences, 2023, Vol. 641: 119137
- [7] 岑远红, 李娟编. Python 程序设计[M]. 重庆：重庆大学出版社, 2022
- [8] 王剑, 田卫红, 邓力夫主编; 刘造新主审. 面向对象程序设计[M]. 西安：西北工业大学出版社, 2021
- [9] 李增刚, 沈丽编. Qt for Python PySide6GUI 界面开发详解与实例[M]. 北京：清华大学出版社, 2022
- [10] HANDS-ON QT FOR PYTHON DEVELOPERS[S].
- [11] 刘斌, 赵艳红, 李志芳. 数据结构与算法[M]. 上海：上海交通大学出版社, 2022
- [12] 裘宗燕著. 数据结构与算法 PYTHON 语言描述 第 2 版[M]. 北京：机械工业出版社, 2022