**作业 HW5 实验报告**

1. **涉及数据结构和相关背景**

本次选做的四道题目—和有限的最长子序列、二叉排序树、换座位、哈希表主要涉及了以下

相关知识：

* + - 1. 二分查找：

必须是对有序数据进行处理。它通过每次将查找范围缩小一半来快速定位目

标元素。二分查找的实现依赖于递归或迭代地计算中间位置，并将目标值与中间元素进行比较。如果目标值小于中间元素，则继续在左半部分查找；如果大于，则在右半部分查找。这种方法的时间复杂度为O(log n)，是查找有序数据的最优方法之一。

* + - 1. 二叉排序树BST：

二叉排序树是一种特殊的二叉树，其中每个节点的左子树只包含小于节点值的元素，而右子树只包含大于节点值的元素。这种结构使得查找、插入和删除操作都能以平均O(log n)的时间复杂度完成。然而，在最坏情况下（如树退化为链表），这些操作的时间复杂度会上升到O(n)。因此，在实际应用中，通常会和二叉平衡树一起使用。

* + - 1. 哈希表：

哈希表是一种基于哈希函数实现的数据结构，它能够在平均情况下以O(1)的时间复杂度完成查找、插入和删除操作。哈希表通过将键映射到表中的位置来存储数据，这种映射关系由哈希函数确定。有时会涉及哈希冲突的处理（比如二次探测等）。

以及之前的树等数据结构知识。

**2. 实验内容**

**2.1 和有限的最长子序列**

**2.1.1 问题描述**

给一个长度为n的整数数组nums和一个长度为m的整数数组queries，返回一个长度为m的数组answer，其中answer[i]是nums中元素之和小于等于queries[i]的子序列的最大长度。

子序列是由一个数组删除某些元素（也可以不删除）但不改变元素顺序得到的一个数组。

**2.1.2 基本要求**

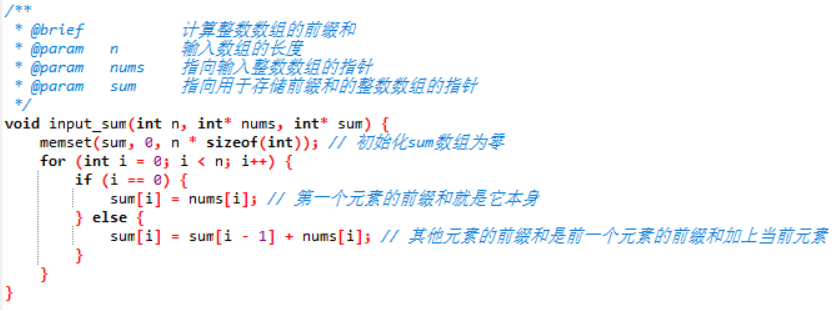
输入：第一行包括两个整数n和m，分别表示数组nums和queries的长度；第二行包括n个整数，为数组nums中元素；第三行包含m个整数，为数组queries中元素；对于20%的数据，有1<=n,m<=10；对于40%的数据，有1<=n,m<=100；对于100%的数据，有1<=n,m<=1000；对于所有数据，1<=nums[i],queries[i]<=106

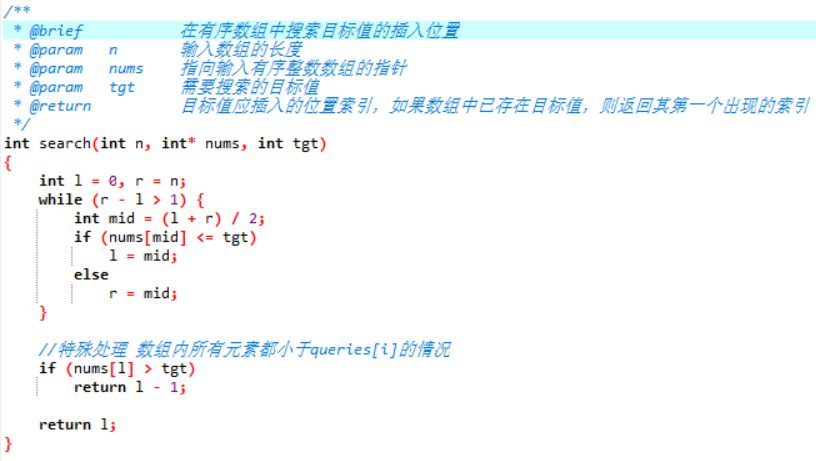
输出：输出一行，包括m个整数，为answer中元素

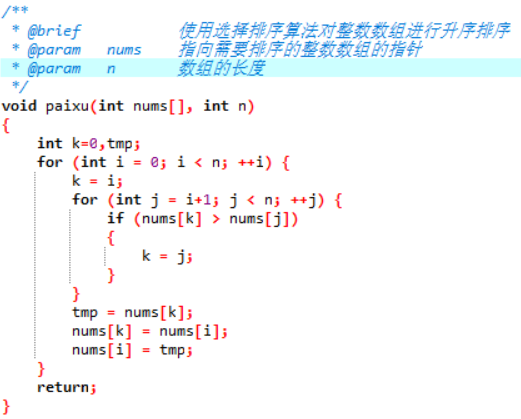
**2.1.3 数据结构设计**

主要涉及的是数组，无需自己定义，直接使用即可。

**2.1.4功能说明（函数、类）**





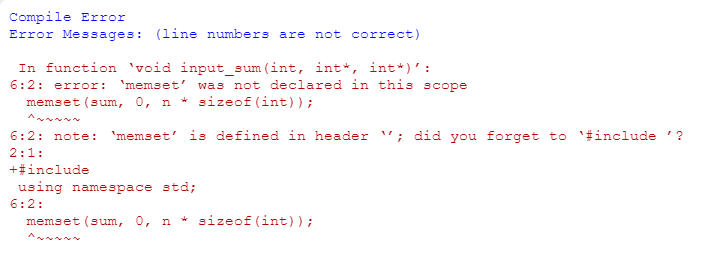


**2.1.5 调试分析（遇到的问题和解决方法）**

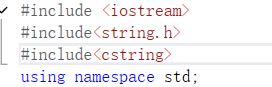
调试过程描述：

1. 整体写完程序后运行，根据编译器报的错误修改程序中的语法错误。
2. 生成可执行文件，根据题目中给出的测试数据生成文件，利用输入输出重定向的方法，检测程序的输出结果是否正确。以此来验证函数逻辑是否正确。

问题：



解决方案：

加上<string.h><cstring>的头文件即可

**2.1.6 总结和体会**

本题整体上来说是简单题，但有一个很巧妙的点是，和有限的最长子序列由最小的前几个数组成。所以nums本身的元素次序对结果没有影响，所以要对nums进行排序。想到这一点，这个题目就很容易解决了。

**2.2 二叉排序树**

**2.2.1 问题描述**

二叉排序树BST（二叉查找树）是一种动态查找表，基本操作集包括：创建、查找，插入，删除，查找最大值，查找最小值等。

本题实现一个维护整数集合（允许有重复关键字）的BST，并具有以下功能：1. 插入一个整数 2.删除一个整数 3.查询某个整数有多少个 4.查询最小值 5. 查询某个数字的前驱（集合中比该数字小的最大值）。

**2.2.2 基本要求**

输入：第1行一个整数n，表示操作的个数；接下来n行，每行一个操作，第一个数字op表示操作种类：

若op=1，后面跟着一个整数x，表示插入数字x

若op=2，后面跟着一个整数x，表示删除数字x（若存在则删除，否则输出None，若有多个则只删除一个），

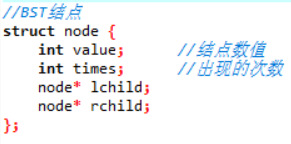
若op=3，后面跟着一个整数x，输出数字x在集合中有多少个（若x不在集合中则输出0）

若op=4，输出集合中的最小值（保证集合非空）

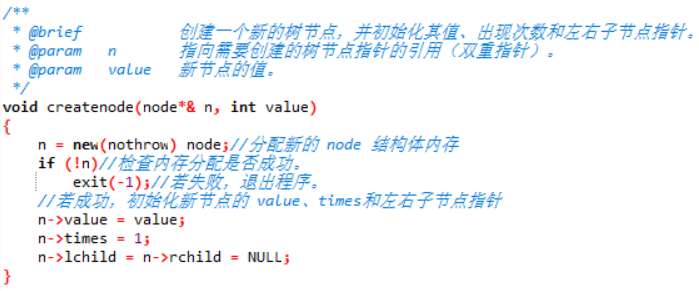
若op=5，后面跟着一个整数x，输出x的前驱（若不存在前驱则输出None，x不一定在集合中）

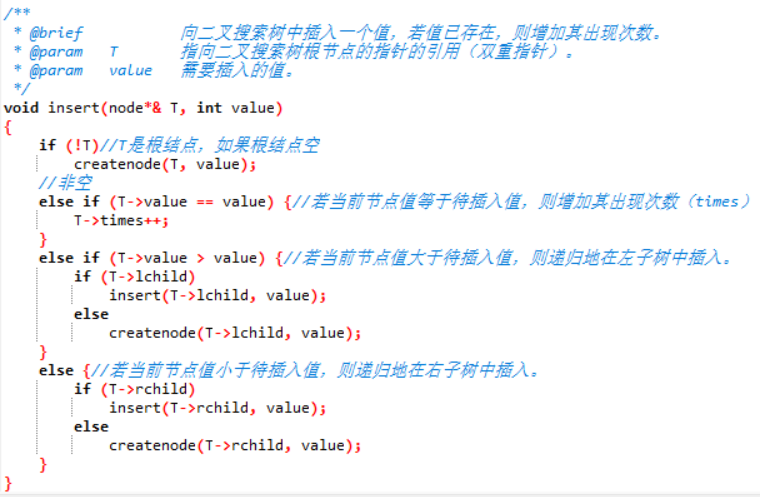
输出：一个操作输出1行（除了插入操作没有输出）

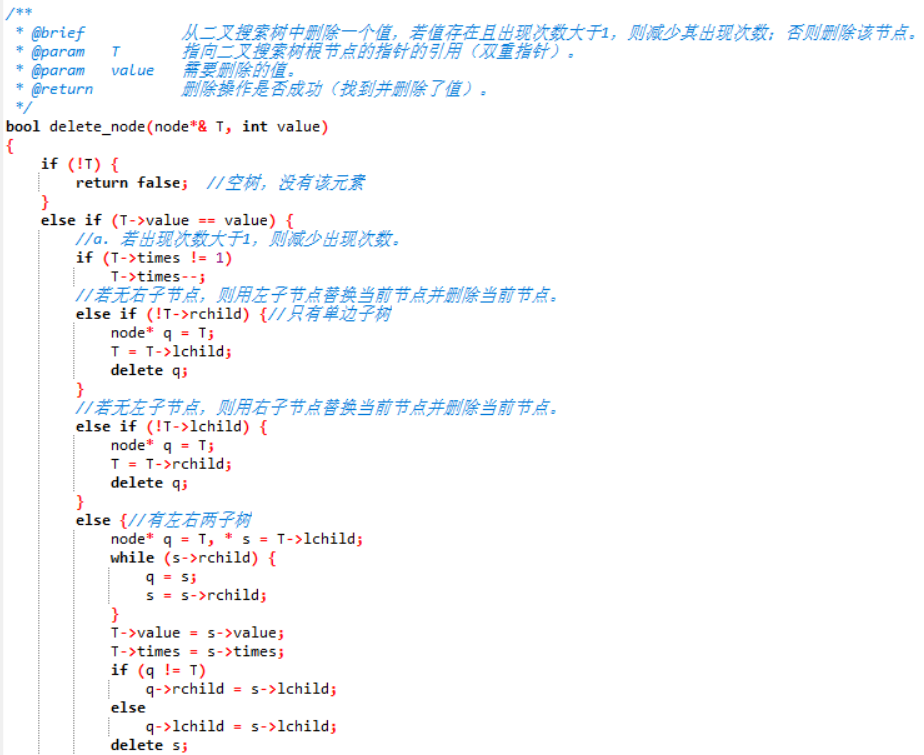
**2.2.3 数据结构设计**

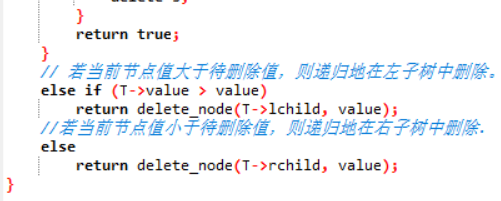
****

**2.2.4功能说明（函数、类）**

****

****

****

****

**2.2.5 调试分析（遇到的问题和解决方法）**

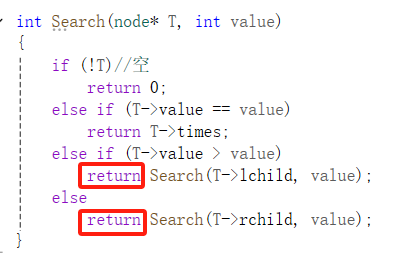
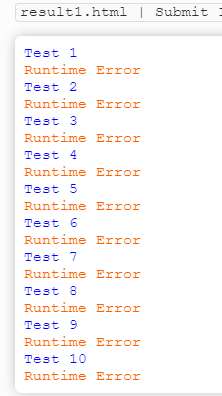
调试过程描述：

1.整体写完程序后运行，根据编译器报的错误修改程序中的语法错误。

2.生成可执行文件，根据题目中给出的测试数据生成文件，利用输入输出重定向的方法，检测程序的输出结果是否正确。以此来验证函数逻辑是否正确。

问题：提交到oj系统上后出现runtime error问题，但明明测试了几组数据都是对的，为什么？（如下图）

解决方案：由于编译器的不同（我用的VStudio），在VS上，我的Search函数忘记加return是正确的，但是在oj平台就会报错，添上return就好了。

**** ****

**2.2.6 总结和体会**

本题主要考察了二叉排序树的相关知识，既涉及了之前所学的树的相关知识，又涉及了最近所学的查找知识，综合性较强。但细细分解下来，也还是比较简单的。但根据函数的具体实现，不难看出delete操作是最复杂的，因为要考虑删除结点后依然保持排序树的有序性。通过这道题的练习，也算是加强了我对具体操作的代码实现的理解吧。

**2.3 和有限的最长子序列**

**2.3.1 问题描述**

本题针对字符串设计哈希函数。假定有一个班级的人名名单，用汉语拼音（英文字母）表示。

首先把人名转换成整数，采用函数h(key)=((...(key[0] \* 37+key[1]) \* 37+...)\*37+key[n-2] )\* 37+key[n-1]，其中key[i]表示人名从左往右的第i个字母的ascii码值(i从0计数,字符串长度为n，1<=n<=100)。

采取除留余数法将整数映射到长度为P的散列表中，h(key)=h(key)%M，若P不是素数，则M是大于P的最小素数，并将表长P设置成M。

采用平方探测法（二次探测再散列）解决冲突。（有可能找不到插入位置，当探测次数>表长时停止探测）

注意：第1步计算h(key)时得到的整数可能很大，需要采用数据类型usigned long long int存储，产生的溢出不需处理，其结果相当于对2^64取模的结果。

**2.3.2 基本要求：**

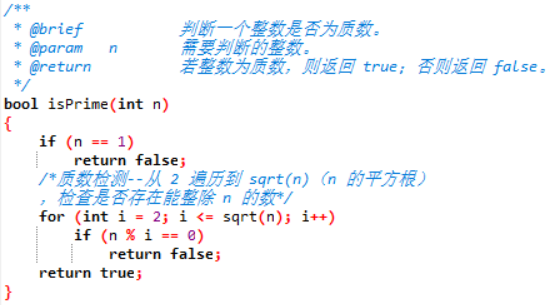
输入: 第1行输入2个整数N、P，分别为待插入关键字总数、散列表的长度。若P不是素数，则取大于P的最小素数作为表长；第2行给出N个字符串，每一个字符串表示一个人名

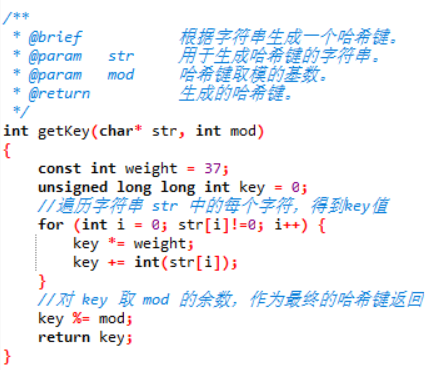
输出：在1行内输出每个字符串插入到散列表中的位置，以空格分割，若探测后始终找不到插入位置，输出一个'-'。

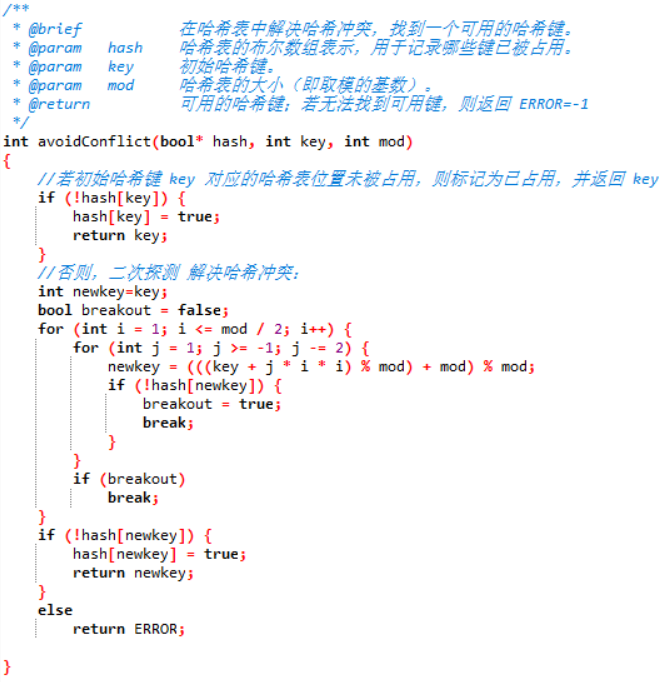
**2.3.3 数据结构设计**

本题主要的是数组，未自定义结构体、类等数据结构。

**2.3.4功能说明（函数、类）**







**2.3.5 调试分析（遇到的问题和解决方法）**

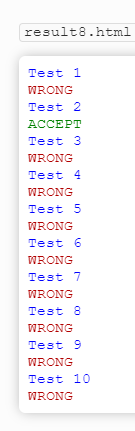
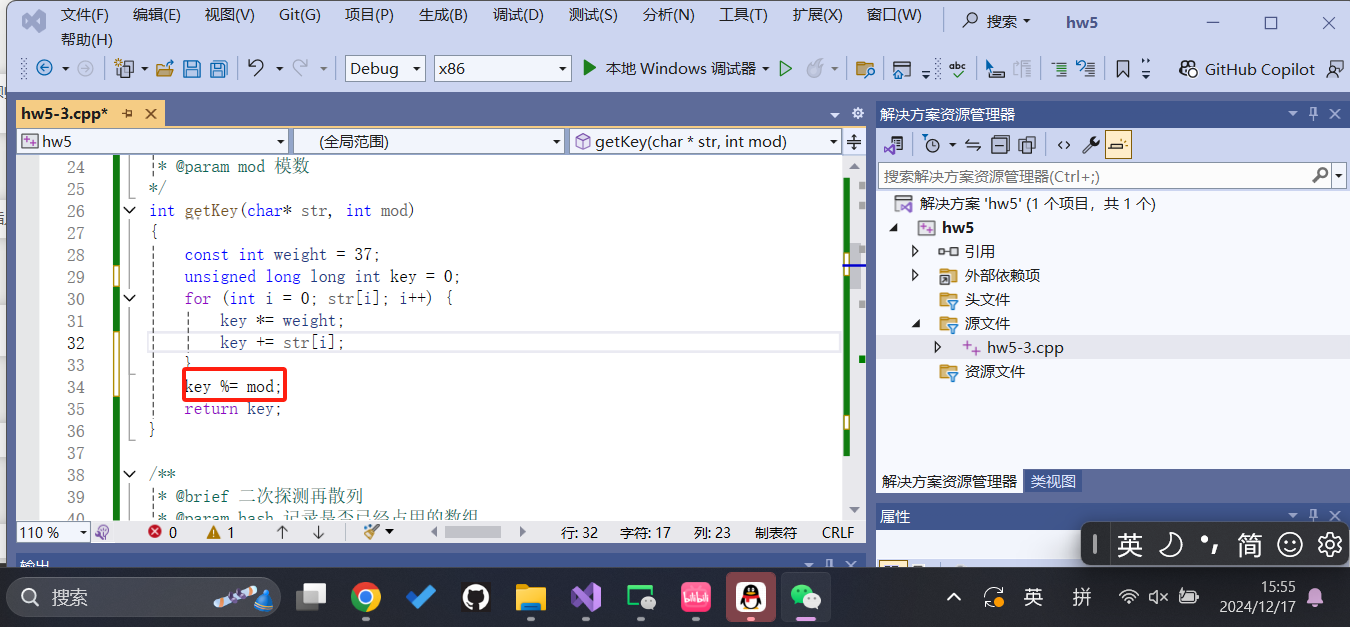
调试过程描述：

1.整体写完程序后运行，根据编译器报的错误修改程序中的语法错误。

2.生成可执行文件，根据题目中给出的测试数据生成文件，利用输入输出重定向的方法，检测程序的输出结果是否正确。以此来验证函数逻辑是否正确。

问题：测试了几组数据，提交后却还是不通过。

解决方案：其实是因为我偷懒只测试了几组简单的数据，简单数据可以，但是数据一旦计算出来的key值太大，就会反应出程序的逻辑问题。经过反复查看，发现是key值取模的执行语句放错了位置，应该放在如图所示位置，但我前面的程序都是放在了循环里。

****  ****

**2.3.6 总结和体会**

本题主要基于哈希函数，考察了哈希冲突的解决方法之一—二次探测。通过这道题目，加深了我对哈希函数本身执行逻辑的理解，也加深了我对解决哈希冲突的方法的理解。

我想这道题的难点就在于题目中给出的提示“第1步计算h(key)时得到的整数可能很大，需要采用数据类型usigned long long int存储”，如果没有这句提示，其实很难查出程序不通过的错误。

**2.4 换座位**

**2.4.1 问题描述**

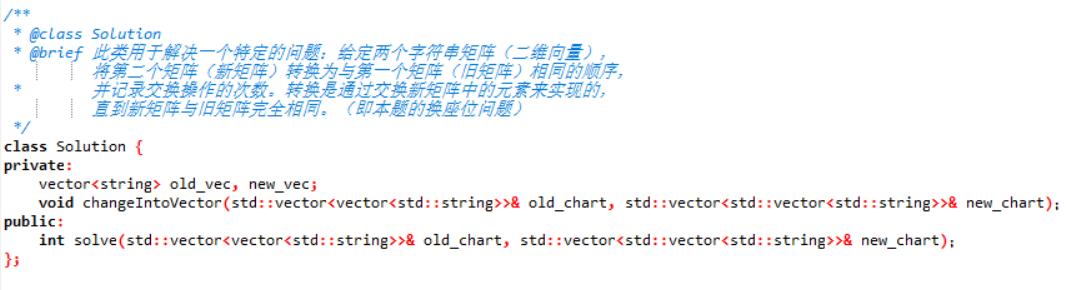
期末考试，监考老师粗心拿错了座位表，学生已经落座，现在需要按正确的座位表给学生重新排座。假设一次交换可以选择两个学生并让他们交换位置，给出原来错误的座位表和正确的座位表，问给学生重新排座需要最少的交换次数。

**2.4.2 基本要求**

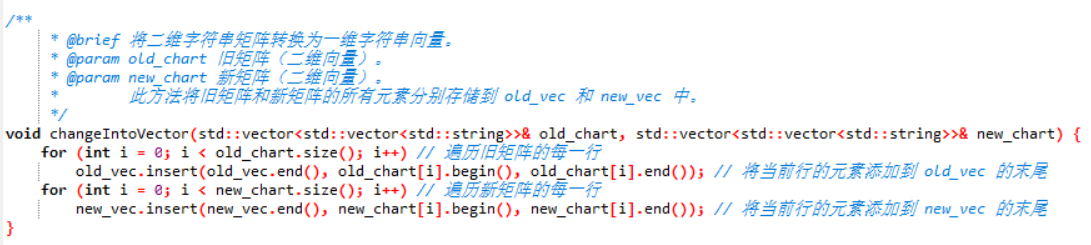
输入：两个n\*m的字符串数组，表示错误和正确的座位表old\_chart和new\_chart，old\_chart[i][j]为原来坐在第i行第j列的学生名字；对于100%的数据，1<=n,m<=200；人名为仅由小写英文字母组成的字符串，长度不大于5

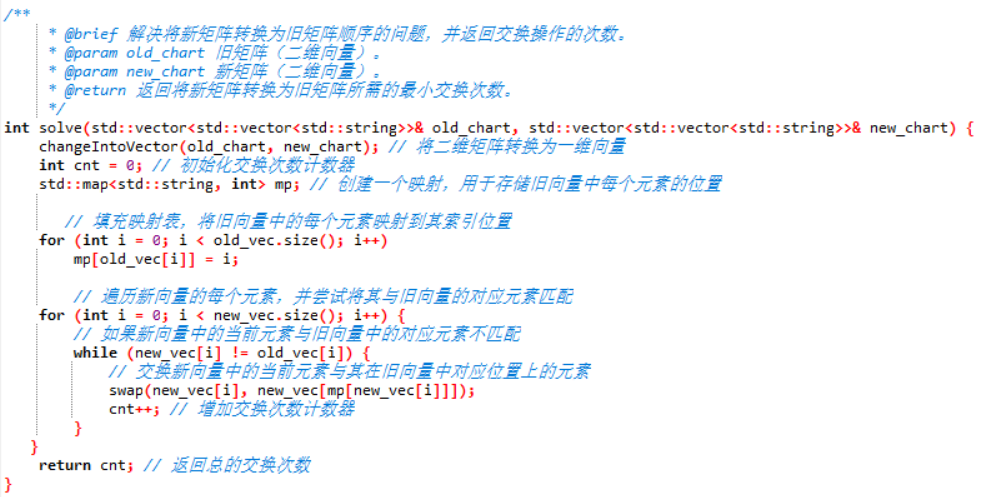
输出：一个整数，表示最少交换次数

**2.4.3 数据结构设计**

****

**2.4.4功能说明（函数、类）**

****

****

**2.4.5调试分析（遇到的问题和解决方法）**

调试过程描述：

1.整体写完程序后运行，根据编译器报的错误修改程序中的语法错误。

2.生成可执行文件，根据题目中给出的测试数据生成文件，利用输入输出重定向的方法，检测程序的输出结果是否正确。以此来验证函数逻辑是否正确。

问题：测试的时候通过，oj上没通过

解决方案：依旧是因为只测试了几组简单数据，当数据量大的时候，程序的逻辑问题就会显现出来。排查出逻辑问题即可。

体会：测试数据一定一定要有代表性！！！！

**2.4.6 总结和体会**

本题n最大能够达到200，意味着座位总数最多为40000个，所以如何快速查找就是要考虑的问题，也是与查找相关联的地方。因此采用哈希表来快速查找学生的位置，以此提高算法的效率。

**3. 实验总结**

本次实验主要围绕查找的相关知识，设置了一系列情景来运用这些知识。通过本次实验的练习，加深了我对各种查找算法的理解，并且在不同实际问题下选择不同的算法的过程中，也加深了我对这些不同查找的时间复杂度等特点的理解，更好地学以致用。通过复杂的实际问题，选择合适的数据结构，提高了自身的编程能力。