学号 专业 信息安全 姓名

实验日期 **2024/05/08** 教师签字 成绩

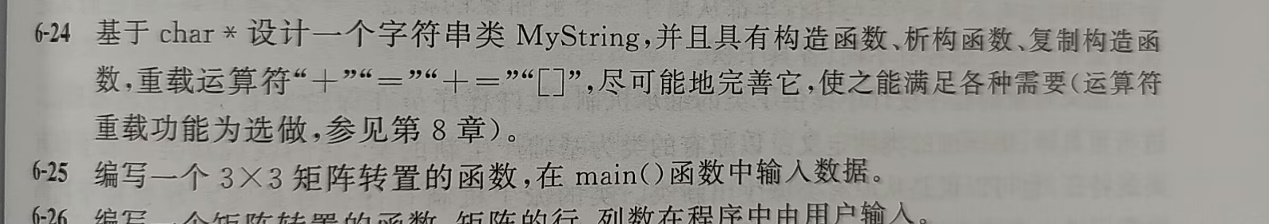
实验报告

【实验名称】 字符串类和矩阵转置

【实验目的】

本次实验旨在掌握C++中字符串类的使用，以及如何通过编程实现矩阵的转置。实验要求学生熟悉字符串类的常用方法，如连接、分割、查找等，并且能够编写程序实现矩阵的转置操作。

【题目】



6-24和6-25

**【实验原理】**

1. **字符串类的使用：**

在C++中，string类是一个非常有用的字符串处理工具。它提供了许多方便的方法来操作和处理字符串。以下是一些常用的string类方法：

## 1. 构造函数

string类提供了多种构造函数，可以根据不同的需求创建字符串对象。例如，可以使用默认构造函数创建一个空字符串：

std::string str; // 创建一个空字符串

也可以使用带有初始值的构造函数创建一个包含特定内容的字符串：

std::string greeting("Hello, world!"); // 创建一个包含"Hello, world!"的字符串

## 2. 获取字符串长度

通过调用length()或size()方法，可以获取字符串的长度：

std::string str = "Hello";

int length = str.length(); // 获取字符串长度

## 3. 连接字符串

使用+运算符可以将两个字符串连接起来：

std::string str1 = "Hello";

std::string str2 = " world!";

std::string result = str1 + str2; // 连接两个字符串

## 4. 提取子串

可以使用substr()方法从一个字符串中提取出指定位置和长度的子串：

std::string str = "Hello, world!";

std::string sub = str.substr(7, 5); // 从第7个字符开始提取长度为5的子串

## 5. 查找子串

使用find()方法可以查找子串在字符串中的位置，如果找到则返回子串的起始索引，否则返回std::string::npos：

std::string str = "Hello, world!";

int index = str.find("world"); // 查找子串"world"在字符串中的位置

## 6. 替换子串

使用replace()方法可以将字符串中的指定位置和长度的子串替换为新的字符串：

std::string str = "Hello, world!";

str.replace(7, 5, "everyone"); // 将第7个字符开始的长度为5的子串替换为"everyone"

## 7. 比较字符串

可以使用==、!=、<、>等运算符来比较两个字符串的大小：

std::string str1 = "apple";

std::string str2 = "banana";

bool isEqual = (str1 == str2); // 判断两个字符串是否相等

bool isLess = (str1 < str2); // 判断str1是否小于str2

6-24设计思想：

**动态内存管理**：使用 new[] 和 delete[] 管理字符串的内存，确保在对象创建和销毁时正确分配和释放资源，避免内存泄漏。

 **运算符重载**：重载了赋值运算符、加法运算符和复合赋值运算符，使得 myString 对象可以像内置类型一样进行赋值和拼接操作。

 **深浅拷贝管理**：在构造函数和赋值运算符中，通过动态分配内存并使用 strcpy 复制字符串内容，确保对象复制时的数据完整性，避免浅拷贝导致的数据共享问题。

**字符串操作的封装**：提供了多个成员函数和运算符重载，使得用户可以方便地进行字符串长度获取、内容修改、拼接、截取等操作，封装了字符串操作的细节。

6-25设计思想：

**封装**：使用类将数据和相关操作封装在一起，隐藏了具体实现细节，提供了简洁的接口供用户调用。

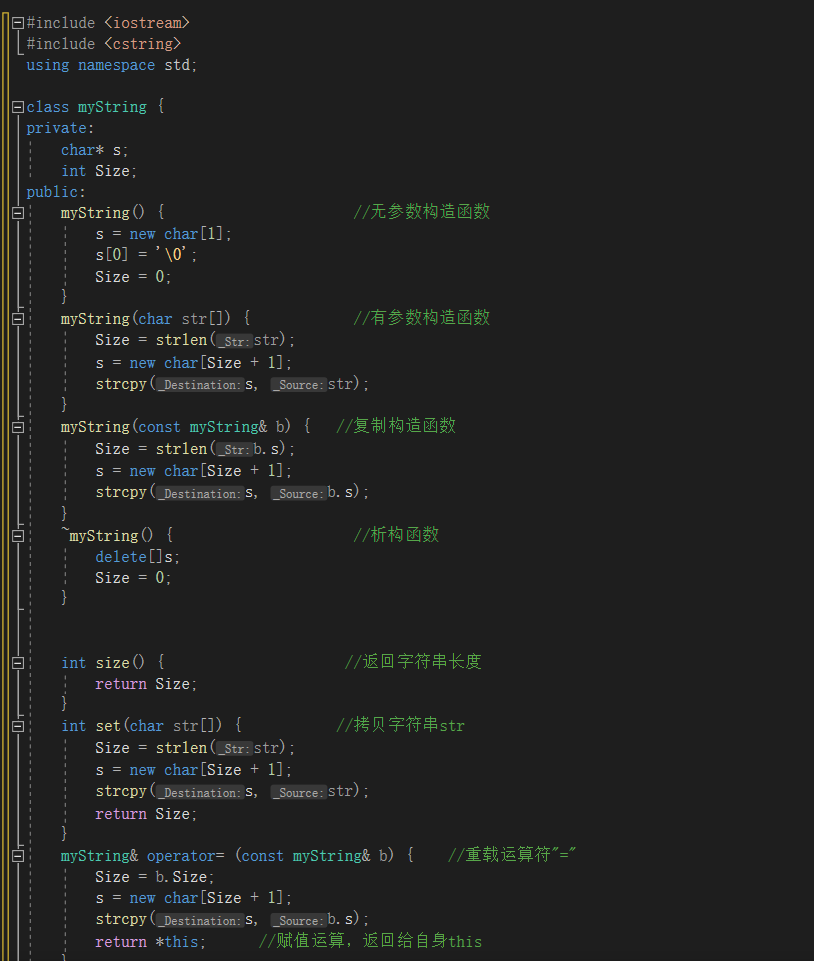
**数据结构选择**：选择了 vector<vector<int>> 作为存储矩阵数据的容器，这是一种灵活且方便管理二维数据的方式。

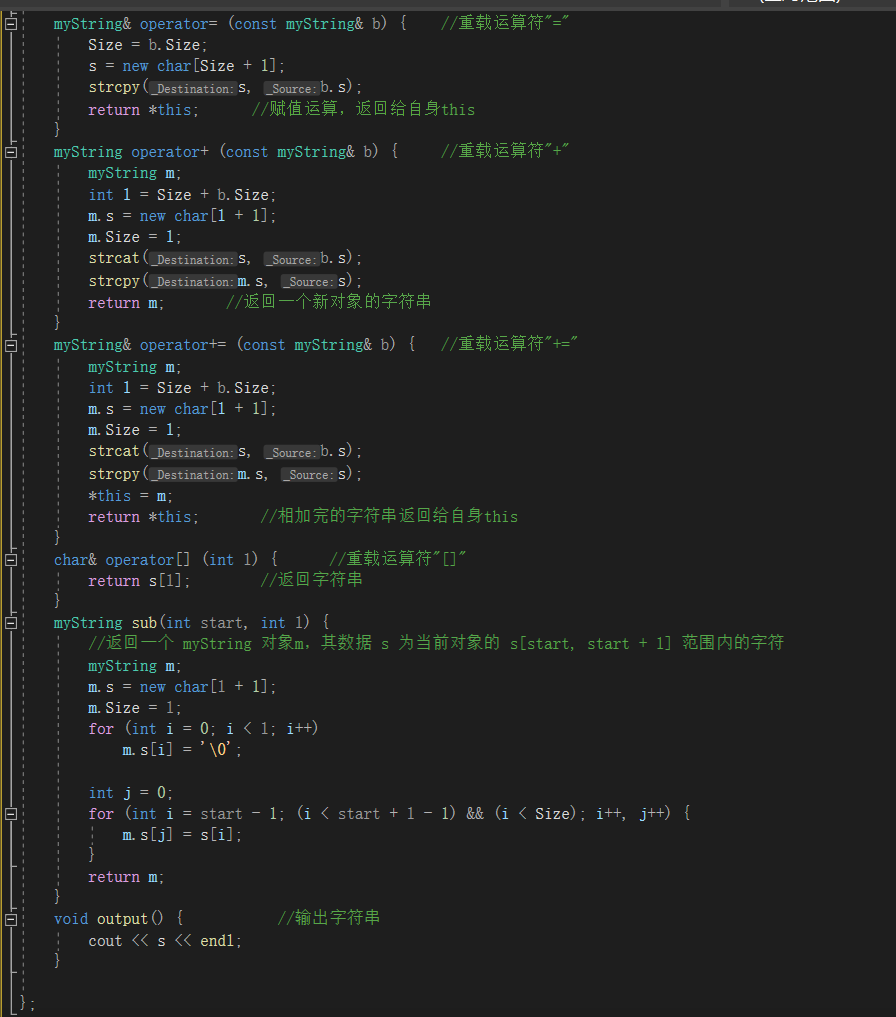
**面向对象**：通过定义类和成员函数，实现了矩阵的输入、转置和输出功能，体现了面向对象编程的思想和优势。

**用户交互**：通过在主函数中使用标准输入输出，实现了与用户的简单交互，使得程序更具交互性和实用性。

【源码】

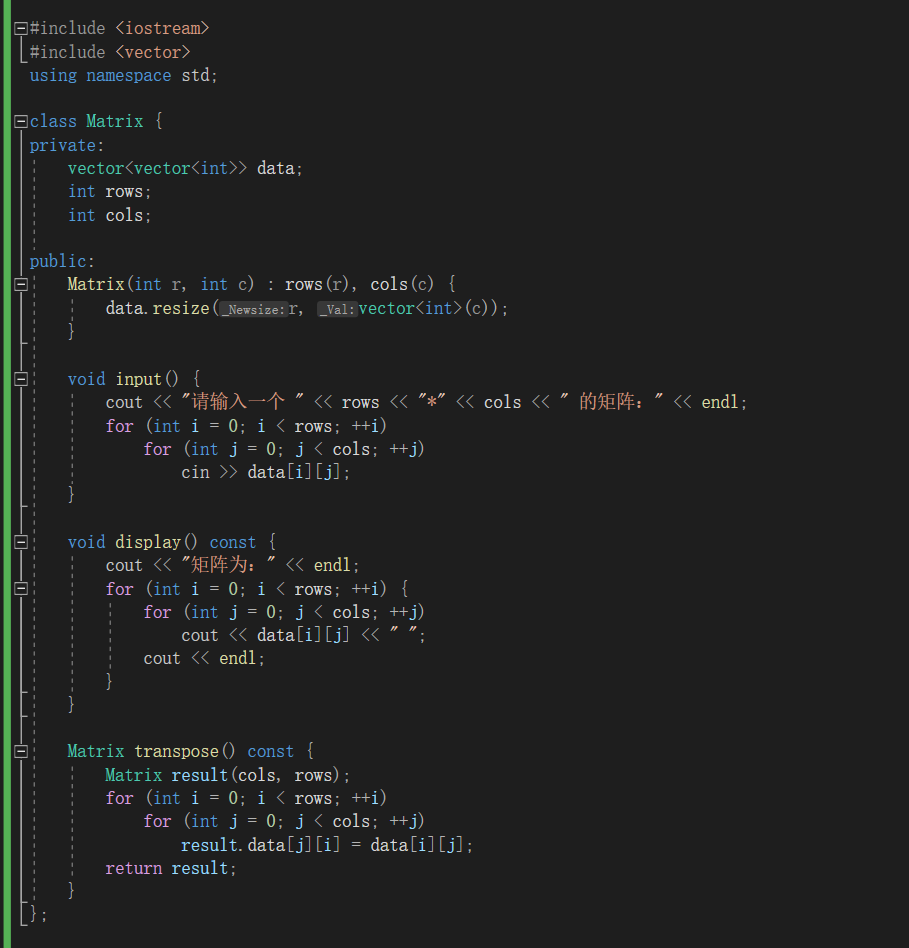
6-24：

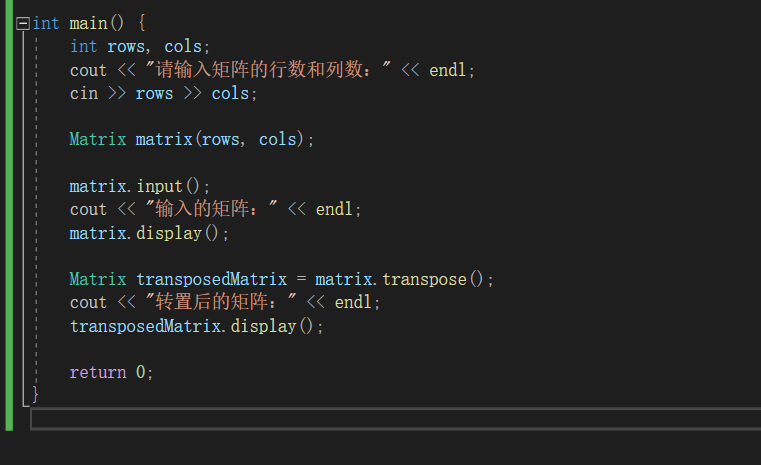






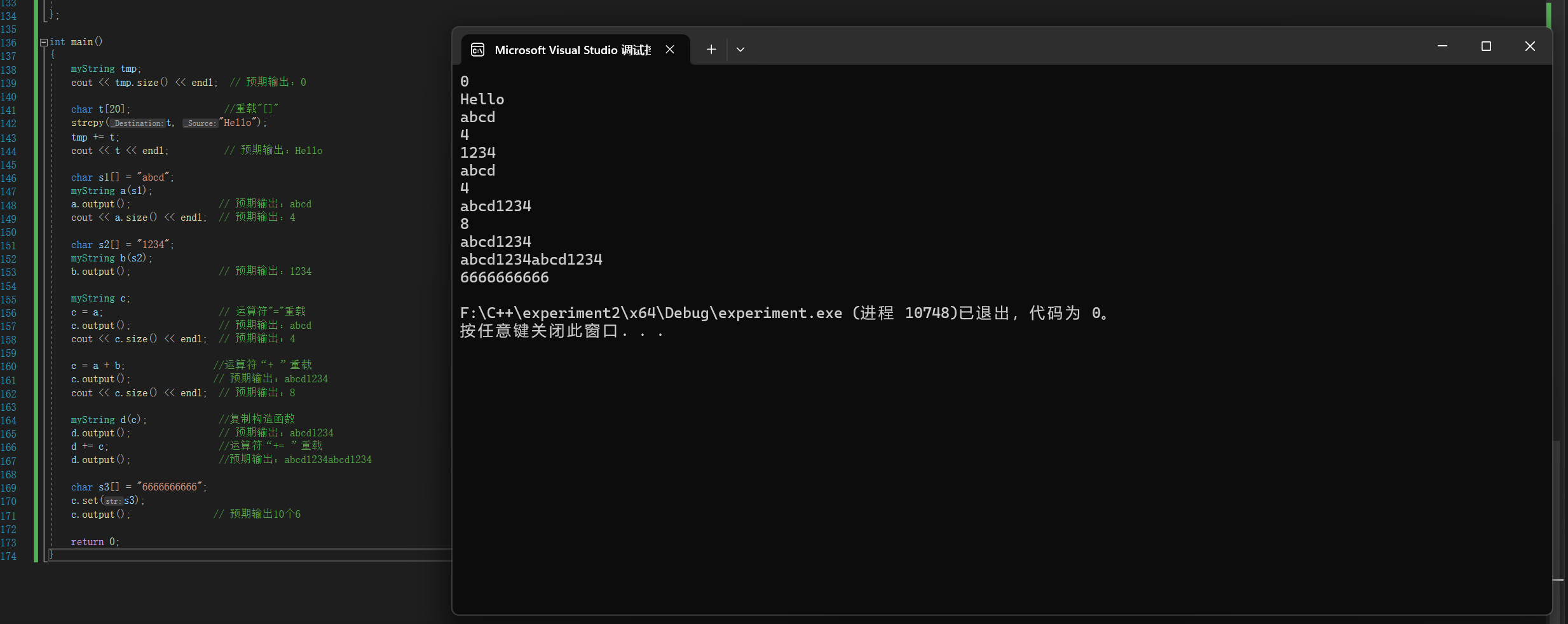
6-25：



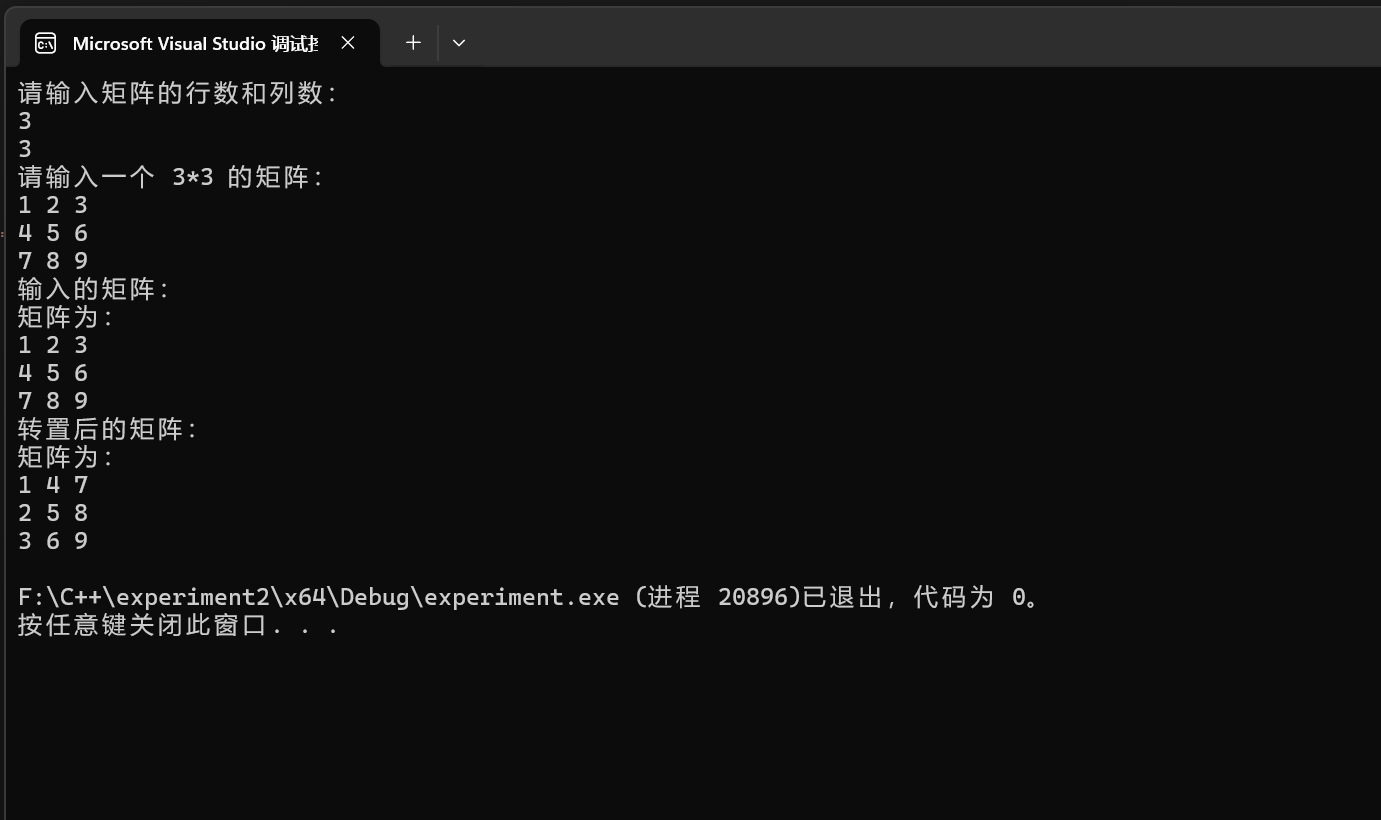


【运行结果截图】

6-24：



6-25：



【实验总结】

1. 在实验中，我首先熟悉了C++标准库中的std::string类及其相关函数。在使用string类时，需要包含头文件 #include<string>,并且引入using std::string; using std::wstring;或sing namespace std;
2. 通过编写程序，我练习了字符串的基本操作，包括字符串的创建、连接、分割、大小写转换等。随后，我设计了一个矩阵类，实现了矩阵的输入、输出、以及转置操作。在实现矩阵转置的过程中，我特别注意了矩阵的内存管理和数据结构的选择。
3. 在实现矩阵转置的过程中，我遇到了内存分配和数据复制的问题。为了解决这些问题，我采用了动态内存分配，利用 vector 动态数组来管理矩阵数据。这样可以自动处理内存分配和释放，避免内存泄漏。同时，在转置函数中避免了不必要的数据复制，通过直接访问和修改数据，提高了程序的效率。

本次实验让我对C++中的字符串类和矩阵操作有了更深入的认识。我学会了如何使用字符串类来处理文本数据，并且通过实现矩阵转置，加深了对数据结构和内存管理的理解。