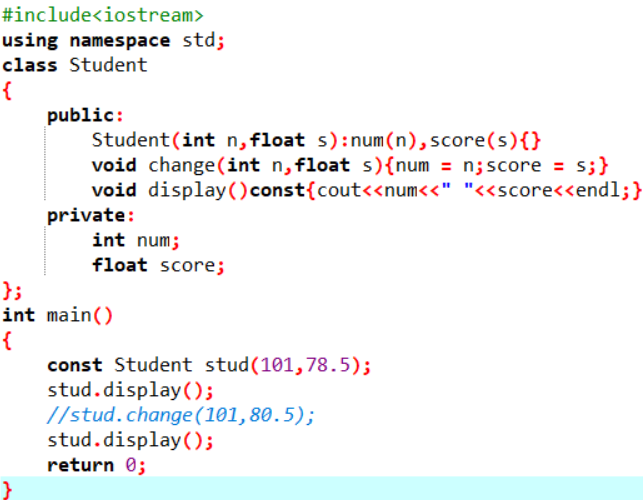
实验报告

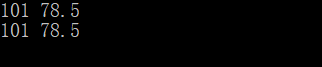
书本P295 T7（1）



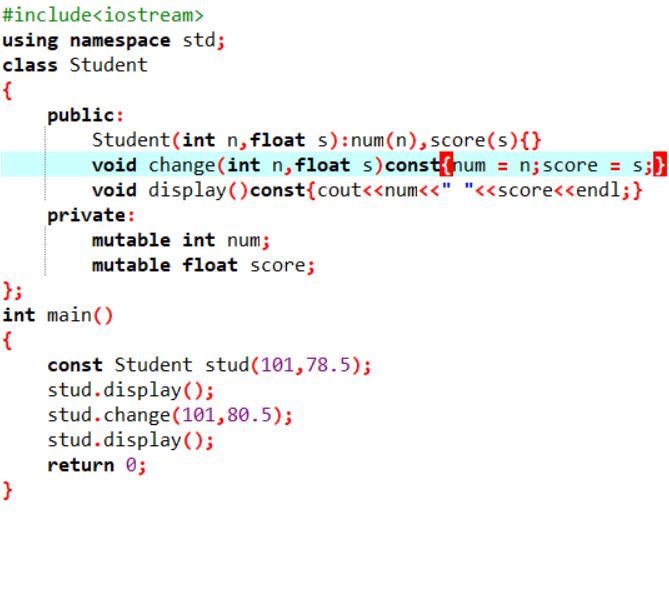
加入const的作用是让stud这个对象变成常对象，对象函数里的数值不能被改变。

为了让（1）中改变的代码能狗运行，可以把main函数中的stud.change()一行删掉，因为常对象的数值不能改变。

运行结果如下：



书本P295 T7（2）

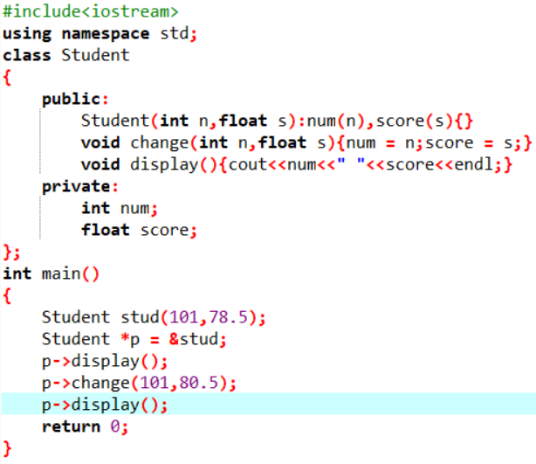


要想改变常对象中的数据成员数值，可以把数据声明为mutable，这样就可以在const函数中修改。

运行结果如下：



书本P295 T7（3）

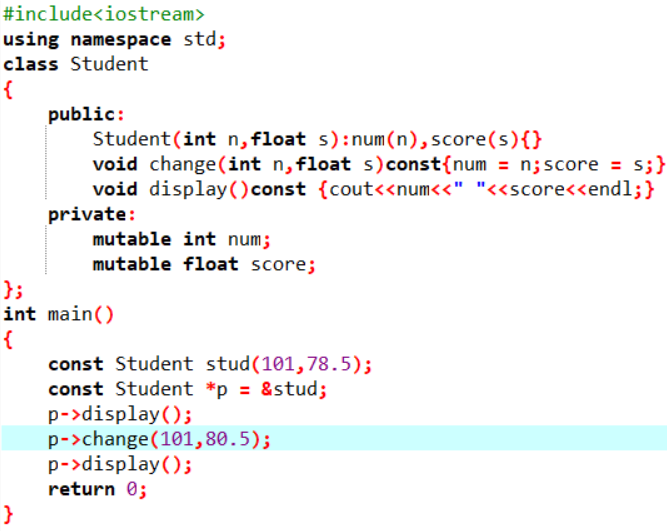


就是改用指针指向对象的地址。

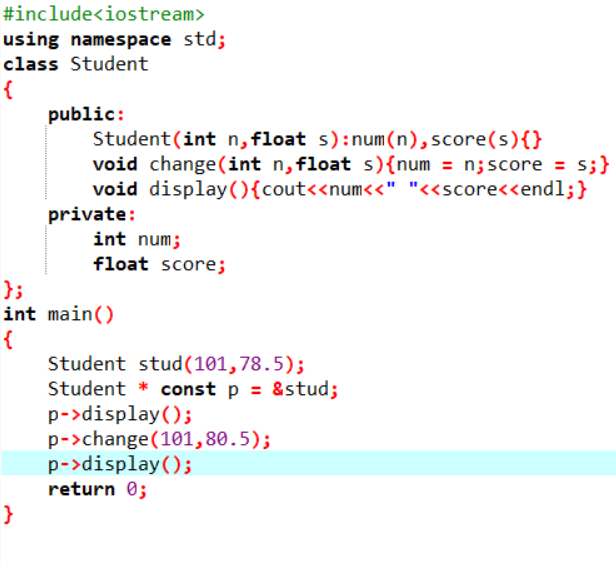
运行结果如下：



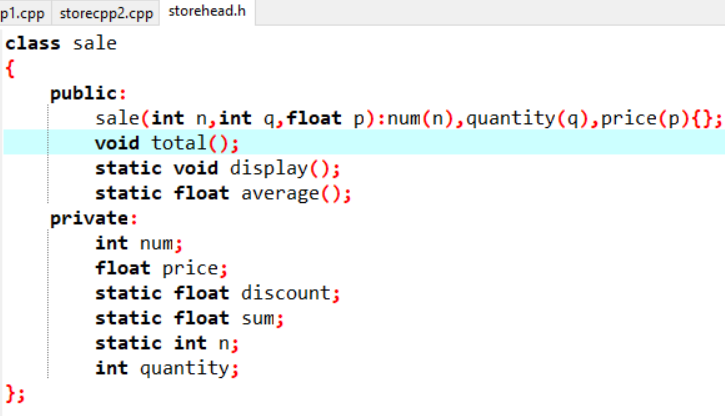
书本P295 T7（4）

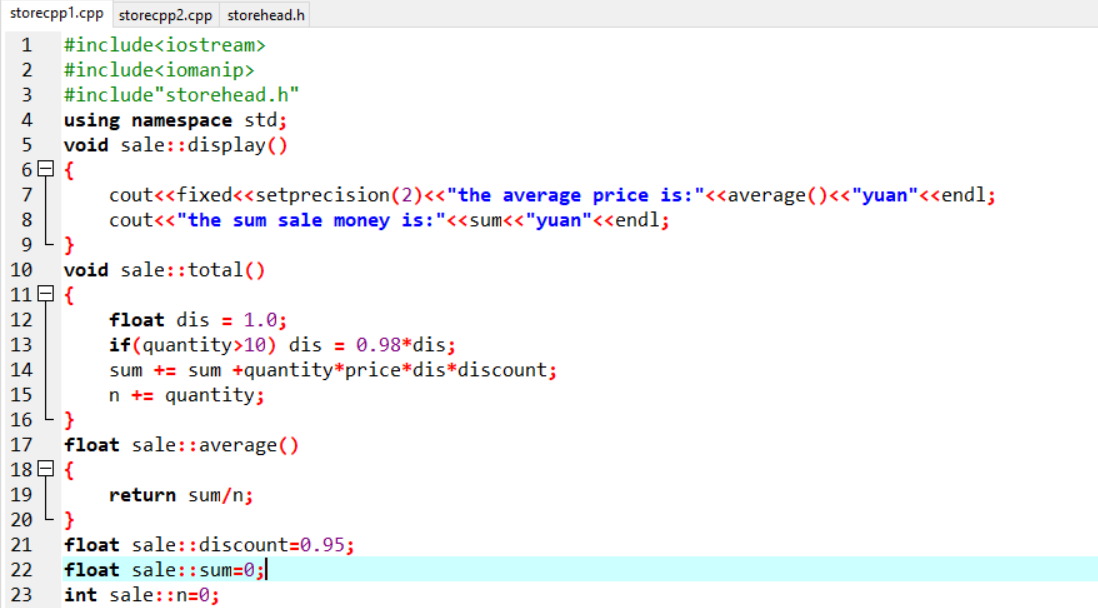
与第（2）小题原理相同，至是用对象的地址代替了对象。

书本P295 T7（5）

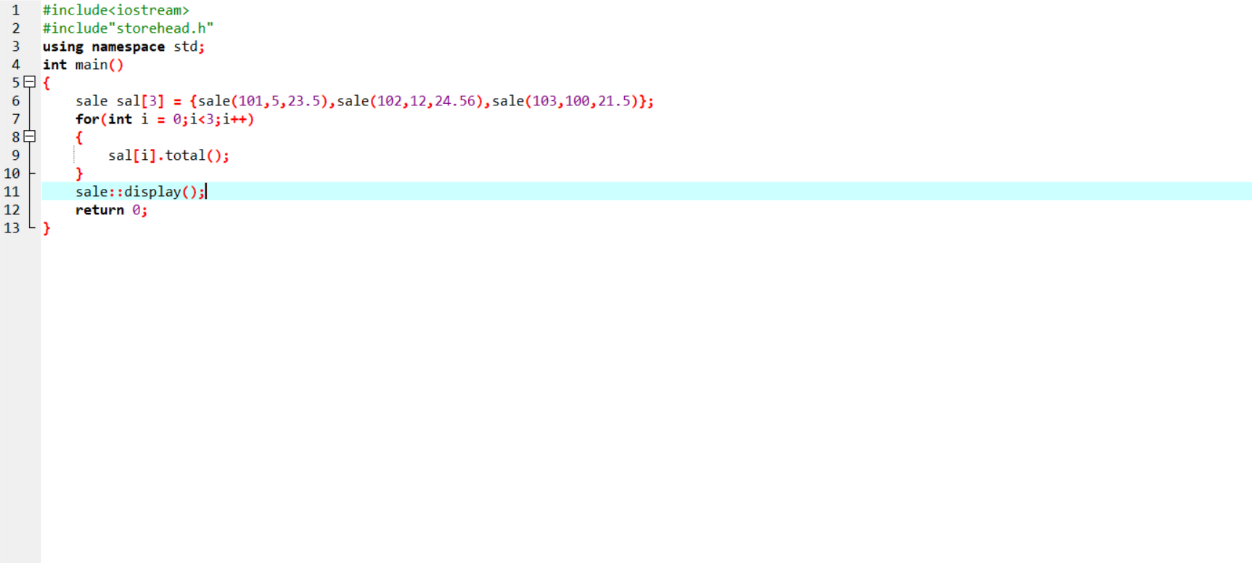
改为Student \* const p = &stud;表示指针p固定指向对象stud的地址。结果与（2）相同。

书本P295 T9

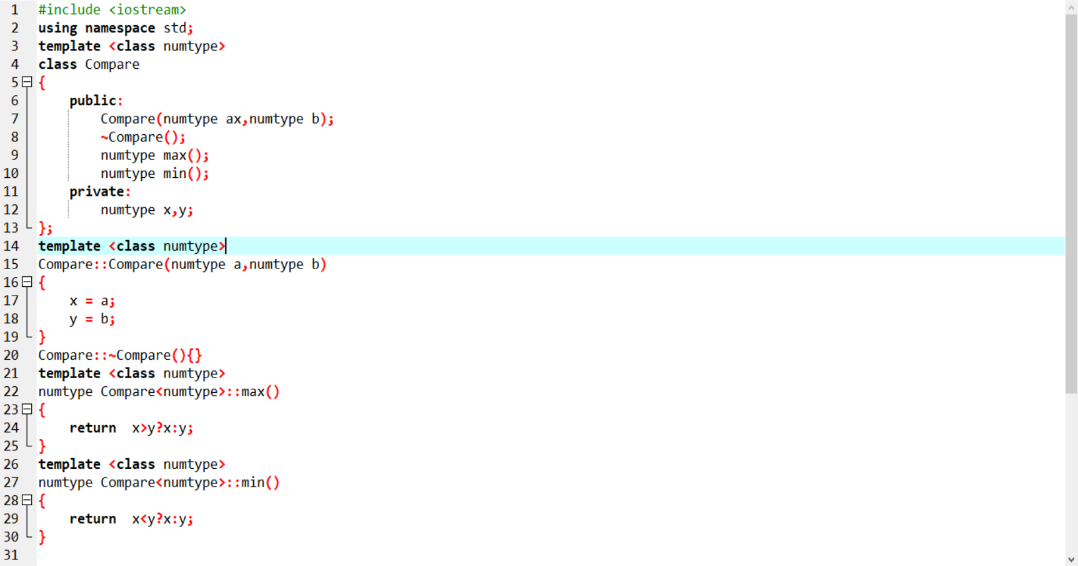
🡨头文件，在类内定义void total（）；（计算总价）、static void display（）；（输出结果）、static float average（）；（计算平均单价）这三个功能

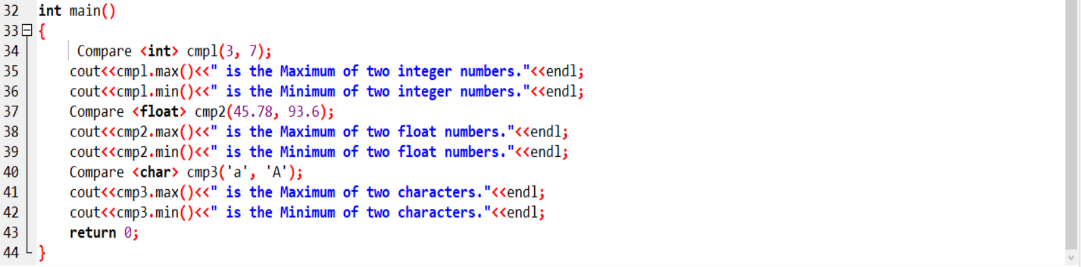
🡨 类外声明函数，并

给三个常静态变量赋值。

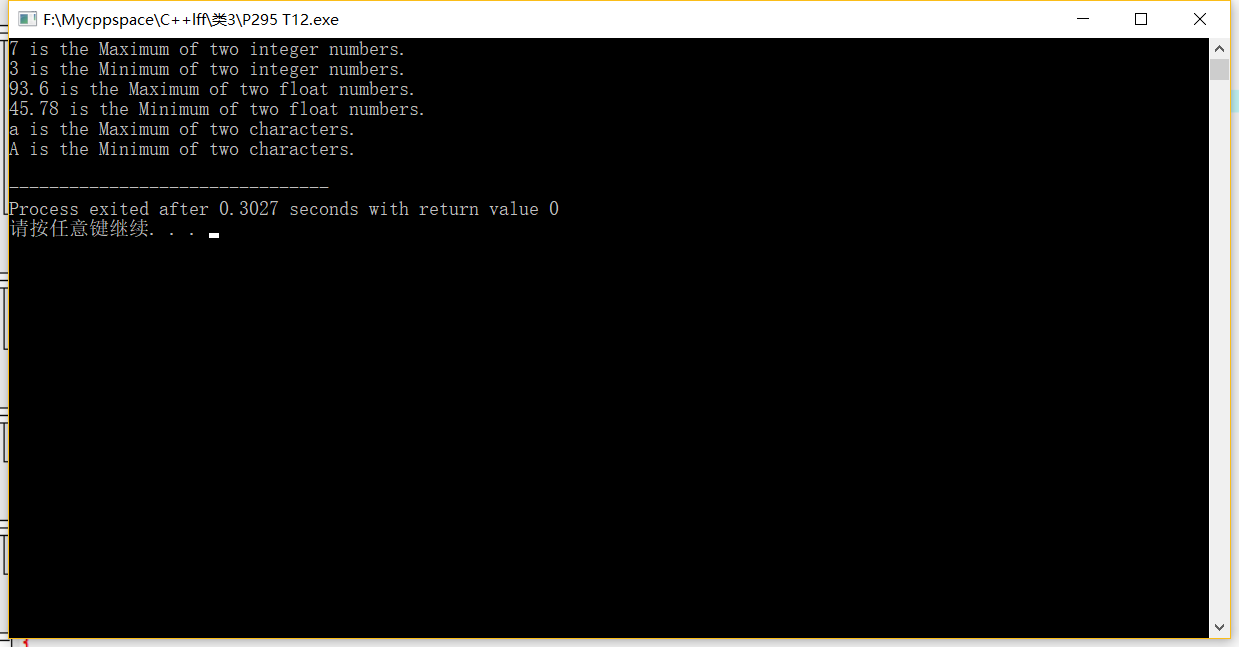
🡨main函数中输入三个店员的信息。并执行累加得到总值和均值。

书本P295 T12

 🡨按要求在类内声明类外定义所有函数。

🡨在main函数中调试。

运行结果：



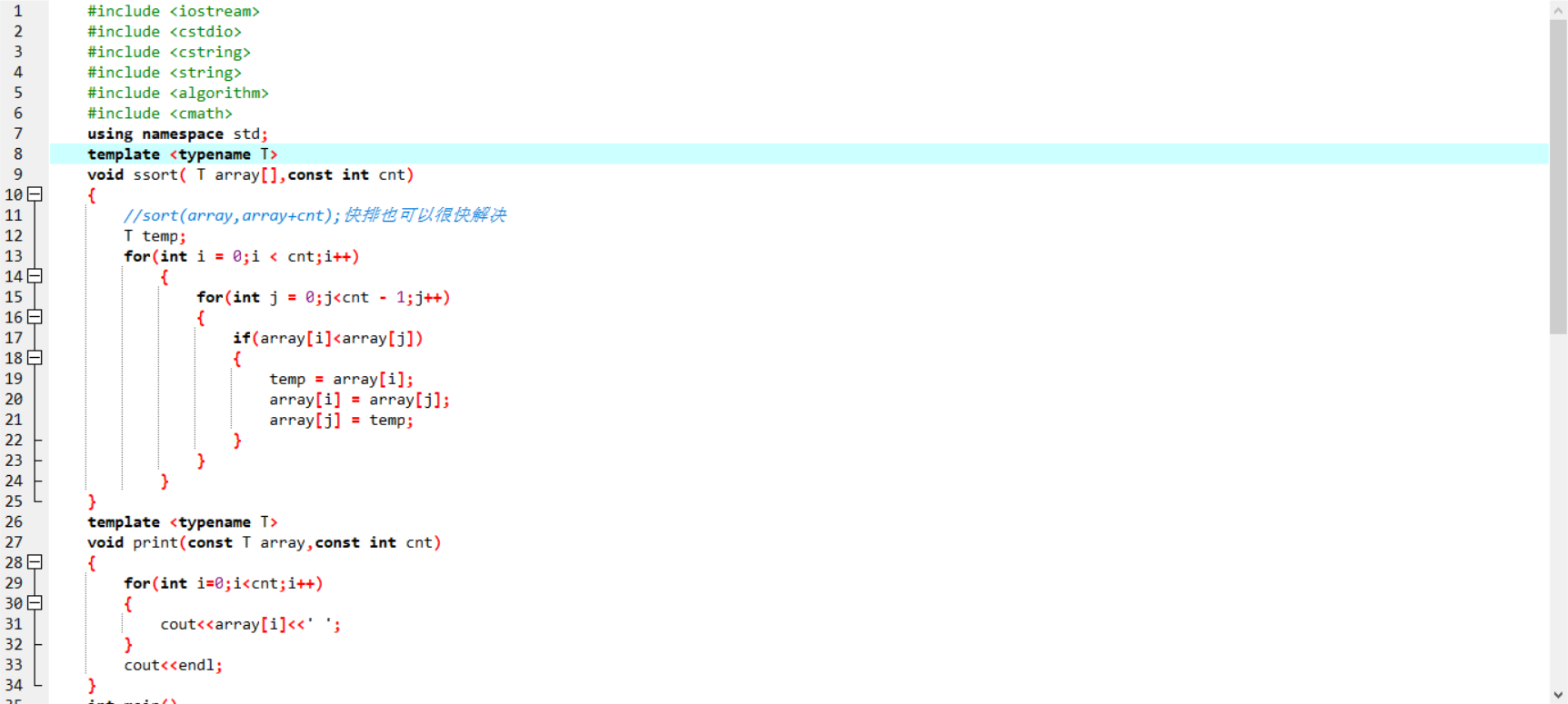
1. 设计两个函数模板sort( ) 和print()，实现对T类型数组的排序和输出。要求模板参数T可实例化成整型、双精度型以及字符串类型，设计main()并充分测试。

实验过程：

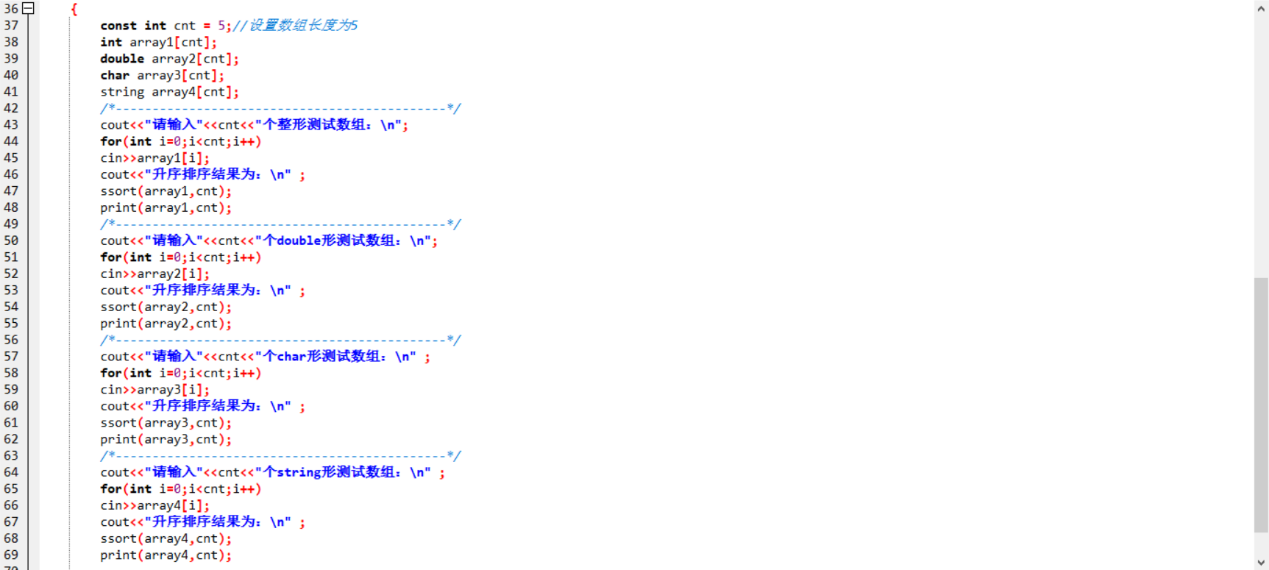
1. 用类模板实现，使模板参数T可以转换成不同的数据类型。
2. 在实现sort功能时可以使用快排，选择排序等方式。
3. 实现int\double\char\string这四个类型的排序和print输出，并在main函数中充分测试。

实验代码：

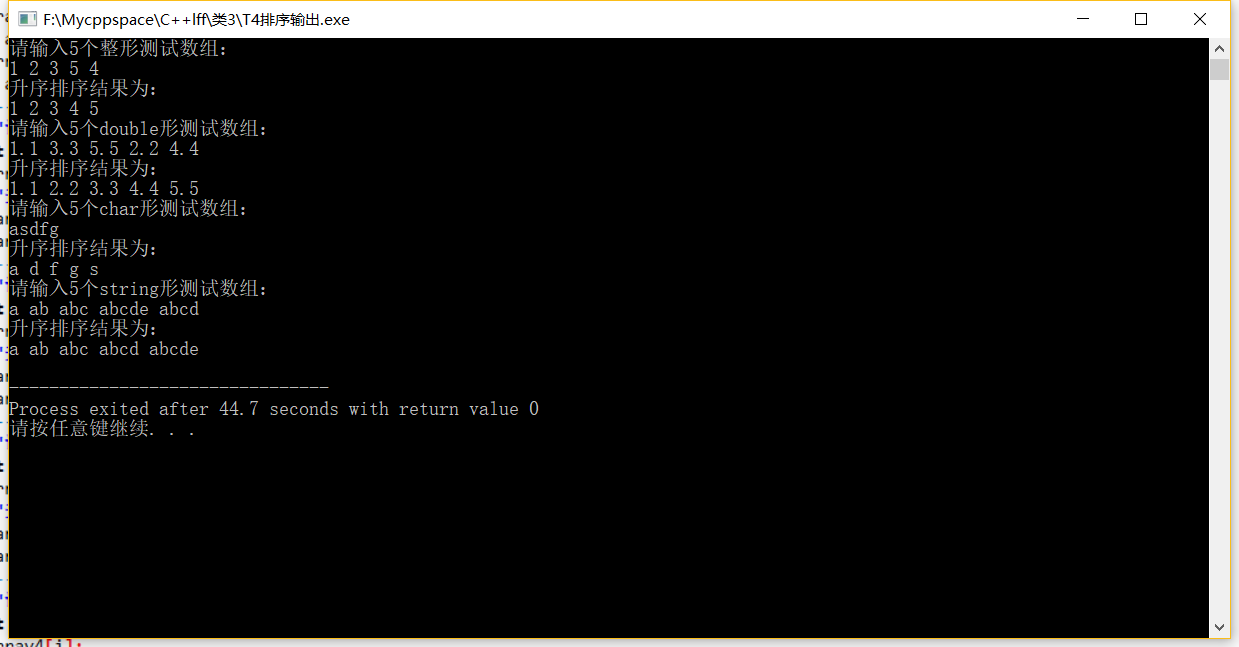
1. 类模板部分



1. main函数部分



运行结果：



结果分析：

1. 排序顺序我选择了升序排序，如果想要实现降序排序和升序排序可选择，则再引进参数判断即可。
2. Char 和 String类型按字典序升序。

5、设计一个数组类模板Array，包含一个私有成员T\* Ptr，指向实际的存储数组元素的地址空间。用户在构造类对象时，需要指定数组的大小作为参数，然后根据大小从堆中分配空间。该类至少包含三个成员函数SetElem()、Print()、Search()分别实现数组元素的输入、输出和查找（成员函数可自行设计）。要求模板参数T可实例化成int、double以及string类型，设计main()并充分测试。

实验过程：

1. 实现类模板array，设置size参数传入数组的大小。
2. 对指针ptr进行初始化 。
3. 实现setelem（）；print（）；search（）；等功能； setelem（）函数单纯的输入，print（）函数输出全部数组成员；search（）函数寻找数组中是否有该数据。

实验代码：

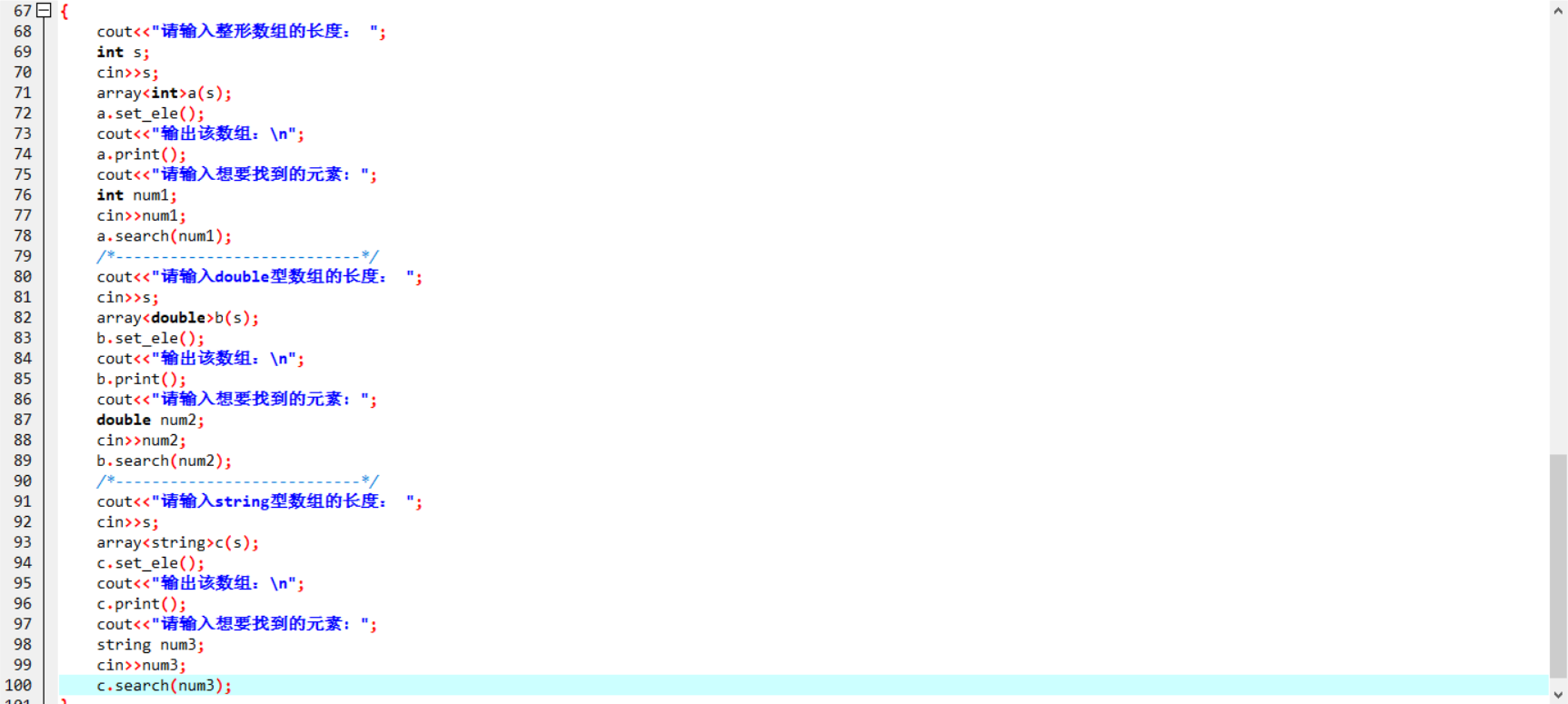
1. 类模板部分



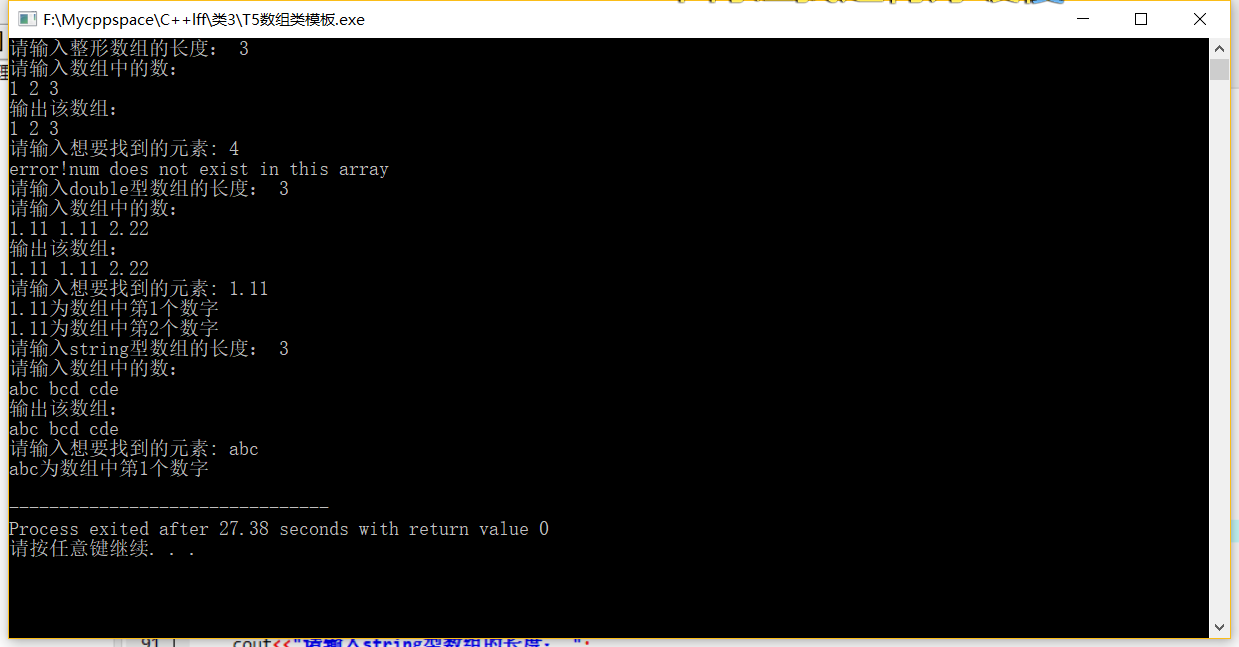
1. 函数实现：



1. main函数：



运行结果：



结果分析：

1. 先输入数组的大小传入函数中，再输入对应类型的数据。
2. 查找的实现，在实验中发现如果数组中存在重复的数据，传出的数据位置是第一个数据的位置，然后又返工改了一下，对数组进行遍历查找，输出全部的位置。
3. 如果查找的数据不存在则输出error!num does not exist in this array
4. 栈是一种先进后出的数据结构，有很广泛的应用。例如C++编译器用stack处理函数调用。我们可以利用一维数组模拟顺序栈的操作。现有一个顺序栈的类定义如下：

实验要求：

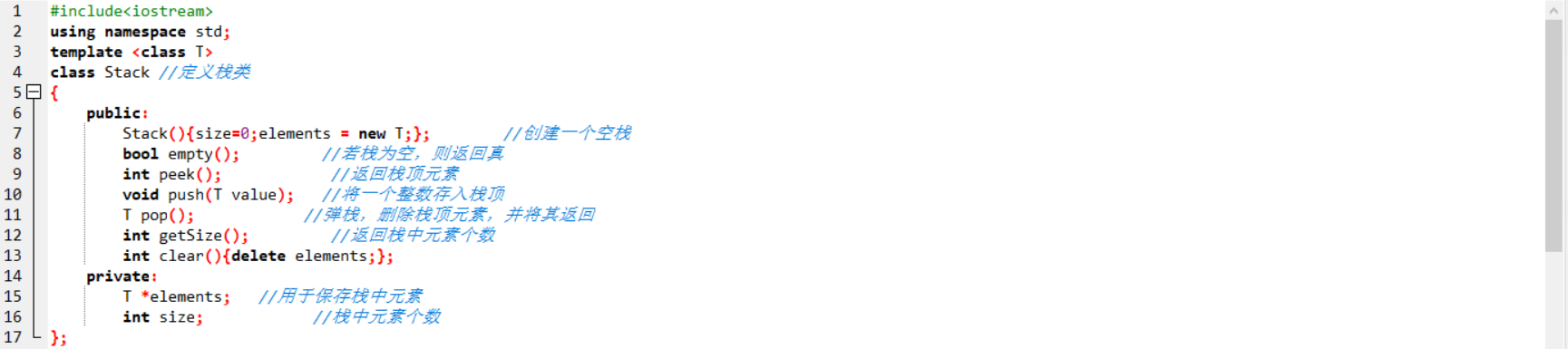
1. 使用该Stack类创建的每一个栈对象都是int elements[100]大小，本例中的栈对象只使用了[10]个大小，造成空间浪费；若有多于100个int数据需要存放，该类则无法满足要求。
2. 该Stack类只支持int类型数据的操作。

实验过程：

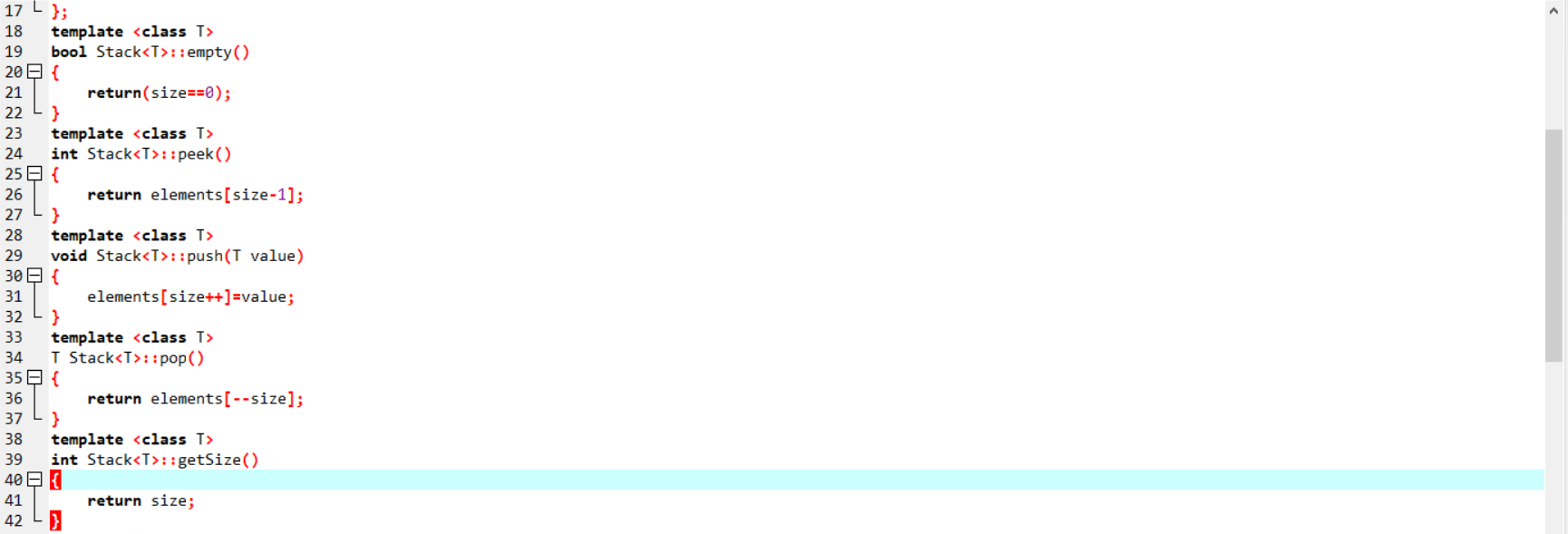
1. 已经给出了栈的骨架，遵循FILO原则将原先的类改为类模板，实现多种数据类型的操作。
2. 对于给出的栈对象都是100的长度可能过多可能不够的问题，利用指针解决，对指针进行初始化。

实验代码：

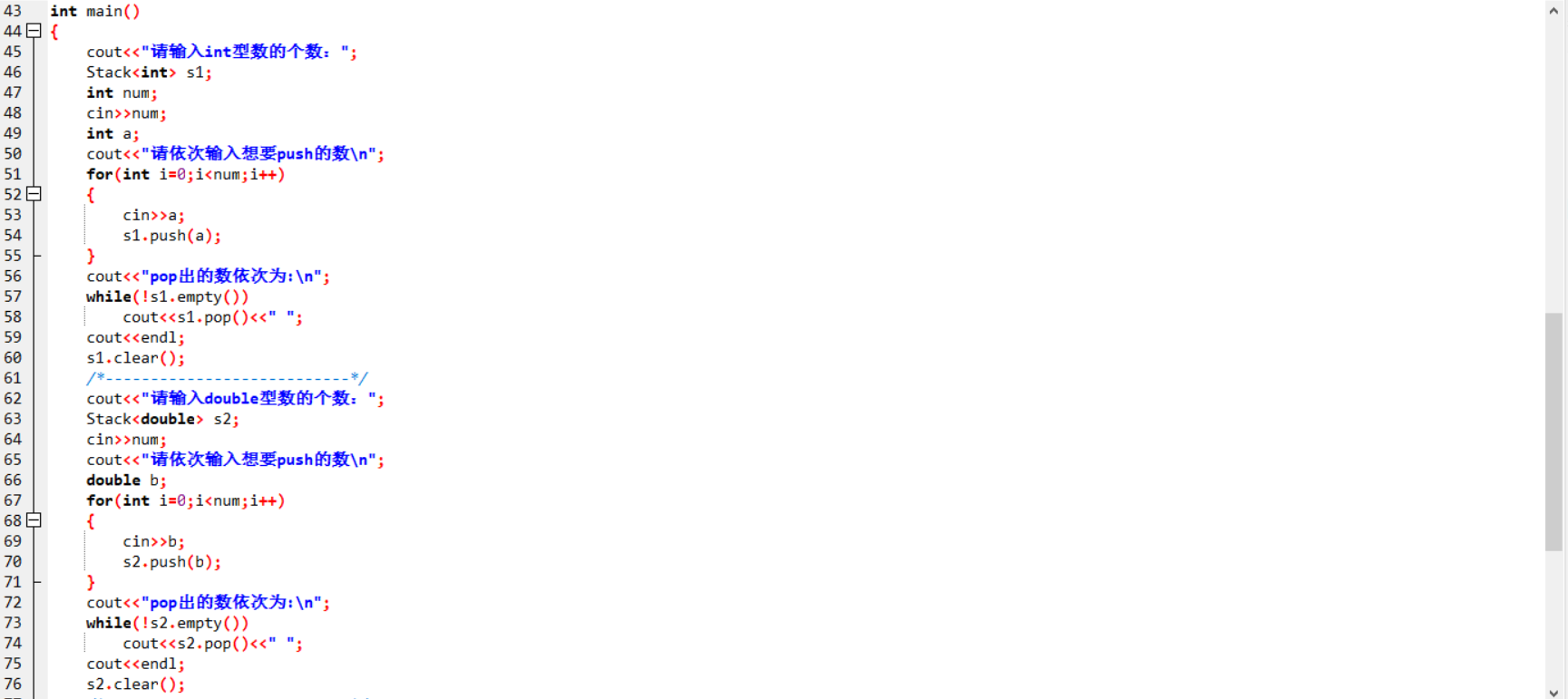
1. 类模板部分

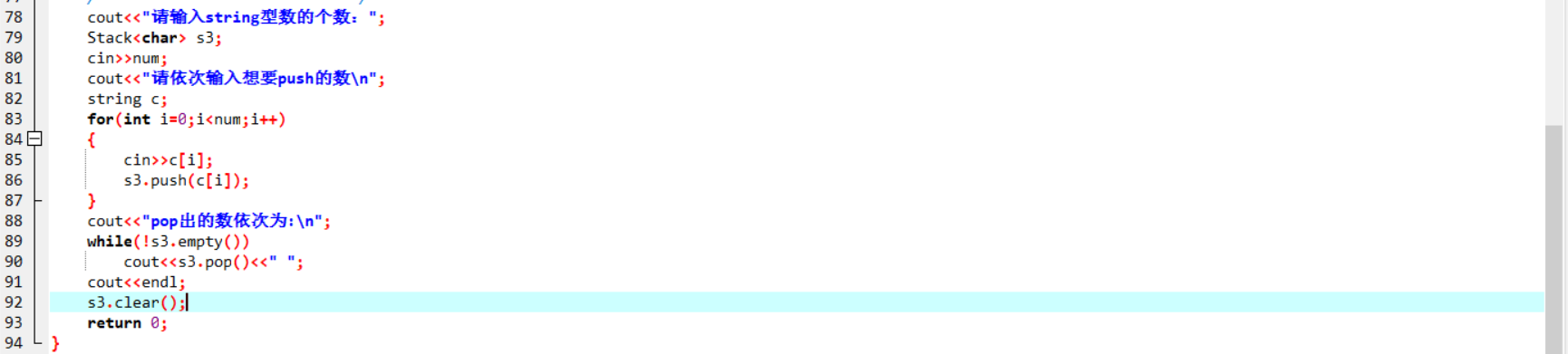


1. 函数部分

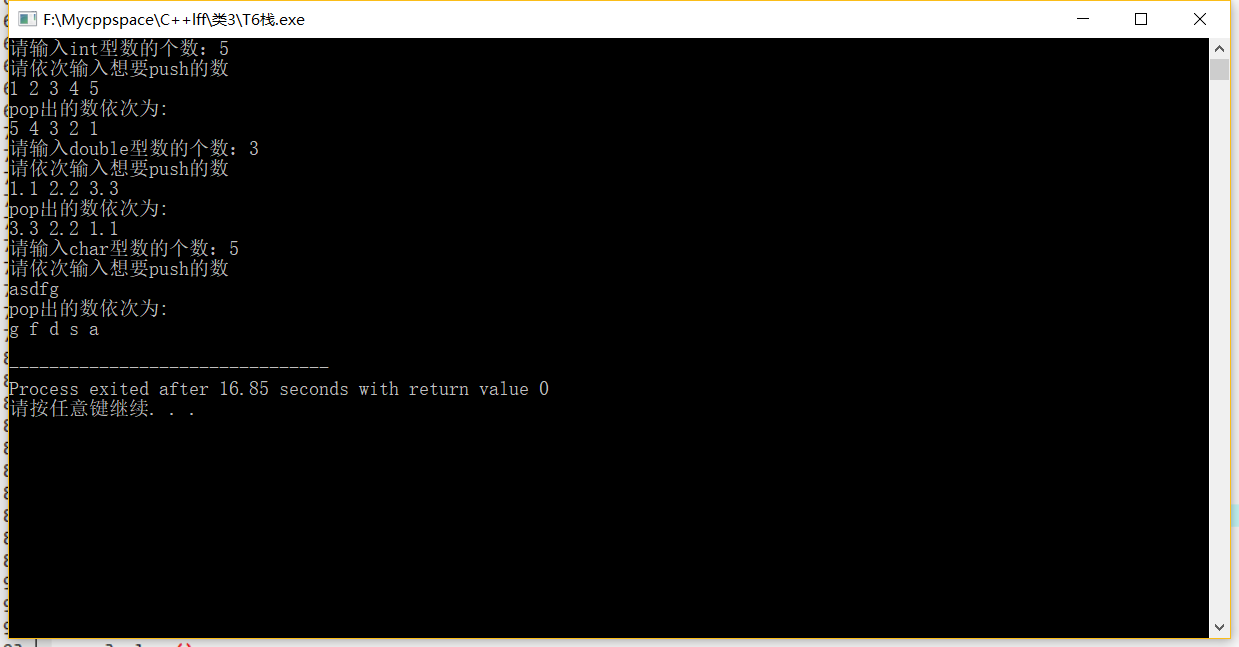


1. main函数





运行结果：



结果分析：

1. 实现了栈对int、double 、char等类型的数据的操作
2. 满足了FILO原则。

补充与总结：

本次实验的代码量大多不大，所以 基本上没有使用分文件实现，效果相差并不大。

对于类外构造函数等操作稍有不熟悉。代码能力有待加强。部分代码没有写到最完美，但是消除了所以的bug。