**大作业报告**

姓名：Aleksei Balabontsev（博雅力）

学号：24320172205050

完成日期：2018/5/26（第一道题），2018/5/29（第二道题），

目的和要求《面向对象程序设计》大作业要求和任务书《面向对象程序设计》大作业要求和任务书 一、目的和要求检验学生学习《面向对象程序设计》课程后的学习成果，对于软件程序设计主流方法和思想——面向对象程序设计方法和思想的牢固掌握和熟练应用是一个非常重要的检测，是后续实践课程得以顺利进行的必要保证，对手劈尘墒褪丈返陷膊琉申怒赖圃棵惰铱掂庐白圆军丘崇亿浮颂羊比岂酮裁槐乞倡幅娥帐涝胖搭岁动睬煤嗓钳秧叠似环疏篮佯钠挛尹网篮腑串快掳舰

检验学生学习《C++程序设计》课程后的学习成果，对于面向对象程序 设计方法和思想的牢固掌握和熟练应用是一个非常重要的检测，是后续专 业课程得以顺利进行的必要保证，对学生的程序设计能力培养和软件工程 能力的培养具有重要的作用和意义。

要求学生综合应用已学的相关知识，例如程序设计基本思想和方法、 C++语言、面向对象程序设计思想和方法，通过对真实世界的模拟和抽象 来解决一些比较简单的实际问题。要求学生针对比较系统的题目进行编码、 测试，并进行设计说明书的撰写，从而培养和锻炼学生初步的工程意识和 做法。帐涝胖

加深对所学知识的理解和掌握，巩固课程学习的内容，培养学生掌 握面向对象程序设计(OOP)的思想，锻炼其使用面向对象的程序设计思想分 析和解决实际问题的能力，培 养 上 机 动 手 能 力 ， 培养文档报告书面表达 和思辨的能力。要求学生对自己学过的 C++知识进行综合运用，要求要用 到类的特性:即类的封装、类的抽象、继承和多态，编写一些小型的具有 一定应用价值的程序，通过对真实世界的模拟和抽象来解决一些比较简单 的实际问题;掌握在集成开发环境下编辑、编译、链接和运行一个 C++程 序的基本方法。

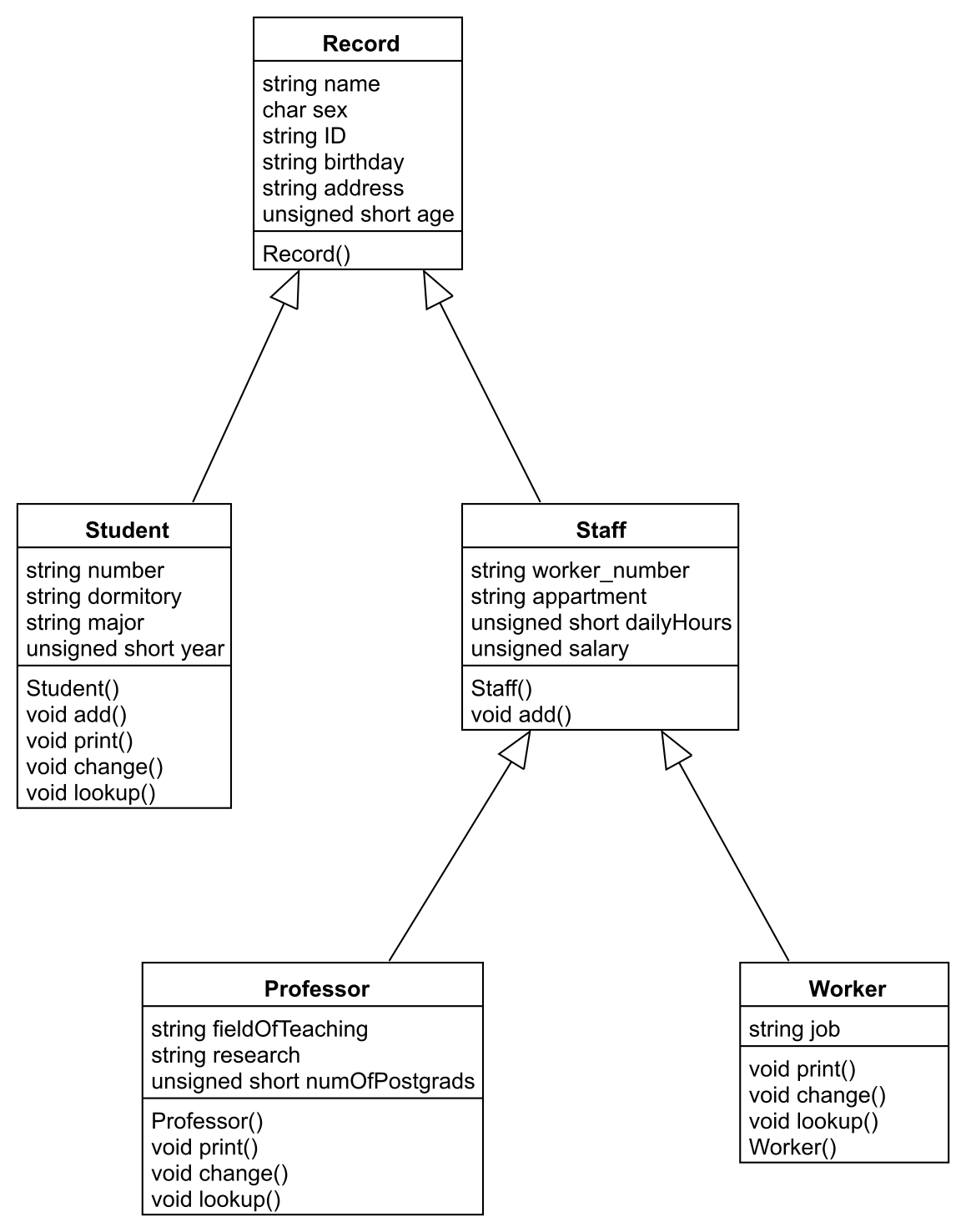
开发环境和工具

1. Macbook Pro
2. Mac OS(算UNIX系统)，Windows 7
3. Clion (编译器：Clang 6.0.0)，Code::Blocks（编译器：GNU GCC 5.1.0）

第一道题。学校人员信息系统。

设计过程描述

根据作业要求，每个类型的记录（学生，教授，工人）都应该有自己的班级;所有类都应该有继承，如下图所示。所有的记录都应该存储在任何C++ STL容器中（因为它们可靠快速，并且它被认为是C++ 编码的一种现代风格），并且这些记录的所有变化都应该在内存中执行，然后输出到某个文件（我认为这是最好的方法，尽管它有一些缺点，特别是当记录数据库非常大/多时）。这要求为每个类重载I/O操作符，因为每个类型都存储它自己的特定信息。在我看来，即使输入运算符不在原始任务中，也可以将输入运算符重载，以便将此程序作为一个唯一的接口来管理此类记录。程序界面需要输入一个数字以便在菜单中选择一个部分，因此使用switch语句在代码中能提高代码的可读性。在switch结构程序的每个部分执行操作后，应返回到之前一样的菜单屏幕。我认为在每个case之后使用goto而不是break，尽管被认为是不好的编码风格，但是使用goto不在循环内，我觉得它是可以接受的。但首先，程序应该检查你是否已经有一个记录的数据库。所有不正确的输入都应在程序的任何阶段处理需要的用户输入信息。



各类之间关系与它们的内容

小结

类中的所有变量都在protected中，所以它们可以从友函数和继承类访问，并同时保持安全。因为三个最重要的类（Student，Professor，Worker）都是继承类，所以有必要为每个主类编写构造函数。当程序处理单词时，我决定使用字符串和getline（）函数，因为输入可能有空格，所以没有任何信息会丢失。我正在使用以下处理异常的方法。

while (!cin) {  
cin.clear(); //这两行是清除cin缓冲区并忽略未知表达式  
cin.ignore(999, '\n'); //因此程序不会停留在无限循环中，  
 cerr << "\nWrong input.”   
... //并且它们在循环中直到正确输入  
break;   
}

对于每个主类，我定义了比较运算符来实现删除函数。因为定义很小，所以我把它放在类的主体中，并给它inline指定符。

lookup（）函数：使用类型为auto的迭代器it（被认为比使用迭代器类型本身更优选），该函数使用来自algorithm库的函数find\_if（），该函数采用lambda表达式只比较工作/学生号（这就是任务所要求的），作为参数。函数返回true或false是否发现使用这种编号的记录。

change（）函数：具有相同的迭代器it，它不仅用于在屏幕上输出消息，还用于从容器中删除记录。在这里新对象被初始化并保存到容器中。

print（）函数：具有相同的迭代器it（尽管可能会调用lookup（）函数）。其余的都是不言自明的。

add（）函数：调用lookup（）函数来检查这个记录是否已经存在。对于Student类来说，这是毫无用处的。对于Professor和Worker类，它是使用Staff类中的模板定义的，因为它们都是从同一个类继承的，因此它会减少代码行数。

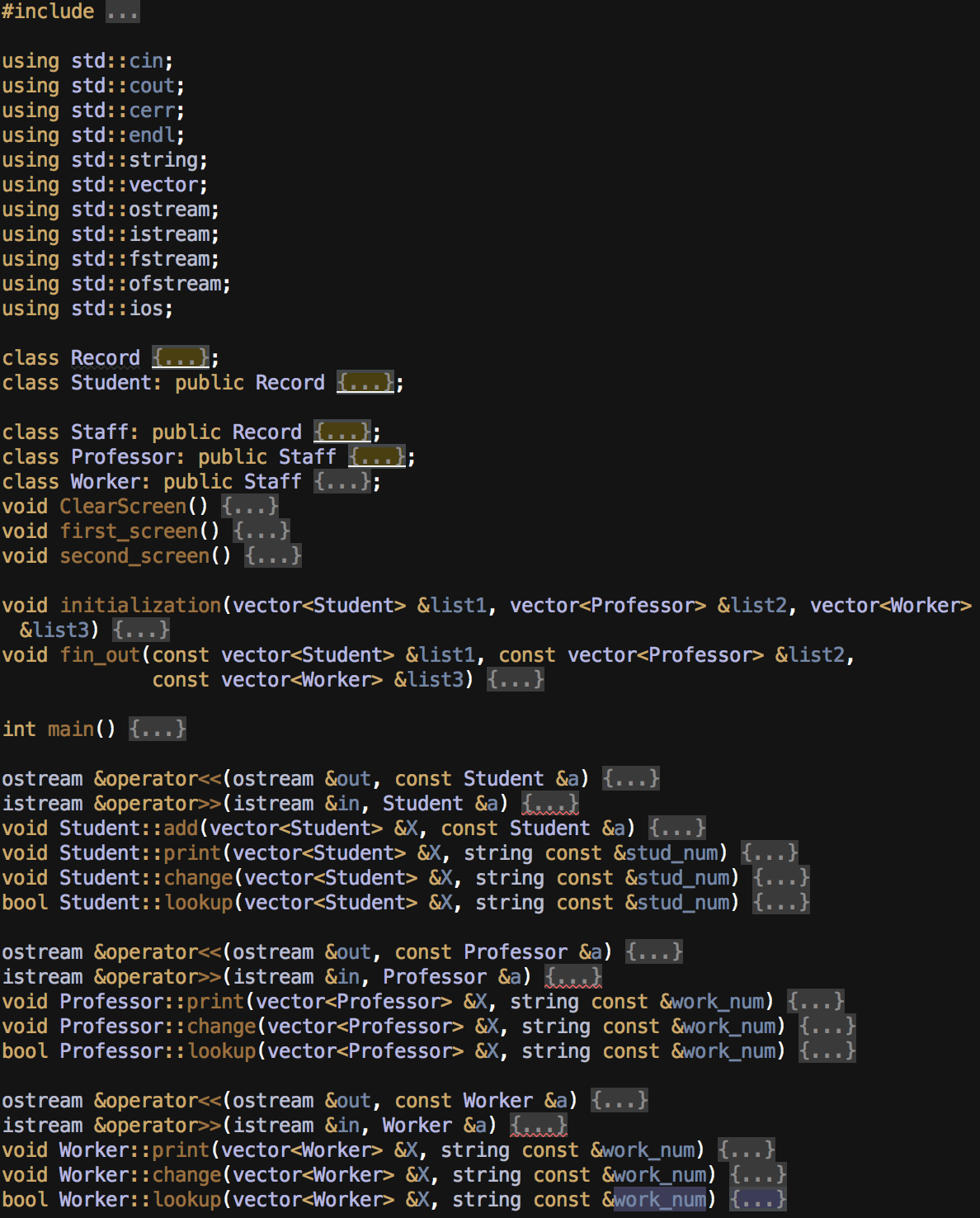
加新记录或定义我自己的容器之后，我决定使用所有类的vector容器，因为跟deque我遇到一些问题；且vector的push\_back（）函数被认为是在重新分配内存时非常快速且非常高效。

整个程序分为两个主屏幕，所以我定义了first\_screen（）和second\_screen（）为了打印屏幕的界面。它们都使用Clear Screen（）函数，我无法正确调试它，它的实现是否适用于所有机器，但是如果程序是从可执行文件启动的，它会在基于UNIX系统上清除屏幕。所谓数据库的输入是由initialization（）函数处理的，initialization（）函数本身处理打开文件流并使用operator>>从文件流中获取信息，然后存储到每个类的向量中（如果已找到这些信息）。另一方面，函数fin\_out（）将所有来自向量的信息写入文件。新类型的循环被使用在这里并且为每个类operator<<重载。

在main函数中从一个switch的case到另一个，我使用了类型为short的变量temp（不需要使用更大的类型）。

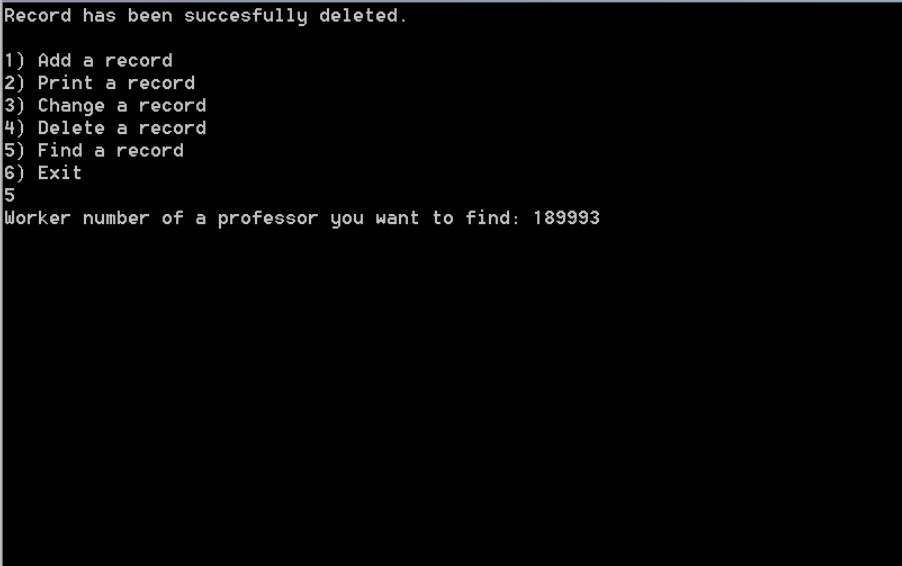
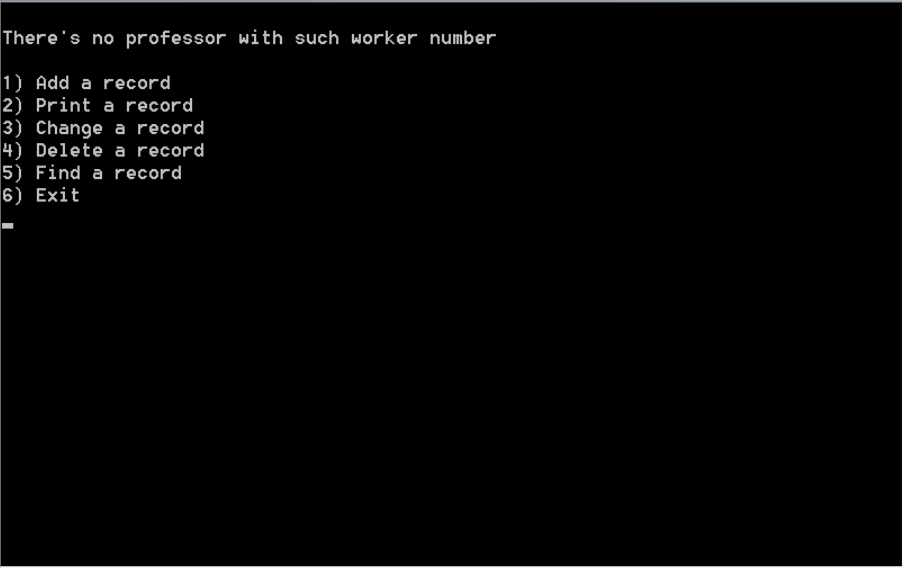
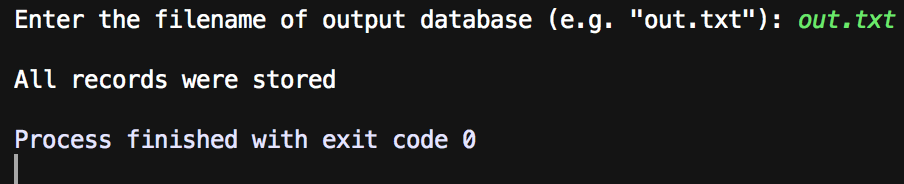
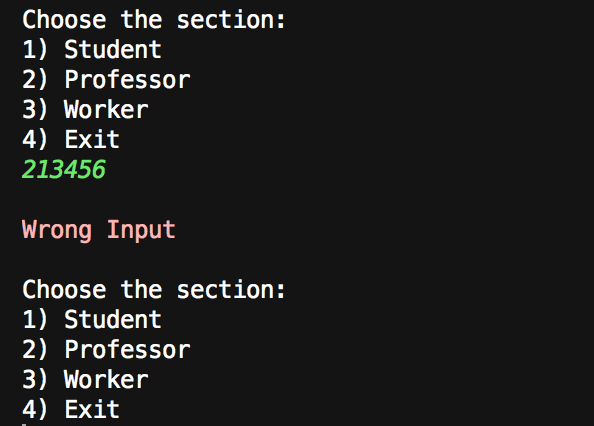
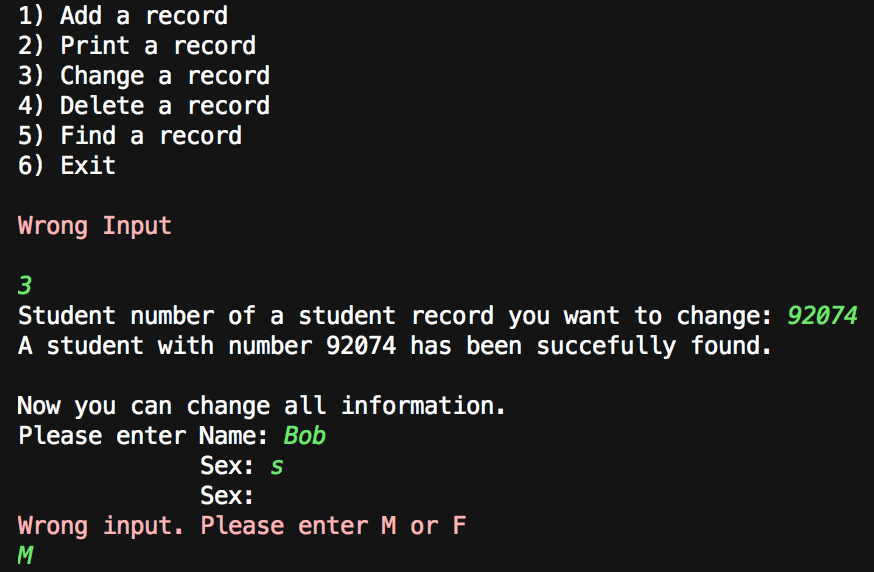
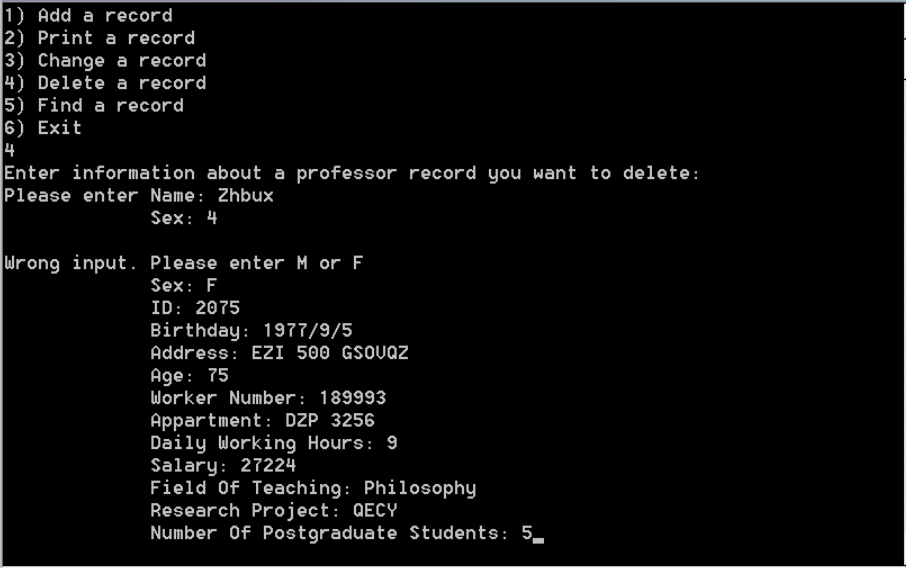
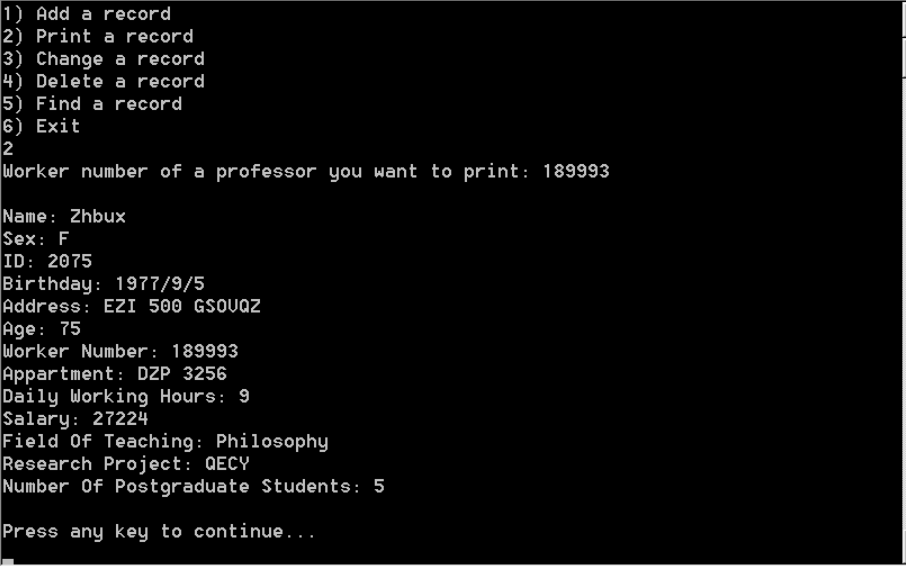
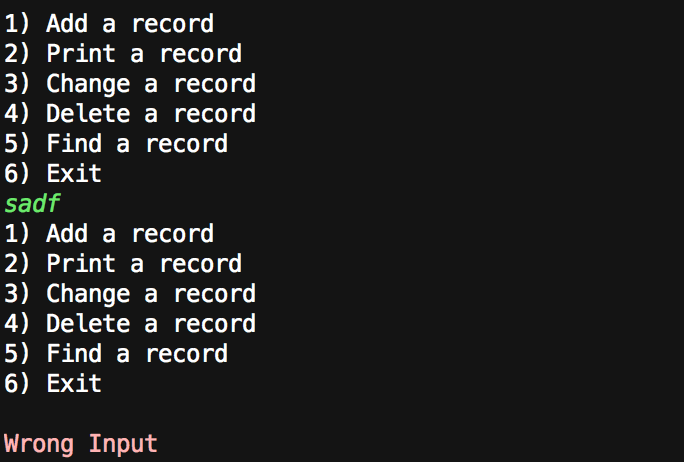
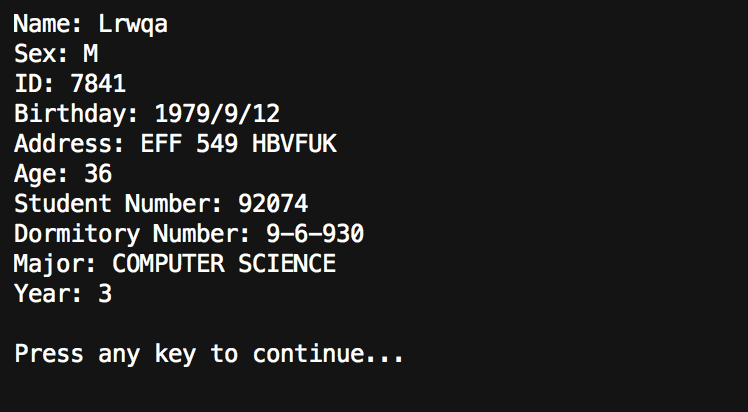
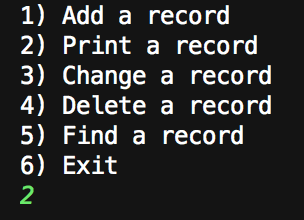
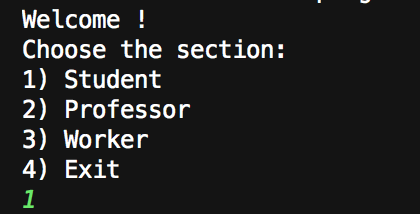
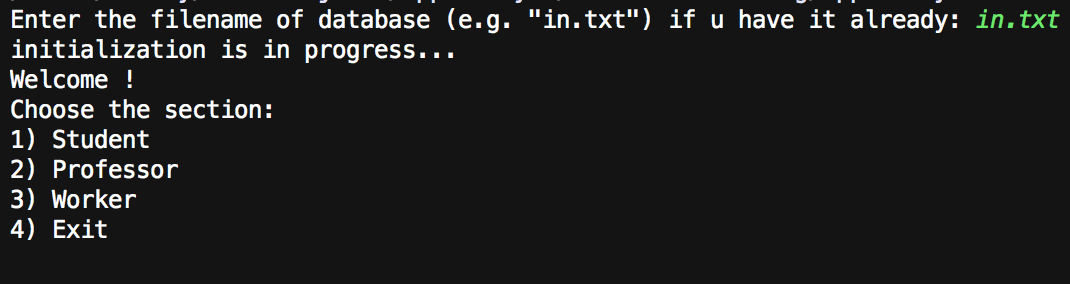
补充笔记：

* 我没有使用throw-catch结构来做到这一点，因为它被认为很慢，已经过时并已被[弃用](https://msdn.microsoft.com/en-us/library/wfa0edys.aspx)好[多次](http://www.open-std.org/jtc1/sc22/wg21/docs/papers/2010/n3051.html)。
* 我没有使用using namespace std，因为它被认为是不好的代码风格，特别是如果我们考虑改进框架。 但因为它是作业而已，我仍然使用using声明，以保持代码可读性并且不会增加行数量。
* 在这个项目中，main（）函数的第一行是完全没有必要性，但是我让它提高了对这个特性的认识，并且潜在地改进了C ++的I / O接口。
* 为了提高可读性，所有的类定义和声明都应该移到不同的头文件中以。

代码（缩短版）  


测试结果

Windows Mac OS



第二道题。计算器类counter。

设计过程描述

当我们只想得到自定义类的多功能性时，定义模板类是合理的，所以类的实例应该有模板类型。 我使用typename T为表示模板类型.

小结

我设计这个类时面临的第一个困难是，在初始化之前，该类的任何新变量都有counter <T>类型，这会带来一些麻烦。任何次我需要重载一个操作符，需要更多的时间来确定这个操作符应该具有哪种类型，并且正在返回的表达式应该被正确地转换。

当数学表达式既具有常规数字又具有模板类的变量时，编译器无法知道如何处理该变量，因此所有操作符都应该对“变量-变量表达式“和“变量-数字表达式”（以及“数字-变量表达式”）重载。任何重载操作符基本上只是获得存储在类实例变量中的值，所以每个操作符都是友函数，并且由于它看起来并不复杂，所以我已经添加了inline指定符提高运营商的速度。

运算符被声明在类的主体之外，因为这看起来有点提高可读性，并且更易于理解模板名称和函数/运算符类型之间的关系。  
每个使用常规数字处理表达式的运算符在模板中都引入了两个类型名称。尽管处理可变数字表达式和数字变量表达式的操作符之间存在很大差异。第一个应该返回与变量相同的类型（counter <T>），这就是为什么在counter类中，模板中只有一个类型名，而类外部有两个单独的“模板”。第二个运算符重载在类体内部和外部的一个template中有两个typename。我在内外使用了typename的不同命名，以使其清晰明了，模板中和类对象中的这些类型不同，但可能相应/连接。例如：

template<typename UU> //意味着稍后在声明参数被模板化  
counter<T> inline operator/(const UU &); //并且counter <T>保持不变的情况下  
 // 它必须具有与该类的对象相同的类型  
or

template<typename TT, typename UU> //意味着稍后在声明参数的类型被模板化  
friend inline U operator/(const UU &, const counter<TT> &);

我不能明确地使用类的模板中的T，因为返回类型可能不同或者类型转换可能会遇到一些错误。 TT显然对应于T，并且稍后在声明中使用T，以使与类模板的连接更加清晰。

任何其他三角函数可以进口/定义为我已经做到了，由于C + 11（和更新的版）更新的<cmath>库。这库支持将模板类型的对象作为参数。

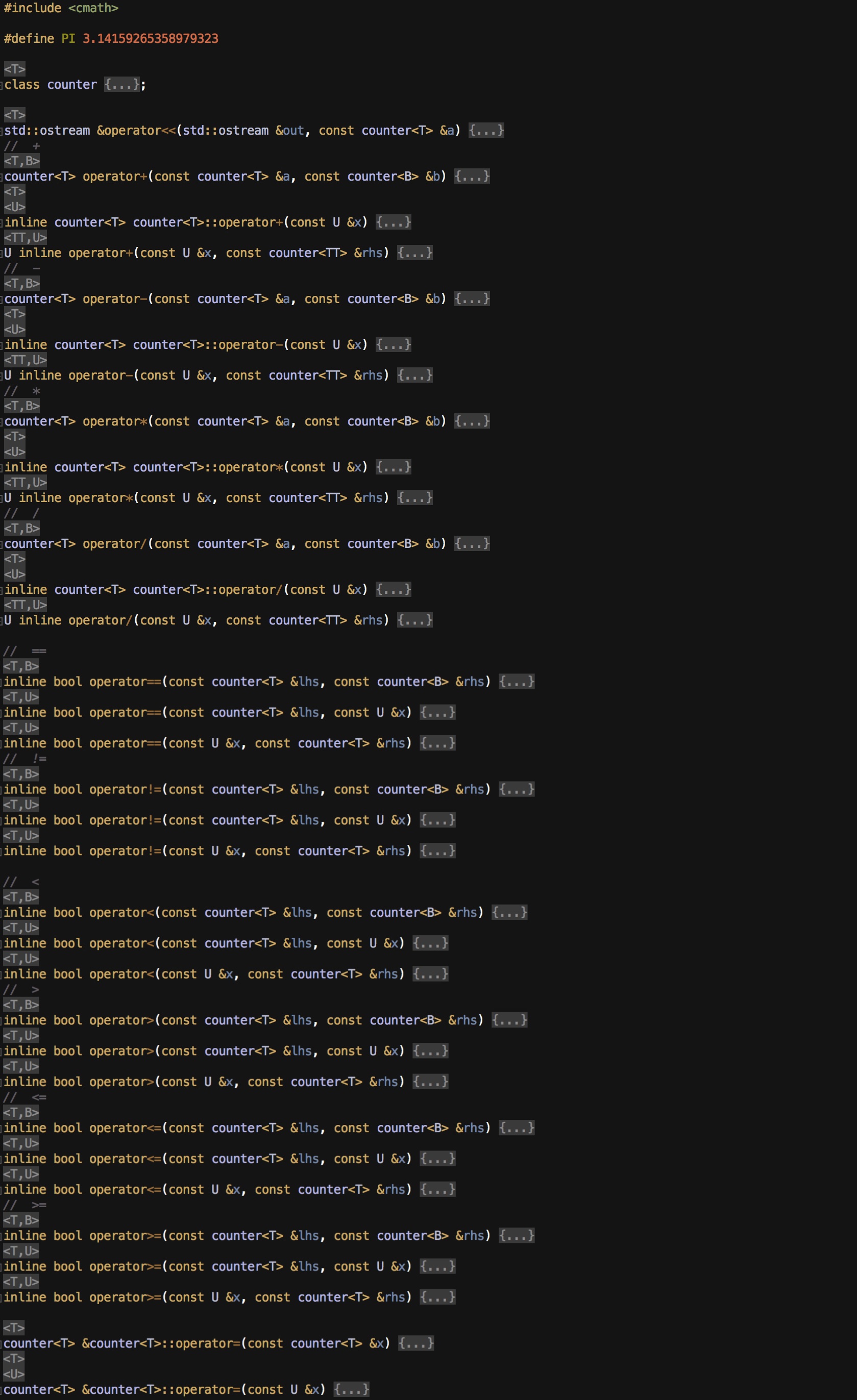
*关于命名补充笔记：  
TT（T）：任何类型; 用于counter类的新变量，它与counter类中的实例val的类型相对应。  
BB（B）：任何类型; 仅在TT（T）已被使用时用于counter类型的变量。  
UU（U）：任何类型的常规号码; 与counter班无关。*

代码（缩短版）

main.cpp

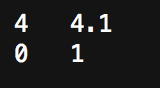
#include <iostream>  
#include "counter.h"  
int main() {  
 counter<double> c;  
 counter a{5};  
 counter b{2};  
 c = 5;  
 std::cout << (a < (1 + c)) << std::endl;  
}

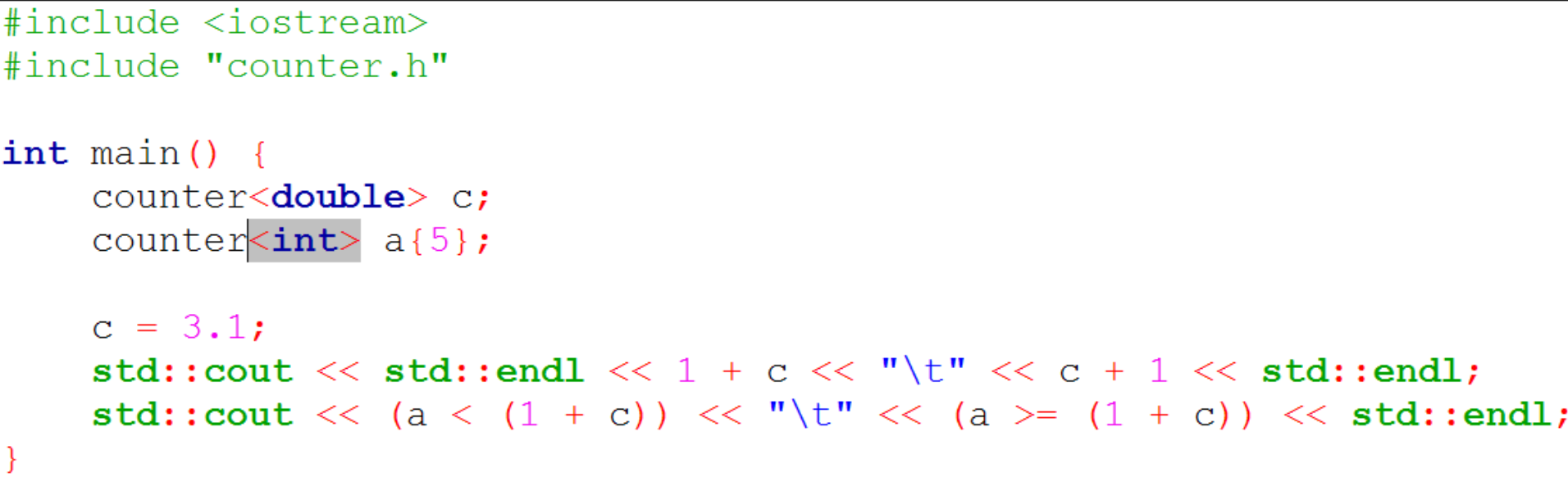
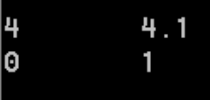
counter.h



测试结果

Mac OS上



Windows上