学号 E42214061 专业 信息安全 姓名 范泽瑞

实验日期 **2024/03/27** 教师签字 成绩

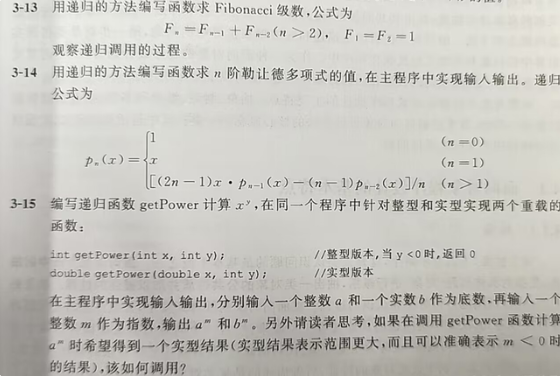
实验报告

【实验名称】 C语言基本编程回顾

【实验目的】

本次实验的目的是通过实践来深入理解C++中的函数递归调用和函数重载。通过回顾C++语言的基本编程，加深对C++语言语法和程序设计的理解。

【题目】



【实验原理】

1. 函数递归：

函数递归调用是指一个函数在它的函数体内直接或间接调用它自身的过程。这种函数称为递归函数。递归调用有两种类型：直接递归和间接递归。直接递归是指函数f直接调用自身，而间接递归是指函数1通过另一个函数2去调用函数1。递归调用对深度有限制，因为每次函数调用都会在内存中占用一定的空间，如果没有终止条件，递归调用会无限进行下去，最终导致栈溢出错误。因此，递归函数必须有一个或多个终止条件，以确保递归过程最终会停止。

1. 函数重载：

函数重载是指在同一个作用域内，可以定义多个名称相同但参数列表不同的函数。这些函数具有不同的参数个数、类型或顺序，以便编译器能够根据传入的参数来确定调用哪个函数。函数重载使得代码更加简洁，避免了命名上的冗余，并提高了代码的可读性和可维护性。通过重载，我们可以为不同的操作或数据类型提供统一的接口，使得函数的使用更加灵活和方便。C++函数重载可以分为以下几种分类：

* 参数个数不同：在同一个作用域中，函数名相同但参数个数不同的多个函数被视为重载函数。
* 参数类型不同：在同一个作用域中，函数名相同但参数类型不同的多个函数被视为重载函数。
* 参数顺序不同：在同一个作用域中，函数名相同但参数顺序不同的多个函数被视为重载函数。

【源码】

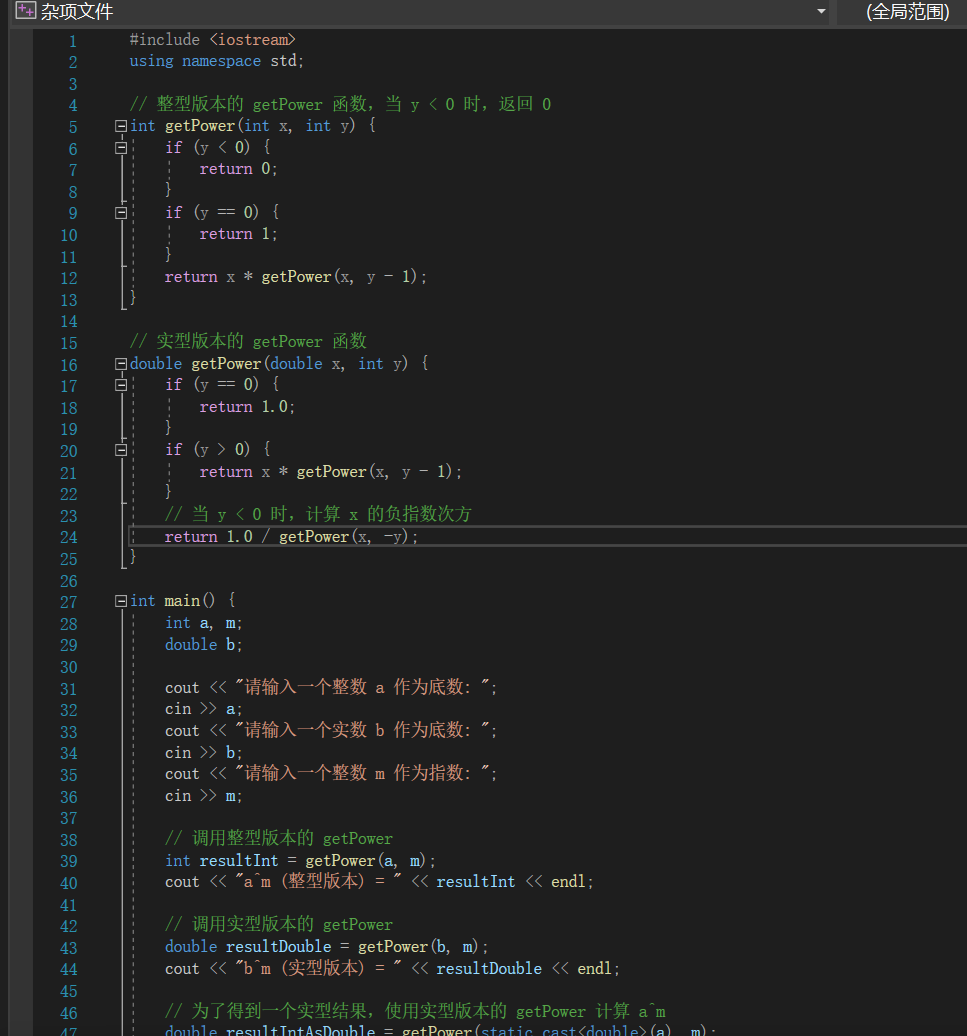
3-13

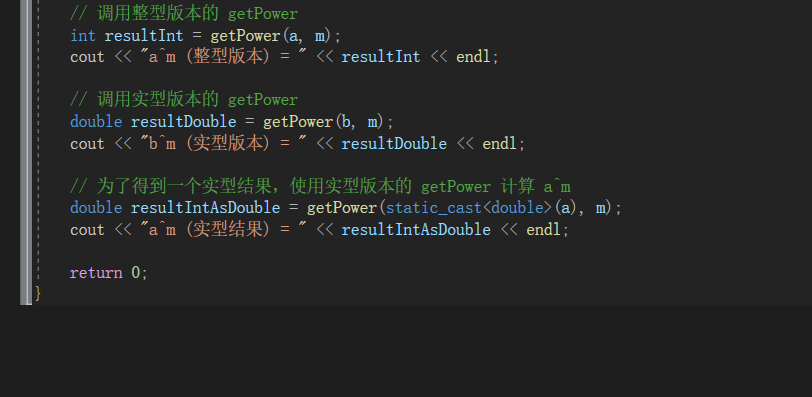


3-14



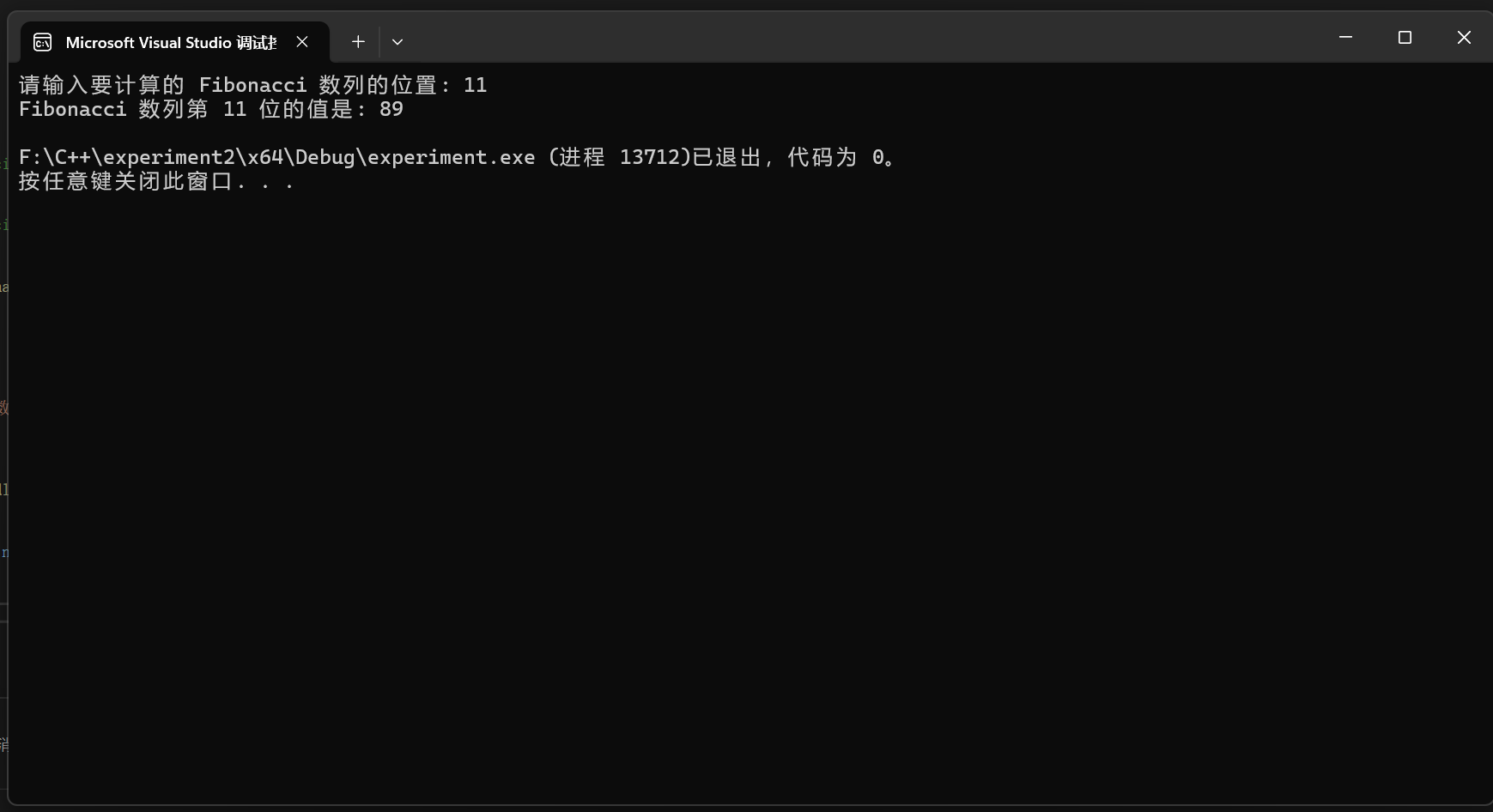
3-15





【运行结果截图】

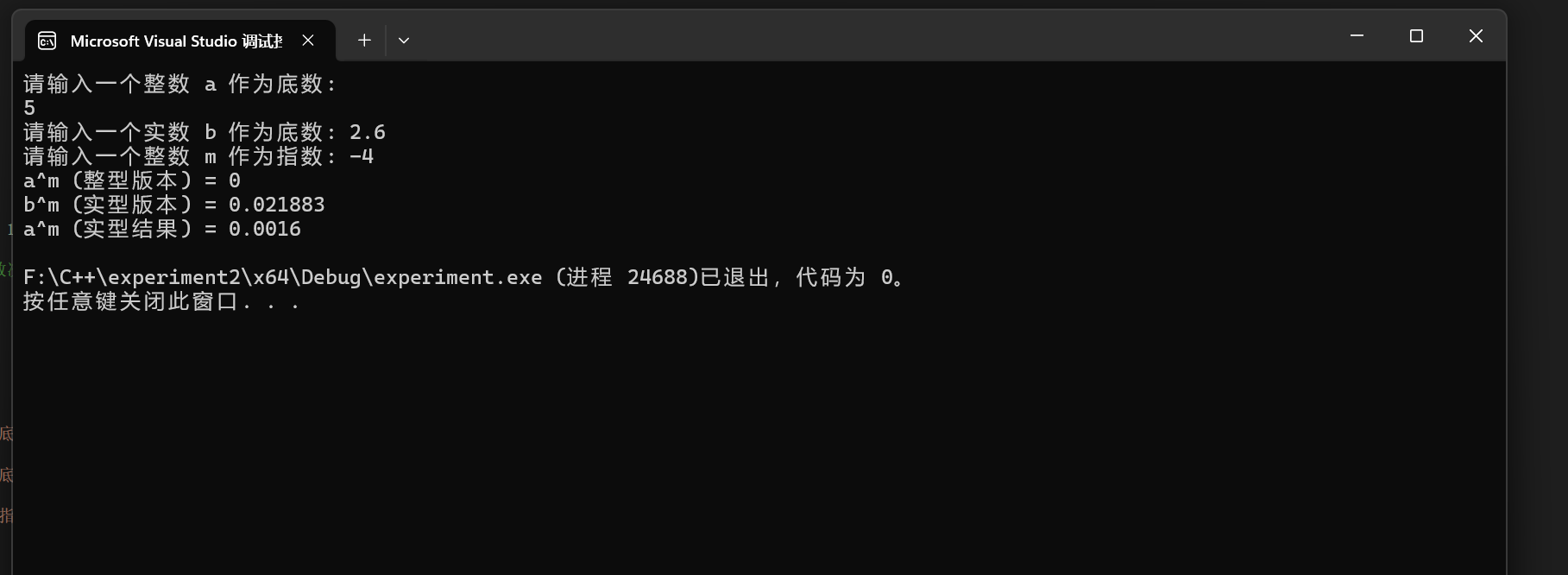
3-13



3-14



3-15



【实验总结】

通过本次实验，我深刻体会到了递归和函数重载在C++编程中的重要性。递归是一种强大的工具，尤其适合解决具有递推性质的问题，但需要谨慎使用以避免栈溢出，无递归终止条件等问题。

函数重载允许我们根据参数的不同来调用不同的函数实现，从而提高了代码的复用性和灵活性。需要注意的是，返回值类型不是函数重载的条件，因为编译器无法通过返回值类型来确定调用哪个重载函数。此外，函数重载必须在同一个作用域内进行，否则编译器无法识别不同作用域中的重载函数。