# 实验三 C++对C的扩展（函数的新知）

## 一、 问题描述

## 

## 实验目的：

 掌握”C++对C扩展“中涉及的若干基本概念和特性，并能够应用于程序编写。

 掌握验证性实验的基本方法和过程(认知、实验、总结）。

## 实验内容：

分别编写一段测试代码来回答任务书中的相关问题（每一个问题，用一个 工程文件，同时需要记录相应的调试过程），具体问题请参考“实验任务 说明03.doc”；

调试的过程；（动态调试的相关截图，比如 设置断点、查看当前变量值等）；

编译出来的可执行程序单独放在一个目录下（bin/exe/debug目录下，同时 附上输入数据说明和输出结果）

## 具体实验

1. **判断题**
   1. 函数带默认参数，以下函数声明情况哪些是正确的

A void Fun(int a=1,int b=2,int c=3);

B void Fun(int a,int b=2,int c=3);

C void Fun(int a,int b=2,int c);

D void Fun(int =1,=2,=3);

答：A和B是正确的，C默认实参不在形参列表的结尾。D没有输入类型说明 符。

1.2、带有默认参数的函数void Fun(int a=1,int b=2,int c=3)，以下调用或说法 哪些是正确的?

A Fun(10,20);

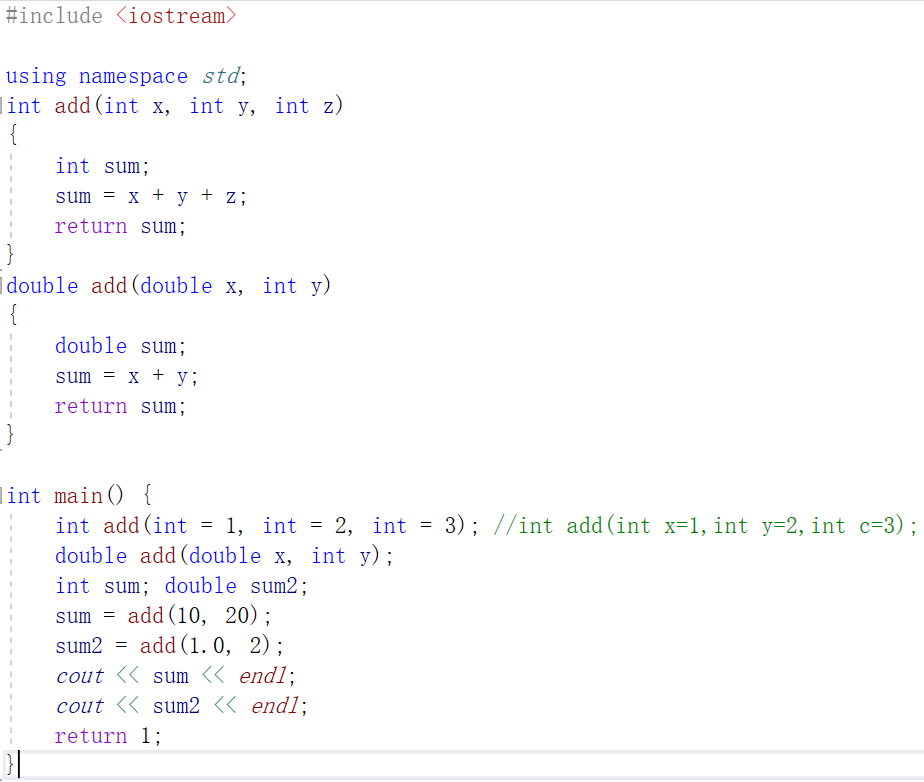
B Fun(10, ,30); //第一和第三个参数用指定值，第二个参数用缺省值

C Fun(10)中实参10为是传递给变量c

答：A是正确的，A通过输入实参使得Fun中的a与b的值取了10和20，代替 了它们原来的默认形参。B是错误的，会提示请输入表达式。C中实参 的10实际上是传给了a，所以C也是错误的。

1. **程序阅读题**

2.1、求结果，并上机验证。简要分析原因

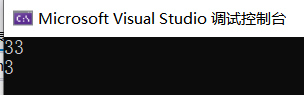


1. 实验结果

33

3

（2）上机验证



（3）A 程序是否能运行？分析下过程

A程序可以运行。分析如下：C++语言允许定义或声明函数时，为函数参数设定默认值。调用时如果给出实参，则用实参初始化形参，如果没有给出实参，则采用预先给定的默认形参值。

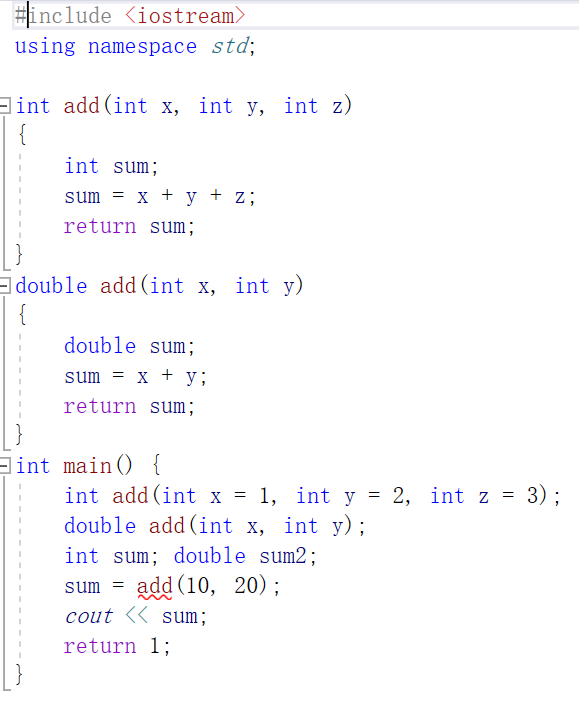
程序中，调用add(10,20)，编译器根据实参的类型判断是调用int add(int x,int y,int z)，这里的x和y被10和20初始化，而由于z没有给给出实参，则采用预先的默认形参值，将得到的结果10+20+3=33返回给sum，即最后输出sun为33。

下一步，调用add（1.0,2），编译器根据实参的类型判断是调用double add（double x，int y）,实参1.0和2被传入函数中，将计算得到的结果3返回给sum2，最后输出sum2为3。

1. B 若把函数定义的首部***int add(int x,int y,int z)*** 修改成 int add(int x=1,int y=2,int z=3)，会如何？

还是和之前一样的输出。C++语言允许定义或声明函数时，为函数参 数设定默认值。不过规范起见，一般在声明函数时为函数参数设定默 认值。

2.2、求结果，并上机验证。简要分析原因

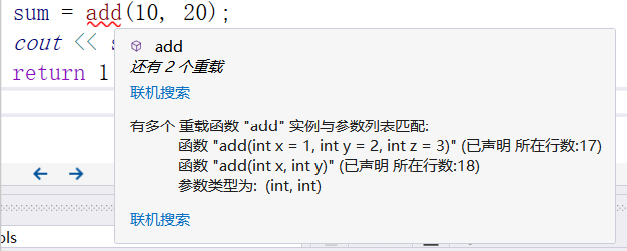


1. 程序是否能运行？分析下原因

程序不能运行，会出现多个重载函数“add”实例与参数列表匹配 的情况。这里int add（int x=1，int y=2，int z=3）的形参都是int型， double add（int x，int y）的形参也都是int型，当调用函数传入10 和20时，编译器无法识别是要调用哪一个函数，故会报错。

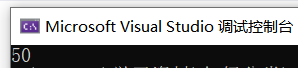
所以，一个函数不能既作为重载函数，又作为有默认参数的函数。 当调用函数时，如果少写一个参数，编译器可能无法判断是调用 重载函数还是调用带默认参数的函数。

1. 上机验证

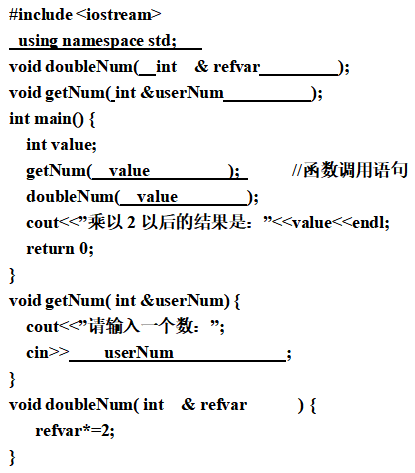


1. 改正代码

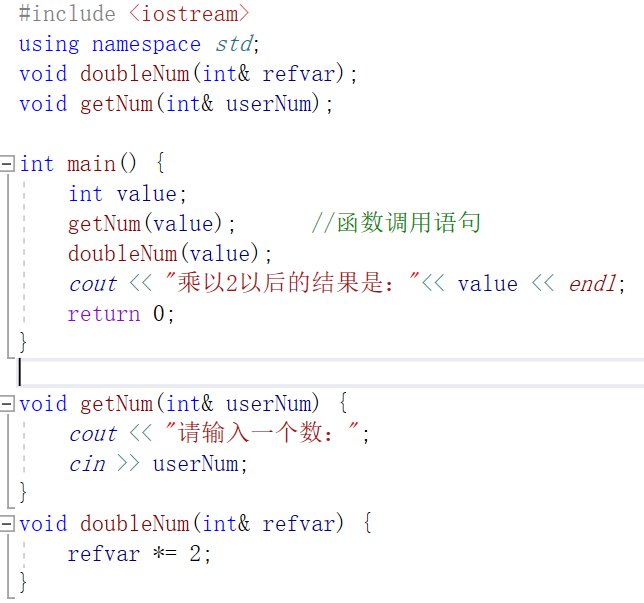
可以在调用add函数时多加一个形参值，如add(10，20，30)， 程序成功运行。



1. **程序填空题**
   1. 函数getNum要求用户输入一个值，并存储在引用userNum中。userNum是main定义的变量value的引用。
2. 填空



（2）上机验证

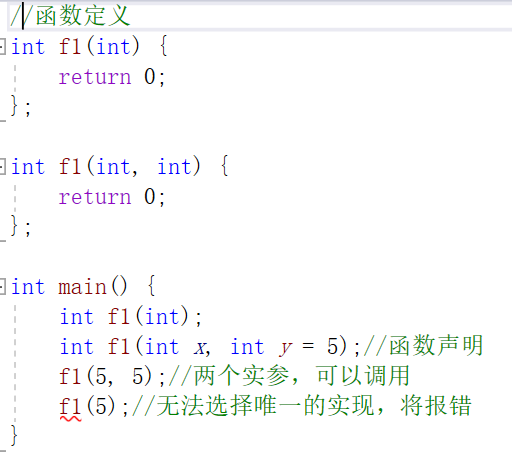


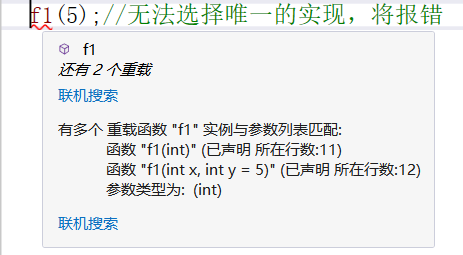


1. **简答及程序验证**
   1. 分析函数重载和函数参数默认可能导致二义性的问题，编写程序来验证（要求：将错误语句作为注释标记在函数体出错语句部分）。简述思考：对编程的启发。

答：一个函数不能既作为重载函数，又作为有默认参数的函数。当调 用函数时，如果少写一个参数，编译器可能无法判断是调用重载函数 还是调用带默认参数的函数。

比如，下图中定义了两个重名函数f1()实现重载，按参数个数的不同进 行选择。而如果再引入默认参数，如注释中所示，此时，出现一个实 参的f1(2)函数调用，将无法选择唯一的实现，编译器将报错。

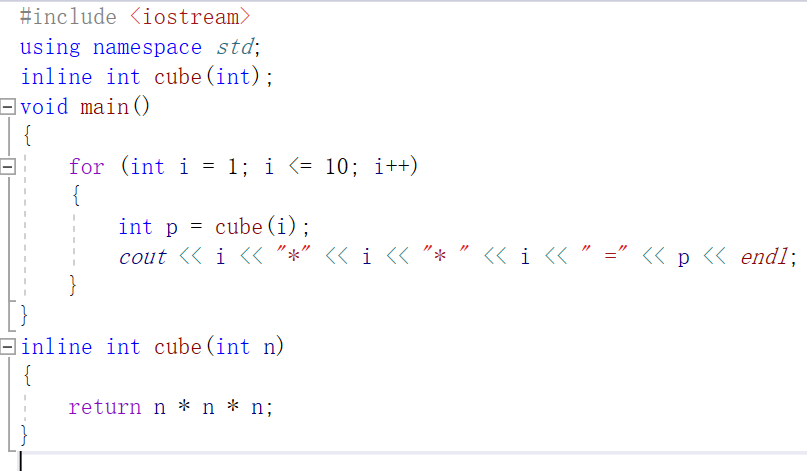




思考：应尽量避免设置默认参数，以保证重载函数的正常选择。

* 1. 请描述inline关键字书写的位置（函数的声明部分？函数的定义部分？）

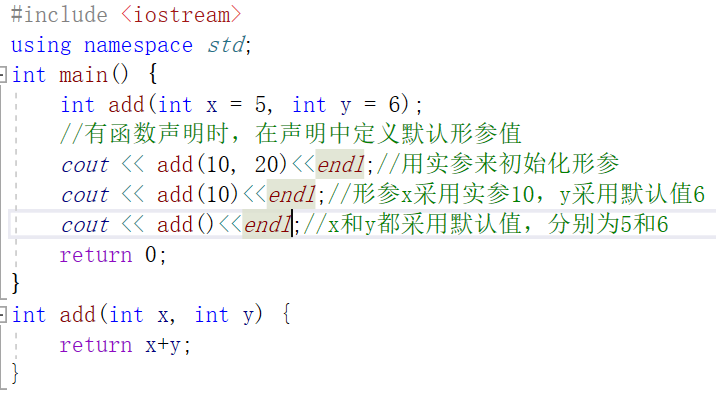
答：在函数定义和函数声明的函数头前面加关键字inline，其他与一般 函数相同。如下图所示。在程序中出现的内联函数的调用将用该 函数的函数体代替，而不是转去调用该函数，内联本质是一函数。

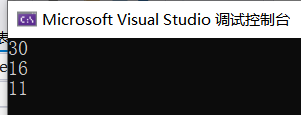




* 1. 描述函数参数默认值书写的位置（函数的声明部分？函数的定义部分？）

答：C++语言允许，定义或声明函数时，为函数参数设定默认值。如果 一个函数需要声明时，默认的参数值应设置在函数的说明语句中，而 不是函数的定义中。如果没有函数声明时，默认的参数值可设置在函 数的定义中。

一个函数有多个参数时，可以给该函数的部分参数或全部参数设置默 认值。在给函数的部分参数设置默认值时，应该从参数表的右端开始， 在设置了默认值的参数的右端不允许出现没有默认值的参数。演示程 序如下：  
 



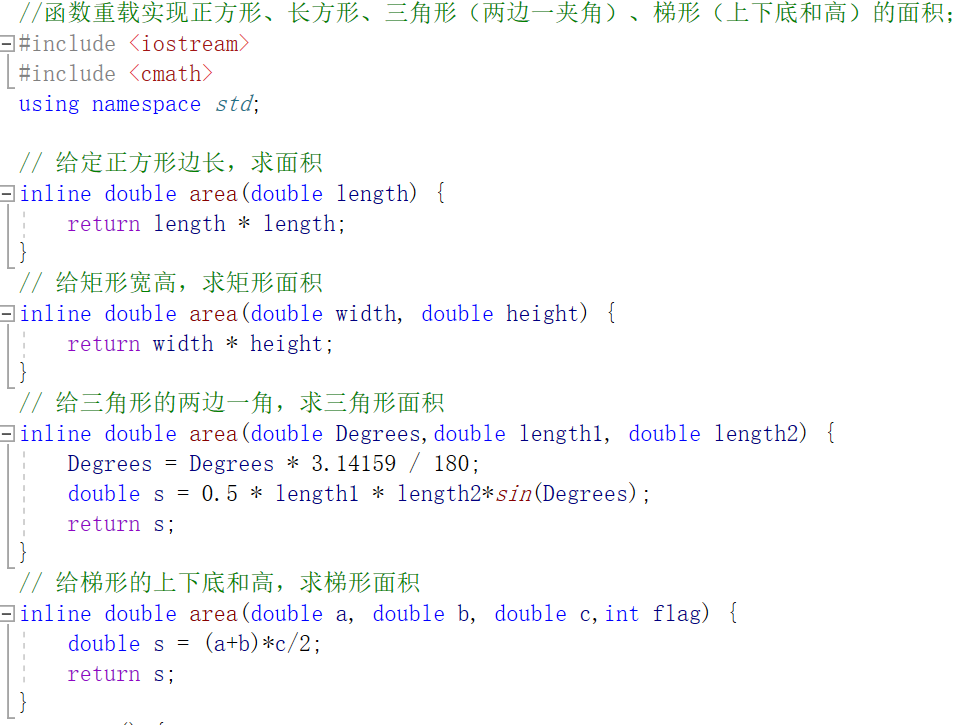
1. **程序设计题**
   1. 编写C++程序，使用函数重载实现正方形、长方形、三角形（两边一夹角）、梯形（上下底和高）的面积；

答：（1）实现思路

程序使用重载函数area()，相同函数名重载函数area()的四个函数具有不 同个数的参数，四次调用area函数的参数个数不同，系统会根据参数的 个数找到与之匹配的函数并调用它。由于梯形和三角形所需要的参数 都是3，类型也都是double，所以在求梯形面积的area函数参数中加了 一个int类型的参数flag以便区分。

（2）实验代码

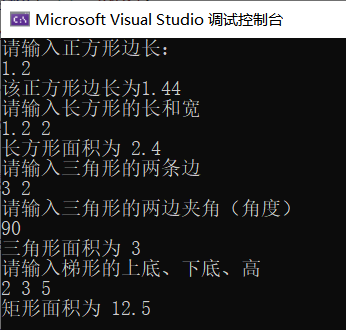
① 求正方形、长方形、三角形、梯形面积的重载函数



② main函数，用于用户输入输出与函数调用



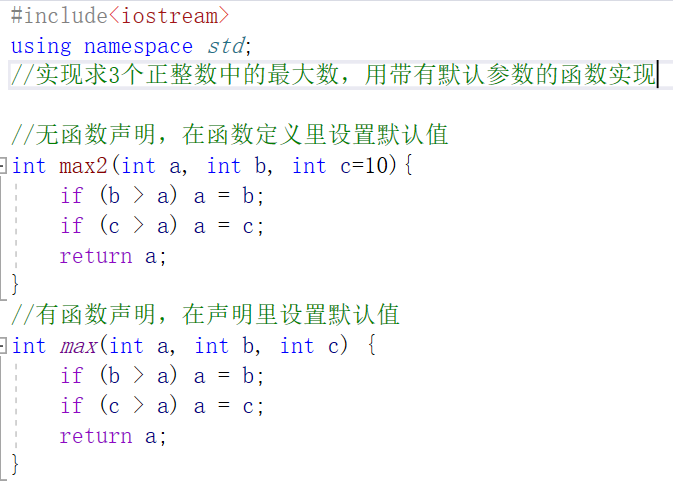
③ 运行结果



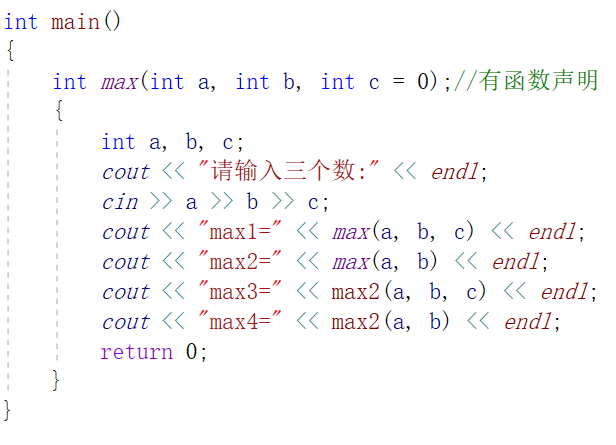
* 1. 编写程序，实现求3个正整数中的最大数，用带有默认参数的函数实现；

分别实现：没有函数声明、有函数声明两种情形，最后分析规律： 在函数定义中，函数头部关于默认形参的书写原则）

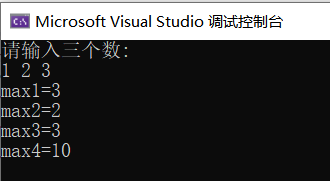
1. 实验代码
2. 有函数声明的max函数与无函数声明的max2函数



1. main函数，用于用户输入输出与函数调用



1. 运行结果



1. 分析规律

① C++语言允许定义声明函数时，为函数参数设定默认值。如果 一个函数需要声明时，默认的参数值应设置在函数的说明语句中， 而不是函数的定义中。如果没有函数声明时，默认的参数值可设 置在函数的定义中。

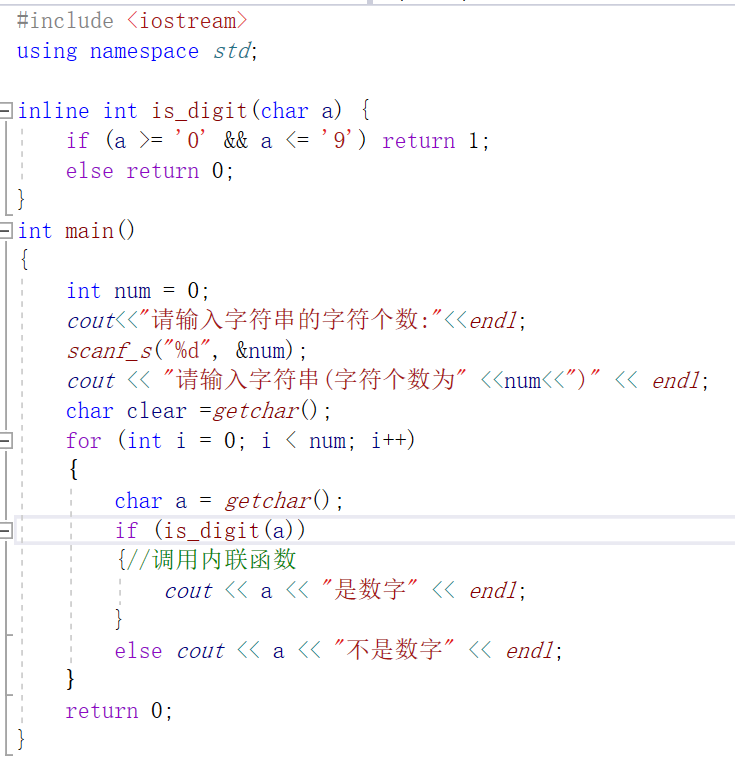
如上面程序中，max函数有函数声明，则其默认的参数值c=0设置 在了函数的说明语句中，max2函数没有函数说明，默认的参数值 可以设置在max2函数定义中。

②一个函数有多个参数时，可以给该函数的部分参数或全部参 数设置默认值。在给函数的部分参数设置默认值时，应该从参 数表的右端开始，在设置了默认值的参数的右端不允许出现没 有默认值的参数。

如上面程序中，默认值c=0就是放在最右边的。调用max（a,b,c） 时，将实参1，2，3传入max函数，故输出的最大值max1为3。 调用max(a,b)时，仅将实参1，2传入，c取默认值0，故输出最 大值max2为2。调用max2（a,b,c）时，将实参1，2，3传入max2 函数，故输出的最大值max3为3。调用max2（a,b）时，将实参 1，2传入max2函数，c取默认值10，故输出的最大值max4为10。

* 1. 从键盘上输入任意多个字母，判断每个字符是否为数字字符。要求判断字符性质的函数描述为内联函数。

1. 实验代码



（2）实验结果



（3）思考

内联函数在函数头前面加关键字inline，其他与一般函数相同，内联本质是一函数。在程序中出现的内联函数的调用将用该函数的函数体代替，而不是转去调用该函数。

内联函数使用的场合：优化程序，提高效率。把函数短小而又频繁调用的函数声明为内联函数。函数体适当小，这样就使嵌入工作容易进行，不会破坏原调用主体。程序中特别是在循环中反复执行该函数，这样就使嵌入的效率相对较高。

比如上面程序中，在for循环中反复执行内联函数，使嵌入的效率相对较高。

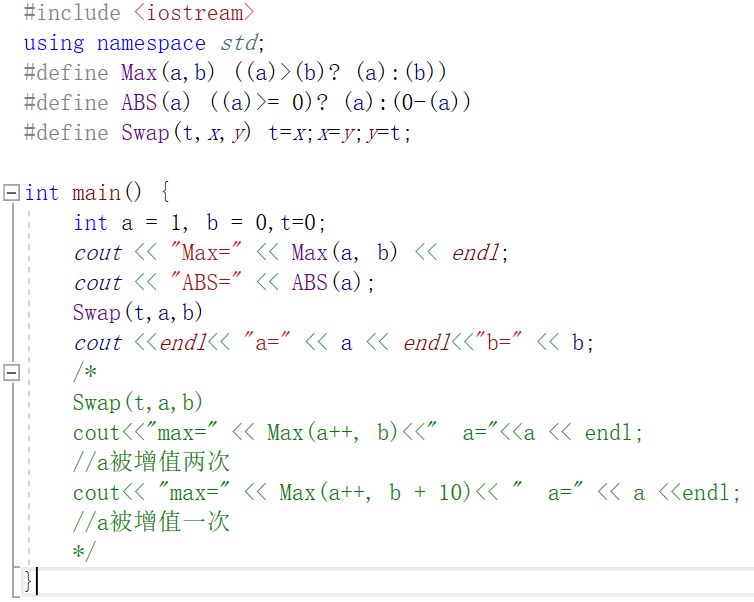
* 1. 将下列的内联函数修改为带参数的宏，给出带main函数的完整代码，并描述执行过程

1、inline int MAX(int a,int b) { if (a>b) return a; return b;}

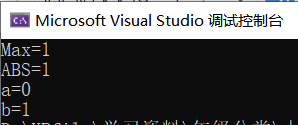
2、inline float ABS(float a) { return (a>=0)?a:0-a;}

3、inline void SWAP(int &a,int &b) { int t; t=a; a=b; b=t; }

1. 实验代码



1. 实验结果



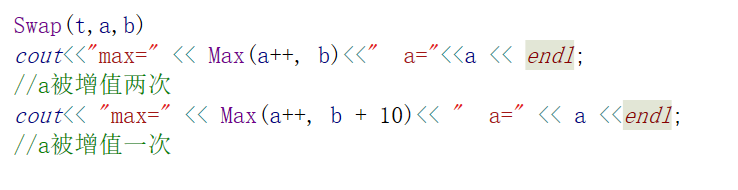
1. 执行过程

宏定义属于预处理命令，在编译过程中的预处理阶段处理。宏定义只是单纯的替换，比如在main函数中将Max(a,b)直接用((a)>(b)? (a):(b))替换，将ABS(a)直接用((a)>= 0)? (a):(0-(a))替换，将Swap（t,x,y）直接用t=x;x=y;y=t;替换。

所以对a，b进行比较，输出的Max就是1，对a取绝对值，输出的ABS即是1，将a，b进行交换，输出a=0，b=1。

1. 思考

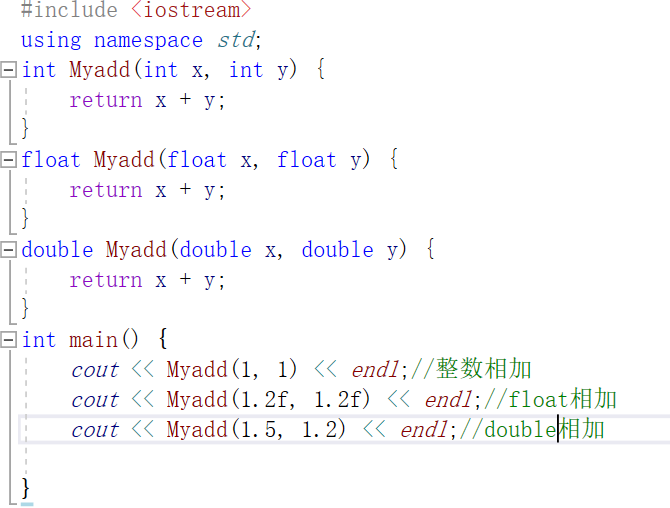
宏定义语句的书写格式有过分的讲究，Max与括号之间不能有空 格，所有的参数都要放在括号里。Max( )函数的求值会由于两个参 数值的大小不同而产生不同的副作用，如下面代码所示。



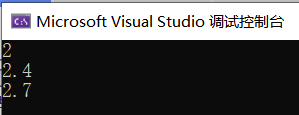


Max(a++,b)的值为2，同时a的值加2。Max(a++,b+10)的值为10，同 时a的值加1。作为改进，可以通过一个内联函数得到所有宏的替 换效能和所有可预见的状态以及常规函数的类型检查。

1. **程序设计题**
   1. 通过对Myadd()的函数重载，实现多个相同数据类型的两个数据的和（包括：int、float、double等）。在此基础上，发现这些函数体之间有什么共性？思考，是否还有更高级的编程机制？
2. 实验代码



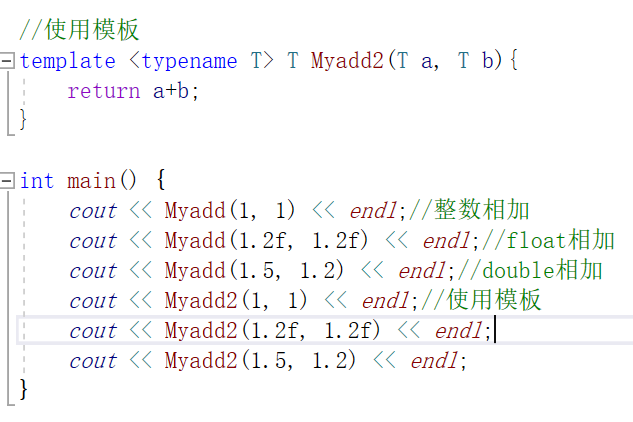
（2）运行结果

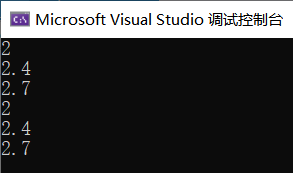


（3）思考

这些功能相同的函数名字相同，各自的函数体不同，对应着不同类型的数 据操作。更高级的编程机制可以使用C++的模板。

模板是泛型编程的基础，泛型编程即以一种独立于任何特定类型的方式编 写代码。模板是创建泛型类或函数的蓝图或公式。于是上面的代码可以更 改为：





其中t是函数所使用的数据类型的占位符名称。

## 附录

源程序文件项目清单： 2.1 2.2 3.1 4.1 4.2 4.3 5.1 5.2 5.3 5.4 6.1