C++对C的扩展（第二次）

**黄勖 22920212204392**

**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02.doc”；

1. **程序阅读题**

* **1.1**

**输出结果：**

**Value=1**

**Value=2**

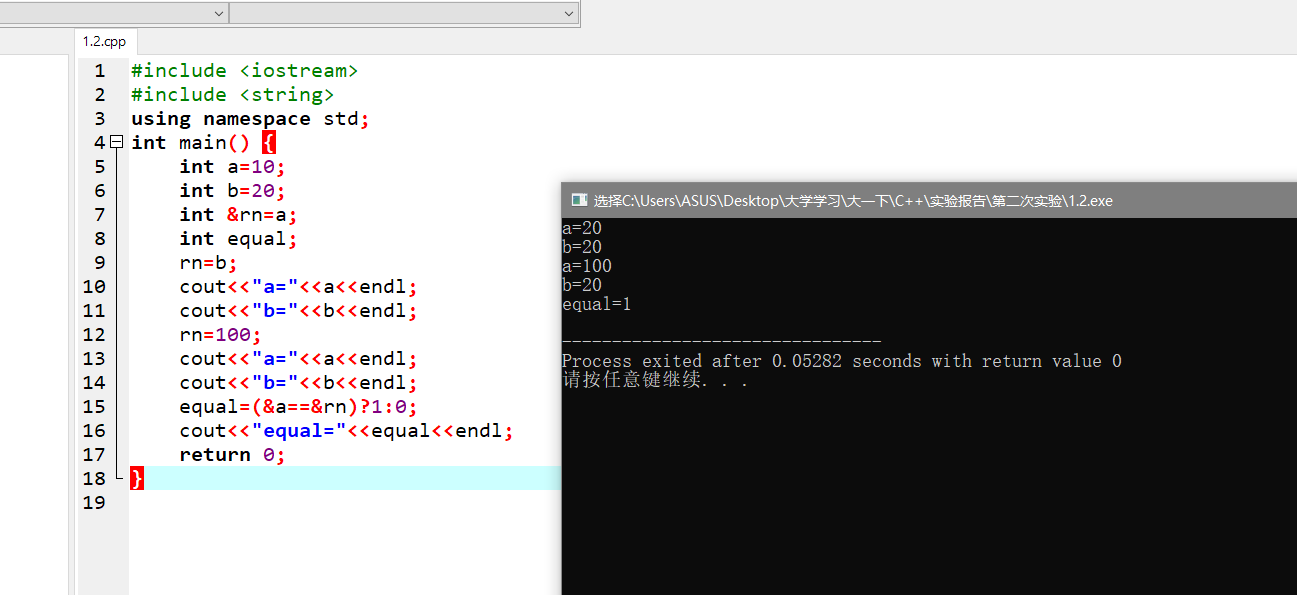
**上机验证：**



**原因：**

第一行输出 ”cout<<"Value="<<value<<endl;” 中程序调用main函数中的局部变量进行输出 所以输出1；

第二行输出调用**printvalue()** 函数进行输出 调用的变量是全局变量，又因为在第十一行调用全局变量进行对value的修改 所以输出2；

* **1.2**

**分析：**

a=20 //在rn=b中 a被赋值20

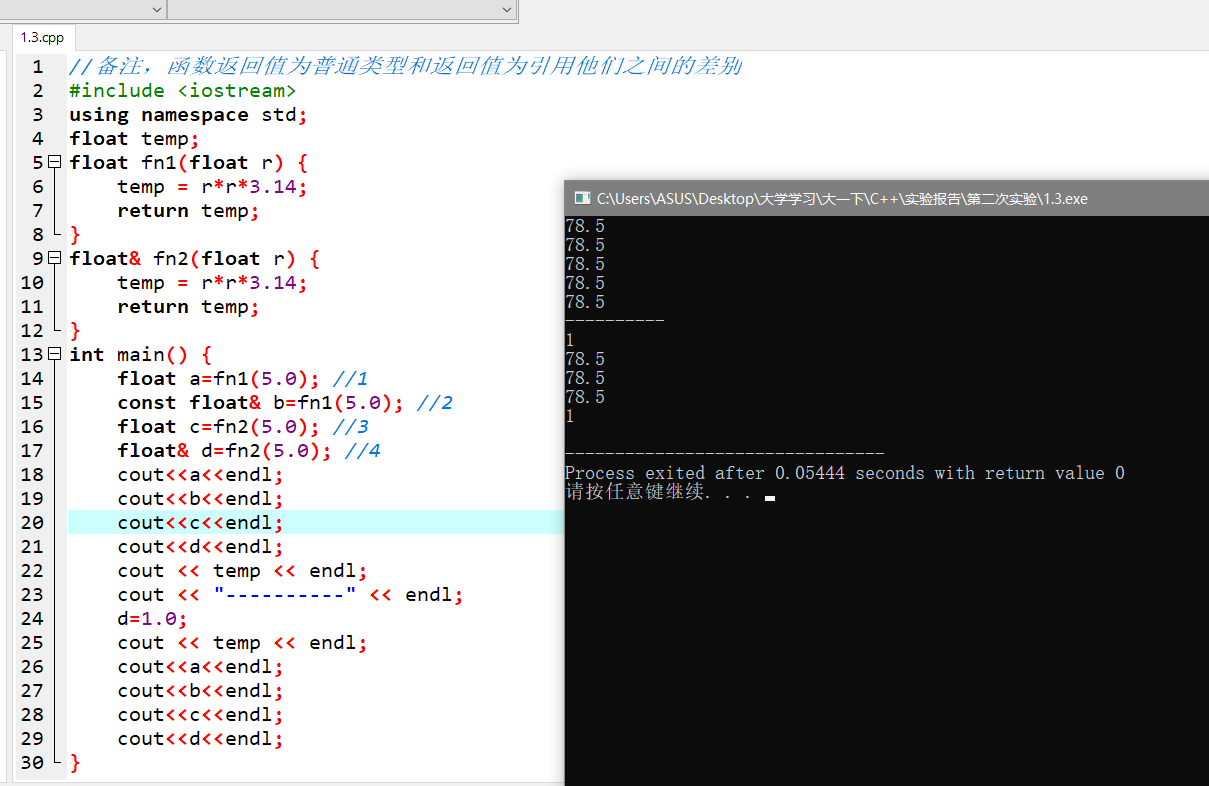
b=20 //b初值便为20

a=100 //在rn=100中 a被赋值20

b=20 //b初值便为20

equal=1 // 在 equal=(&a==&rn)?1:0; &a==&rn地址相等 故equal被赋为1

* **1.3**



（源代码有问题无法编译 代码经过修改）

**分析：**

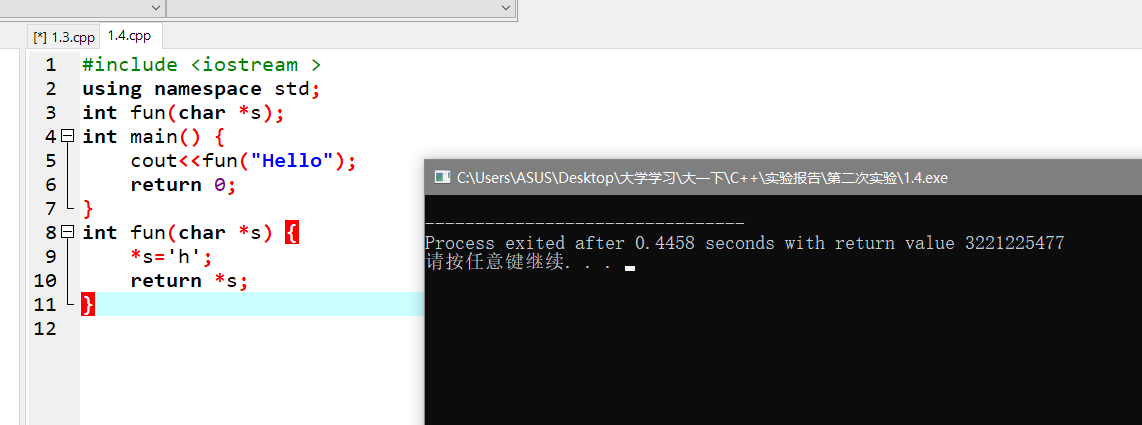
**float a=fn1(5.0); //1 单独变量存储 fn1(5.0);**

**const float& b=fn1(5.0); //2 fn1(5.0)返回的局部变量temp不能被c++接受为引用（局部变量不能被main修改） 故用const**

**float c=fn2(5.0); //3 申请变量c存储temp的当前值**

**float& d=fn2(5.0); //4 d为对temp的引用**

* **1.4**

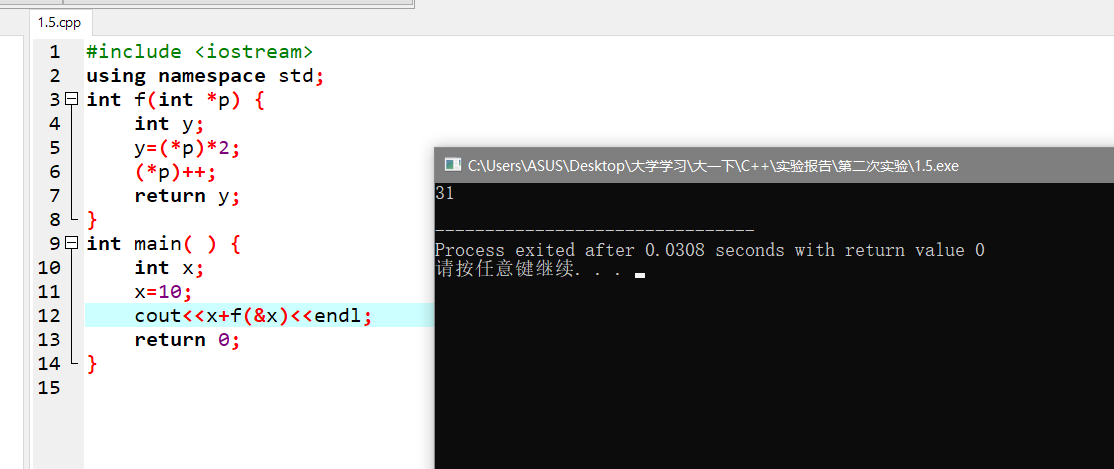


**CPU错误 该程序无法正常运行**

**"hello"是字符串常量，拿来赋值时这个字符串常量返回的是首字符的地址，这个地址上的值（字符）是常量，不可变。而a是字符指针变量，指向的字符变量是可变的。**

**如果普通字符指针（指向字符变量）指向了一个不可变的内存区域，那这块内存的值到底是变呢还是不变呢？当然是不变了， 这是人家本身的性质，因此C++禁止把字符串常量的首地址赋值给字符指针变量。**

* **1.5**



**X=11 f(&x)=20**

**(源代码无法编译)**

**二、程序设计题**

* 2.1 分别用指针和引用写程序，程序中包含一个子函数Myswap()，该函数能交换两个实参变量的值

**见“Myswap\_指针.cpp”与“Myswap\_引用.cpp”**

* 2.2 声明一个复数结构体类型struct Complex，

（一）编写函数：

1、set\_value()：该函数可实现对复数值的设定；

2、display()：该函数可实现对复数的输出；

3、void addi1()：实现两个复数的和；

4、struct Complex addi2()：实现两个复数的和；

5、struct Complex & addi3()：实现两个复数的和；

**编写代码见Complex.cpp**

（二）重点分析：

1. 在set\_value()的参数传递应该采用哪种参数传递的方式？并分析原因

传递引用 这样才能正确给参数赋值

1. 在display()形参中使用const &的优点

使用常量引用 不会改变原有的complex参数的值

1. addi1()和addi2()函数功能目标一样，返回值不一样。从返回值视角分析，addi2()有什么特性？

addi2返回了complex类型的计算值 可以直接赋值给结果变量

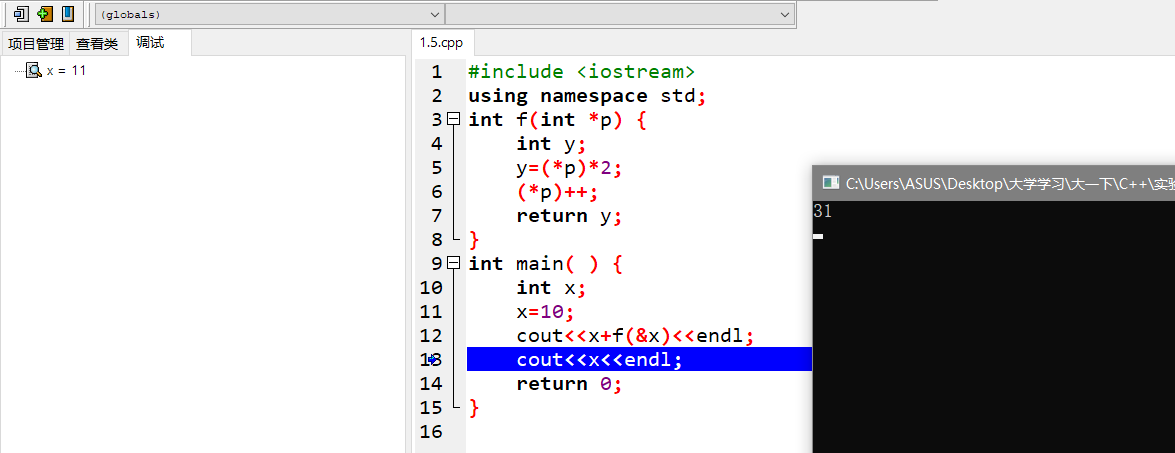
1. addi2()和addi3()函数功能目标一样，返回值不一样。从返回值视角分析，addi3()有什么特性？
2. Addi3返回的是complex的引用 增强了执行效率
3. 分析并验证，addi2()函数是否能作为左值？

经过测试 可以

1. 验证addi3()函数是否能作为左值？分析代码的执行过程。

经过测试 可以；运行时addi3函数对两个复数进行相加 返回结果后 右值再将值赋给左值

* 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）

输入数据说明和输出结果见上方图片。