C++ Primier Plus Stephen Prata

第一章 开始学习C++

绪

1.函数原型：只描述函数接口。描述的发送给函数的信息和返回的信息。

2.函数定义：包含了函数的代码。

应在首次使用函数之前提供其原型，通常的做法是把原型放在main()函数定义的前面。

3.在C和Pescal中，所有的变量声明通常都位于函数或过程的开始位置，但C++没有这种限制，C++可以在首次使用变量前声明它。

using

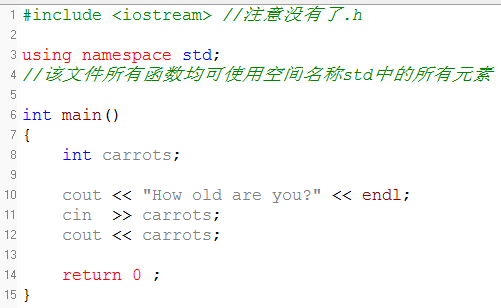
要访问名称空间中的方法（例如cout）

1.将using namespace std;放在函数定义之前，让文件中的所有函数都能使用名称空间std中的所有元素。

2. 将using namespace std;放在特定的函数定义中，让该函数能够使用名称空间std中的所有元素。

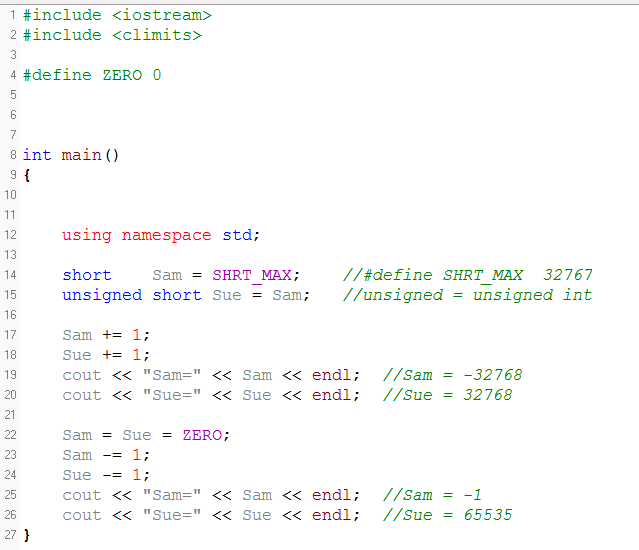
3.在特定的函数中使用类似using std::cout;这样的编译指令，而不是using namespace std;让该函数能够使用指定的元素，如cout。

4.完全不使用编译指令using,而在需要使用名称空间std中的元素时，使用前缀std::,如下所示：std::cout << “i” << std::endl;



第二章 处理数据

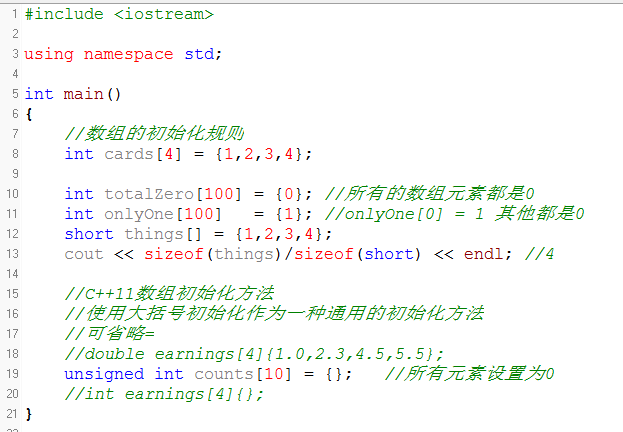
无符号类型



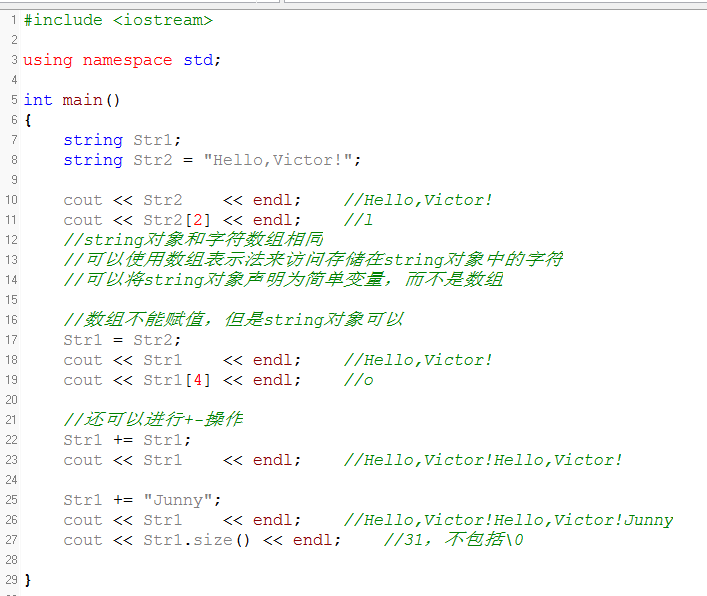
//0-> +32767 -> (+32768不存在) -32768 重置点 -> -1 -> 0

第三章 复合类型

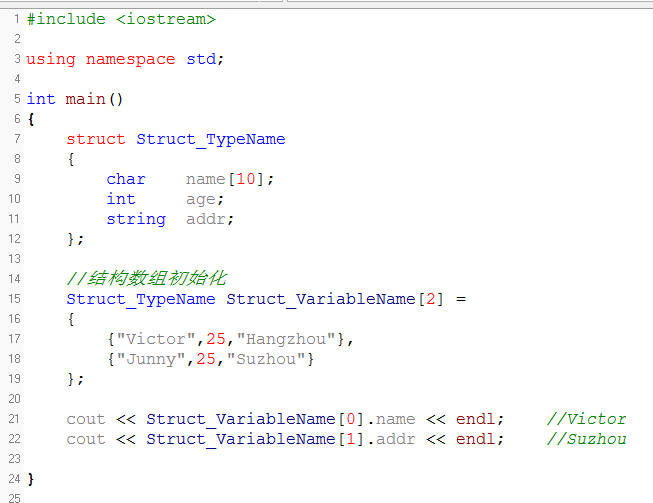
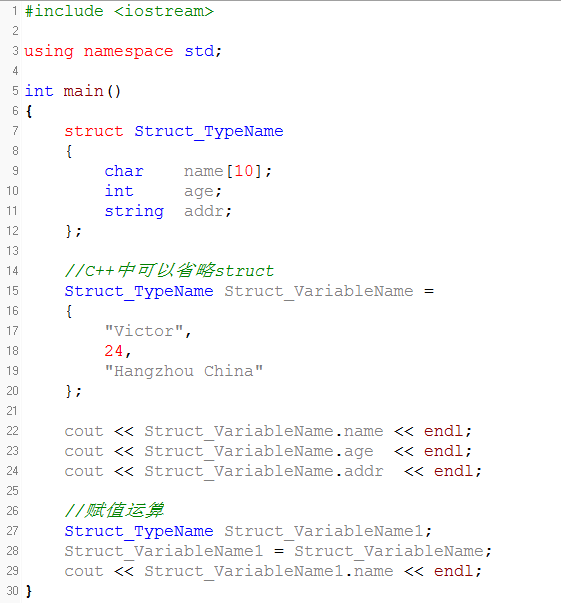
数组



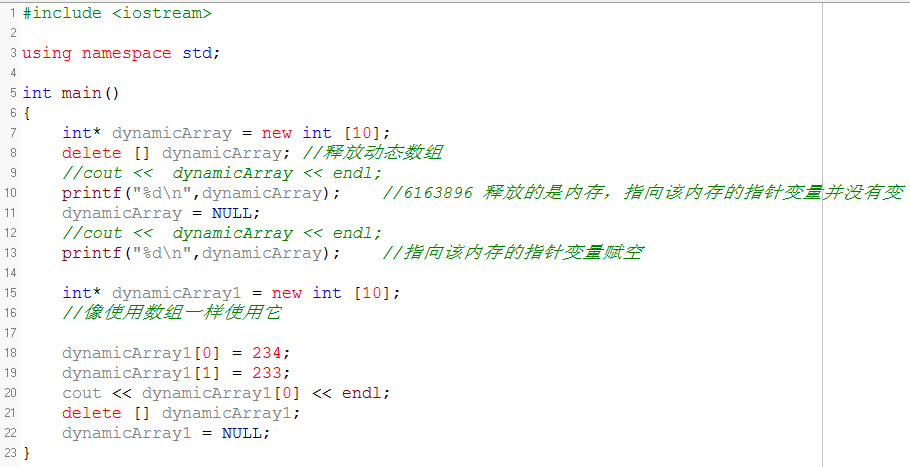
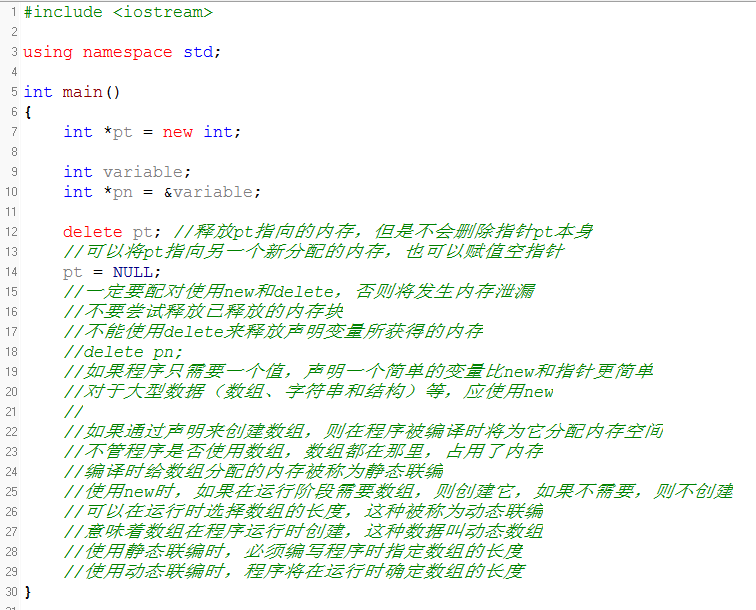
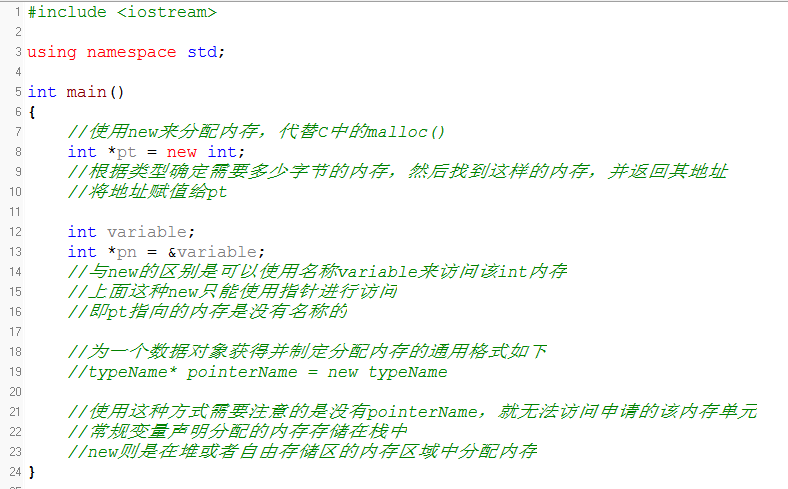
String类



结构体

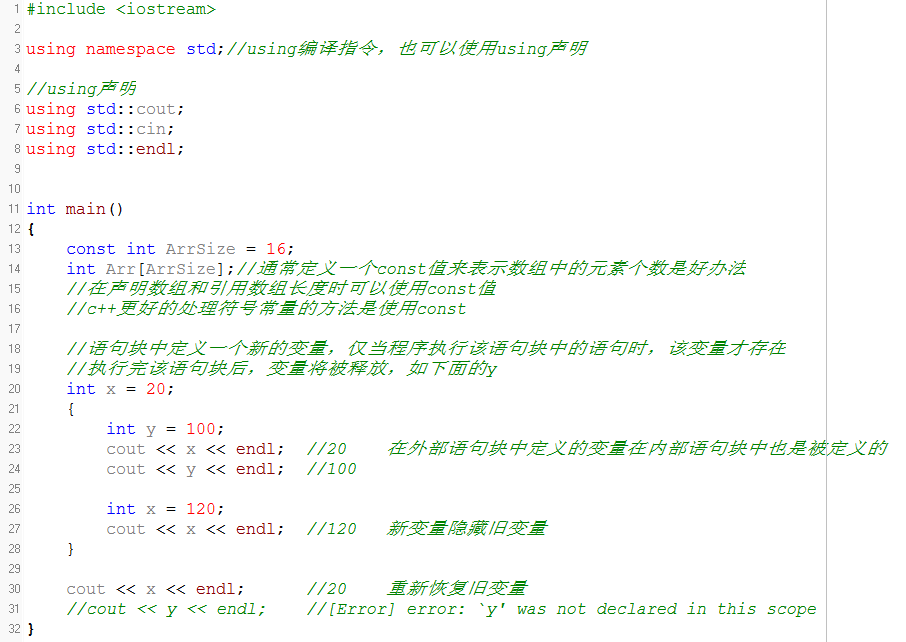


指针

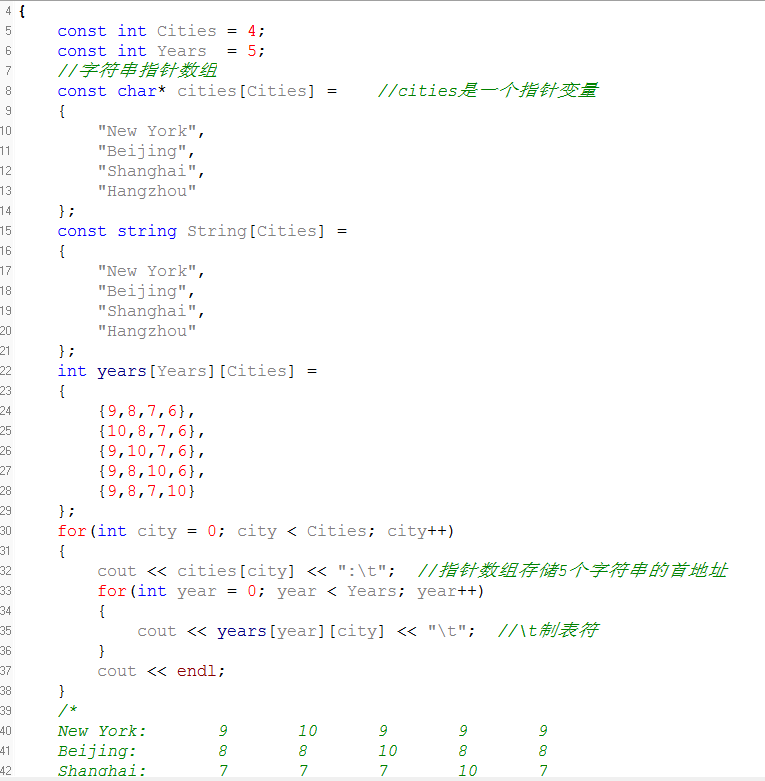
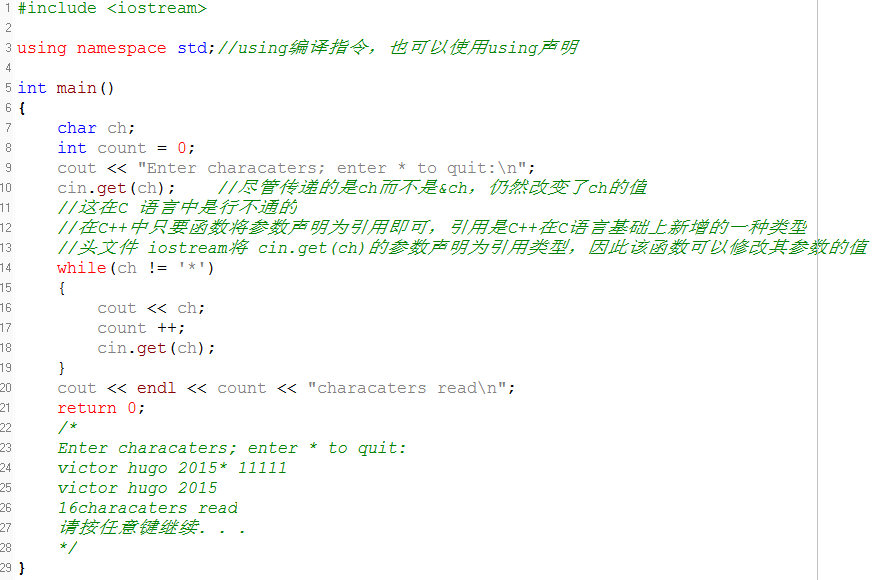
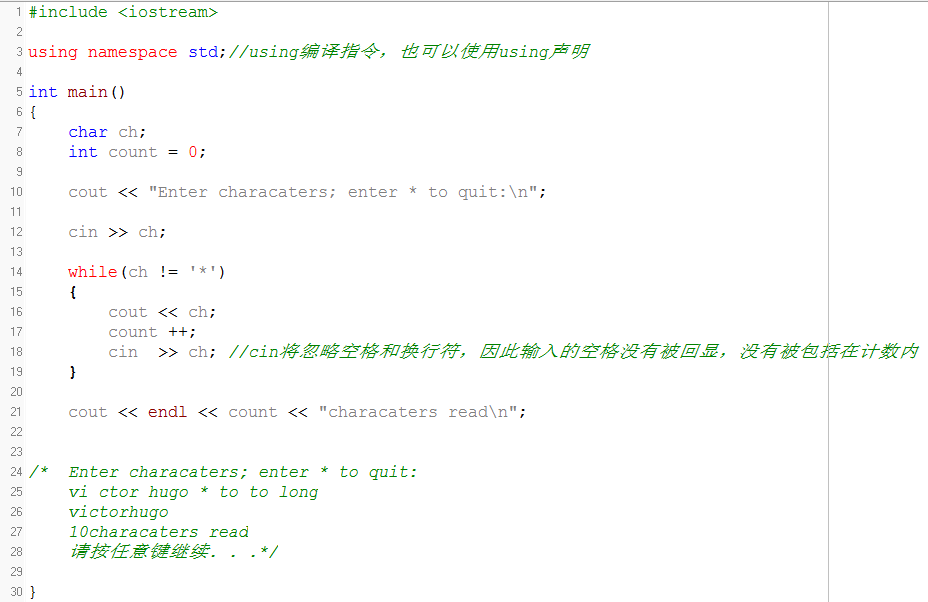
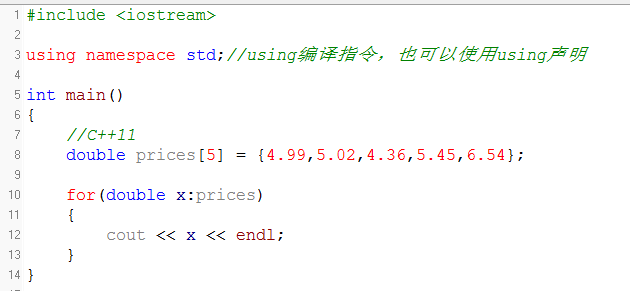


第四章 循环和关系表达式

复合语句（语句块）

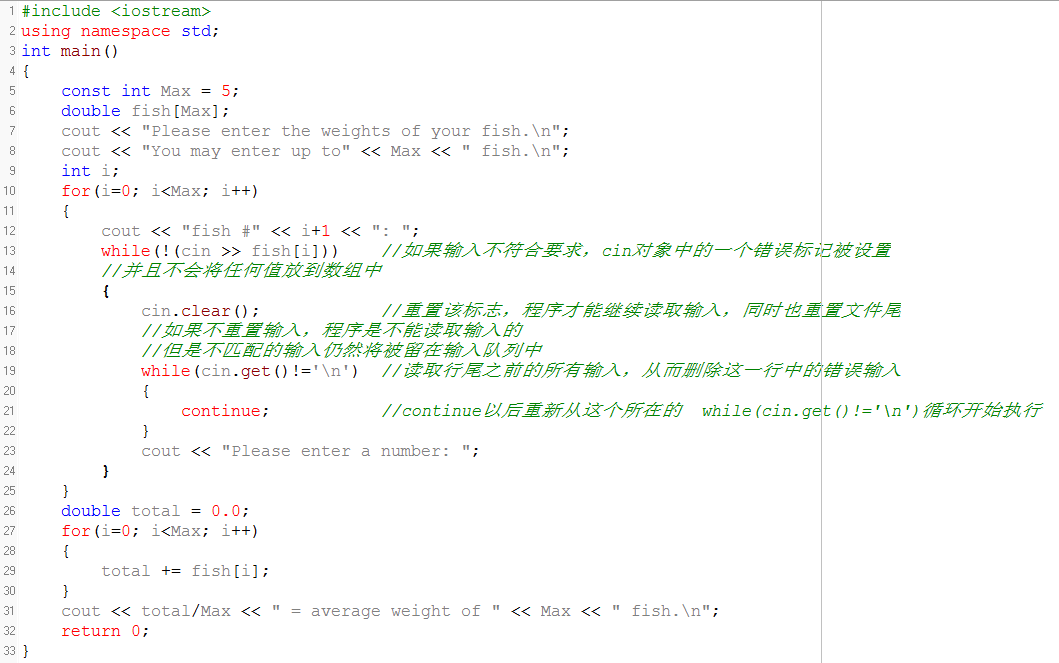
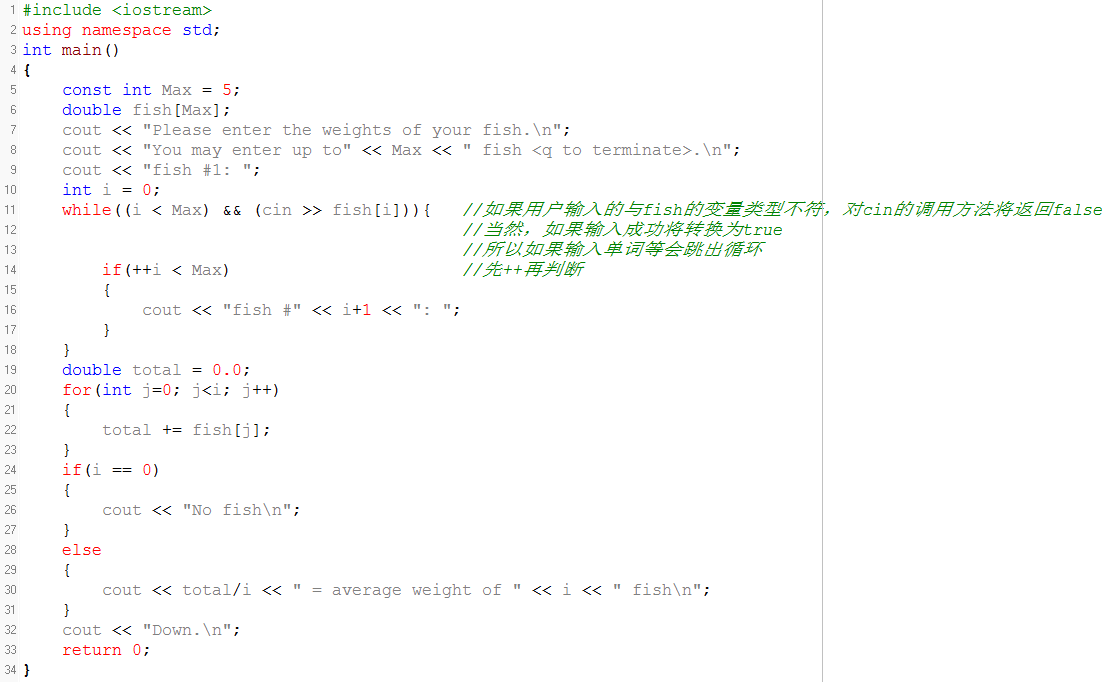


循环和文本输入

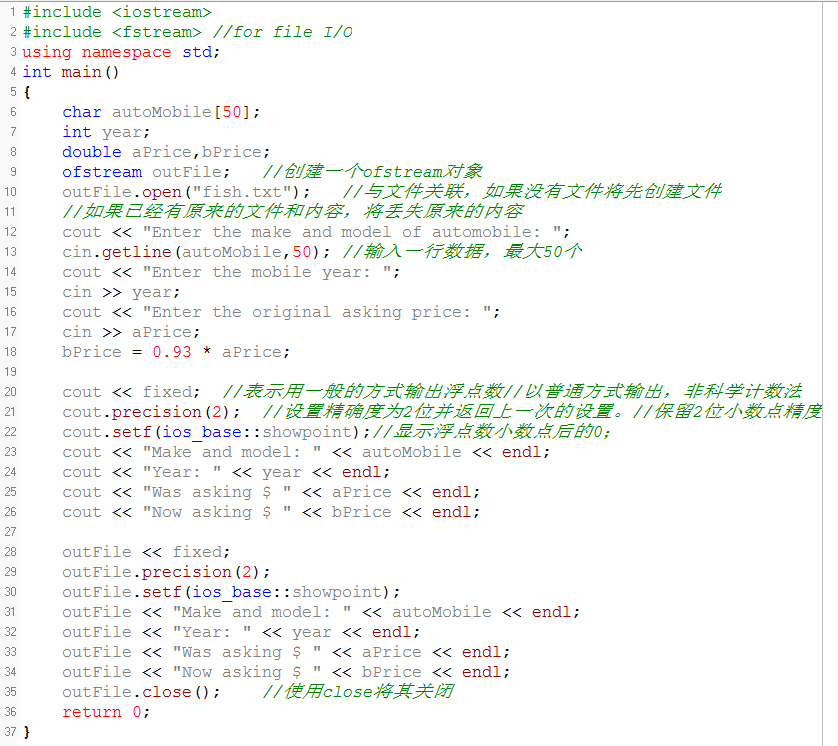


第五章 分支语句和逻辑运算符

读取数字的循环



简单的文件输入/输出



#include <iostream>

#include <fstream> //for file I/O

#include <cstdlib> //support for exit()

using namespace std;

int main()

{

char fileName[50];

ifstream inFile; //创建一个ifstream对象

cout << "Enter the name of datafile: ";

cin.getline(fileName,50);

inFile.open(fileName); //除非在输入的文件名中包含路径，否则程序将在可执行文件所属的文件夹中寻找

if(!inFile.is\_open()) //打开失败

{

cout << "Could nont open the file " << fileName << endl;

cout << "Program terminating.\n";

exit(EXIT\_FAILURE);

}

double value;

double sum = 0.0;

int count = 0;

inFile >> value;

while(inFile.good()) //输入正确并且不是EOF

{

++count;

sum += value;

inFile >> value; //获取下一个输入值

}

if(inFile.eof()) //读取文件的EOF,将返回true

{

cout << "End of file reached.\n";

}

else if(inFile.fail()) //如果读取的类型不匹配，方法fail()将返回true，如果遇到了EOF，该方法也将返回true

{

cout << "Input terminated by data mismatch.\n";

}

else

{

cout << "Input terminated for unkown reason.\n";

}

if(count == 0)

{

cout << "No data processed.\n";

}

else

{

cout << "Item read: " << count << endl;

cout << "Sum: " << sum << endl;

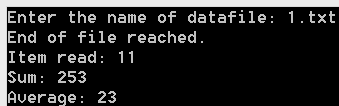
cout << "Average: " << sum/count << endl;

}

inFile.close();

return 0;

}

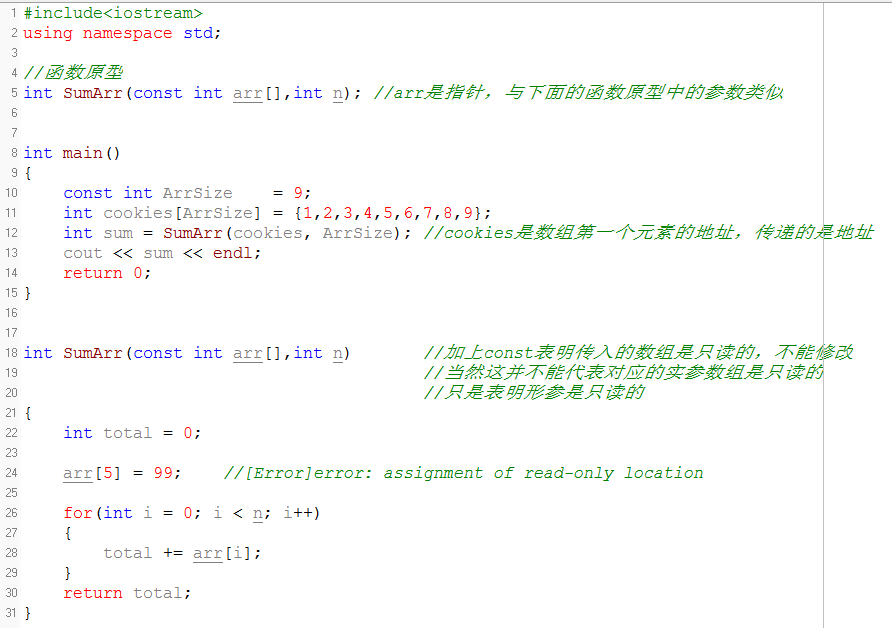


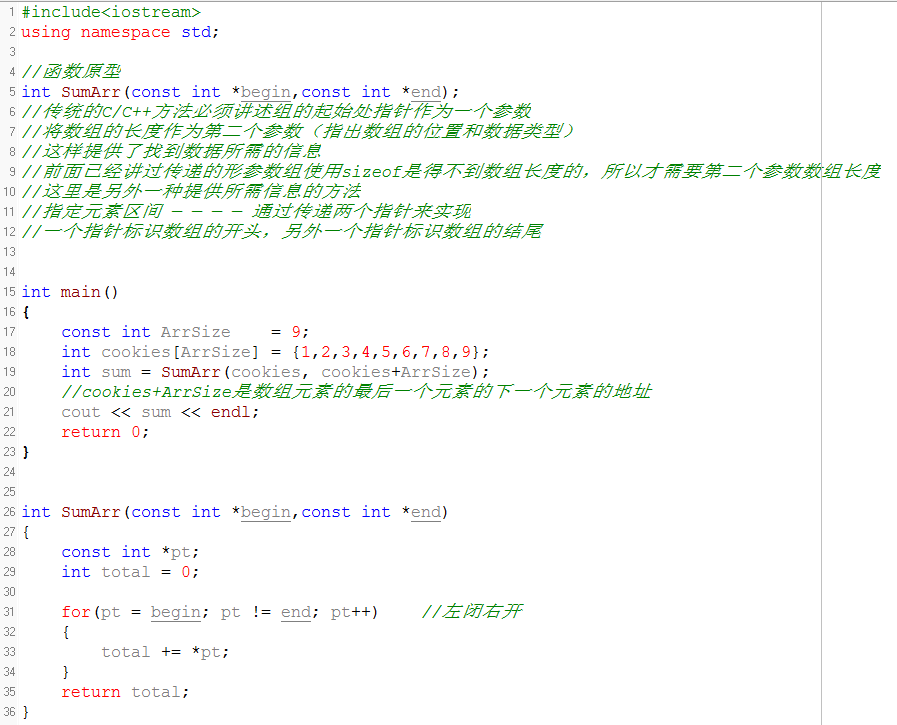
第六章 函数——C++的编程模块

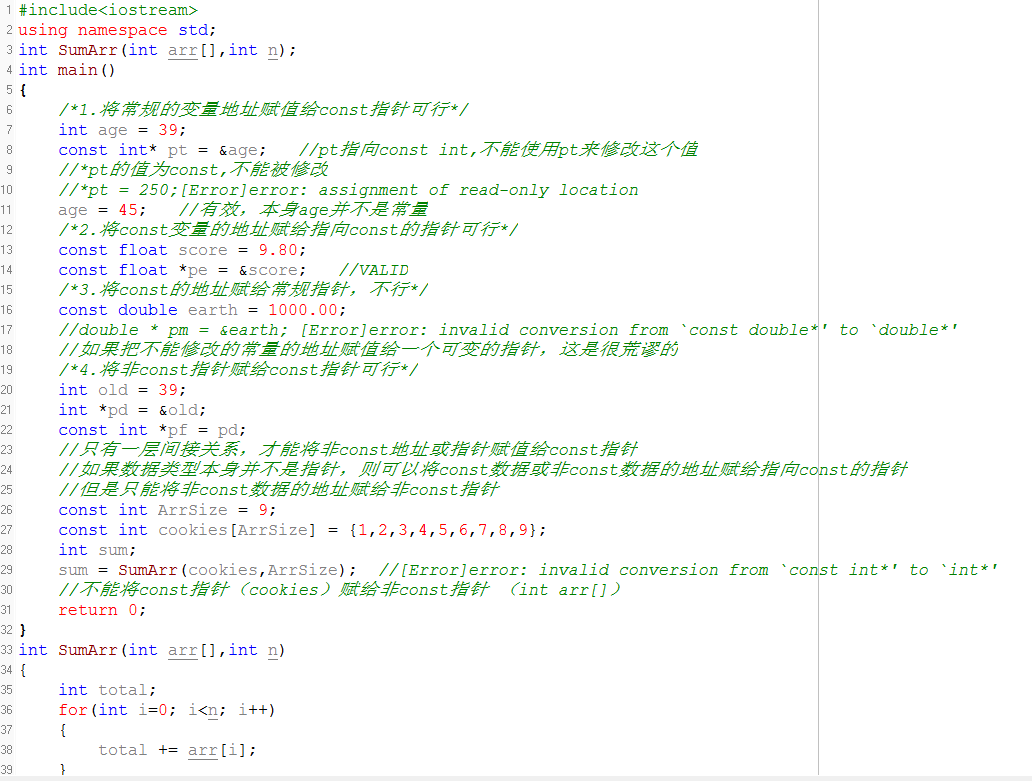
函数和数组

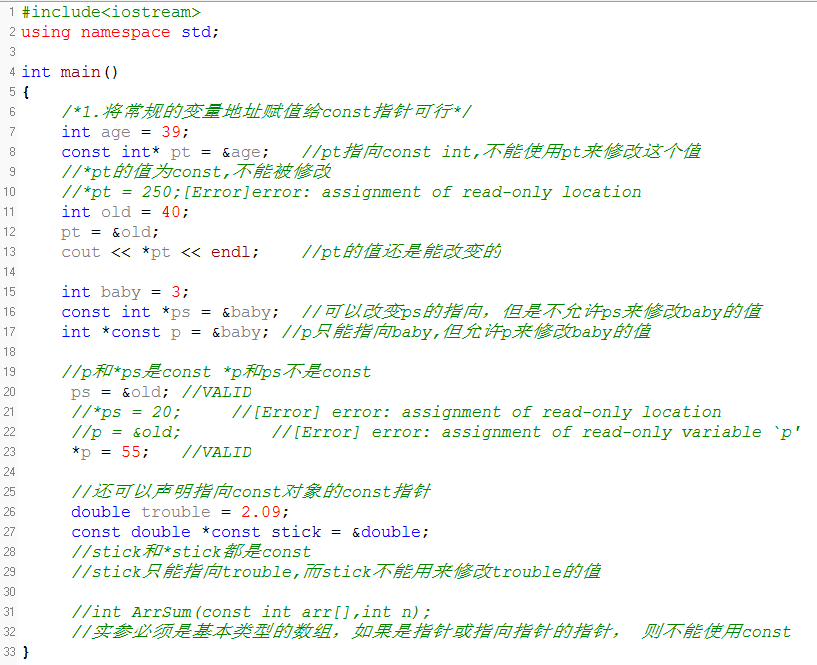






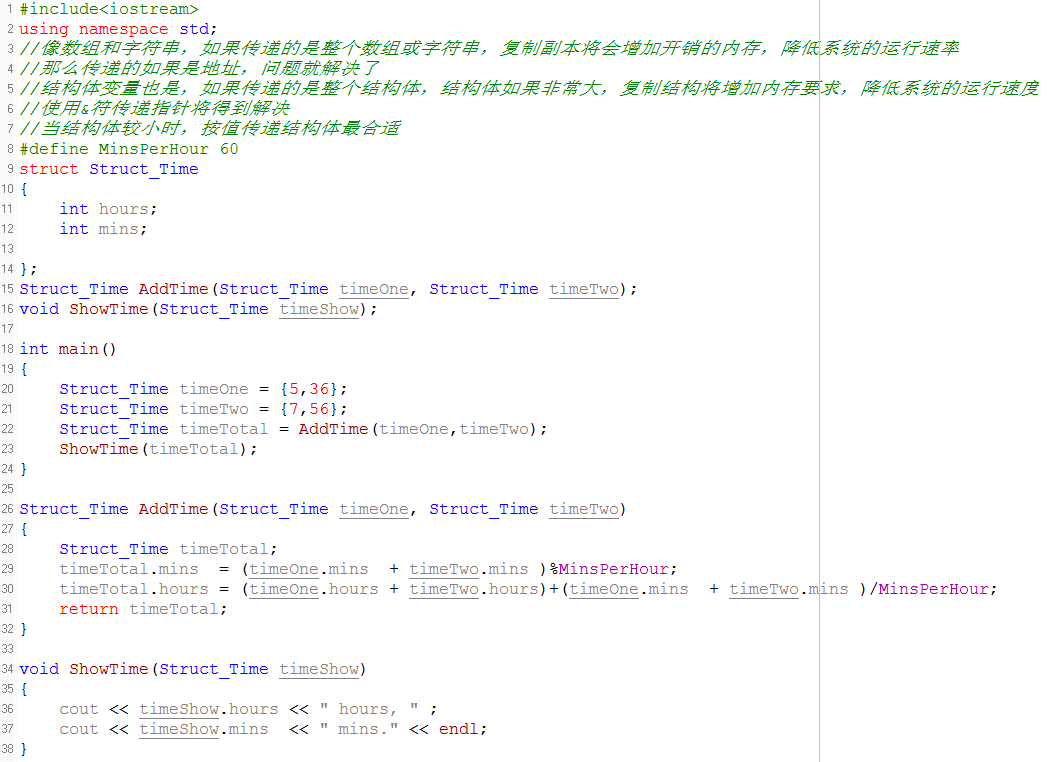


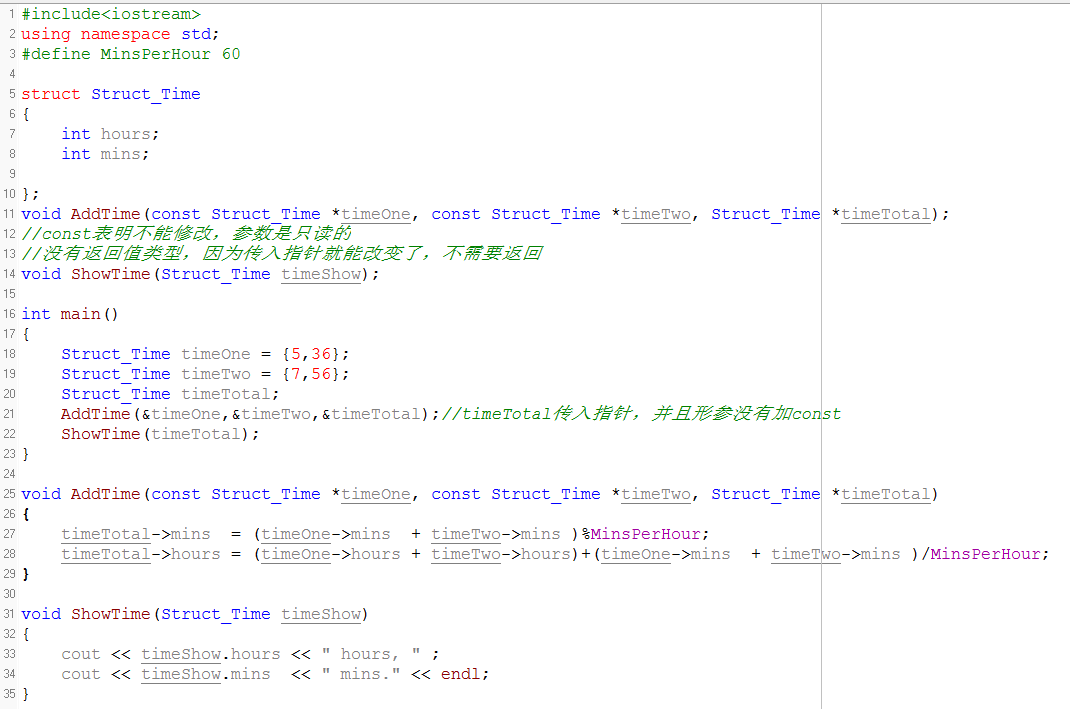




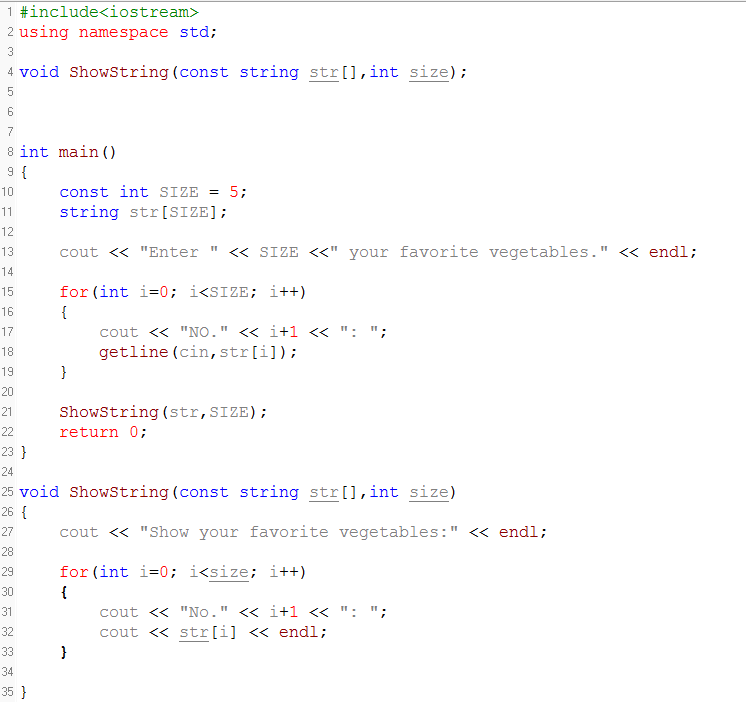


函数和结构

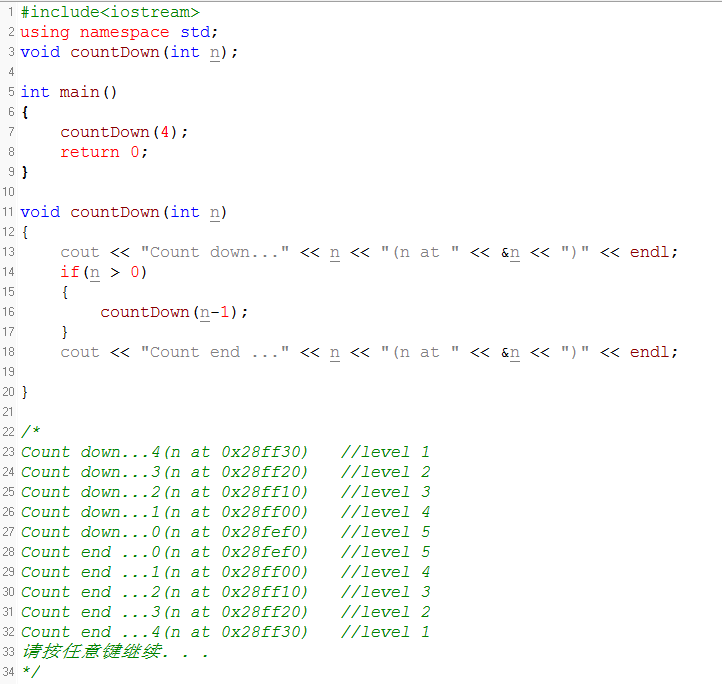




函数和string对象

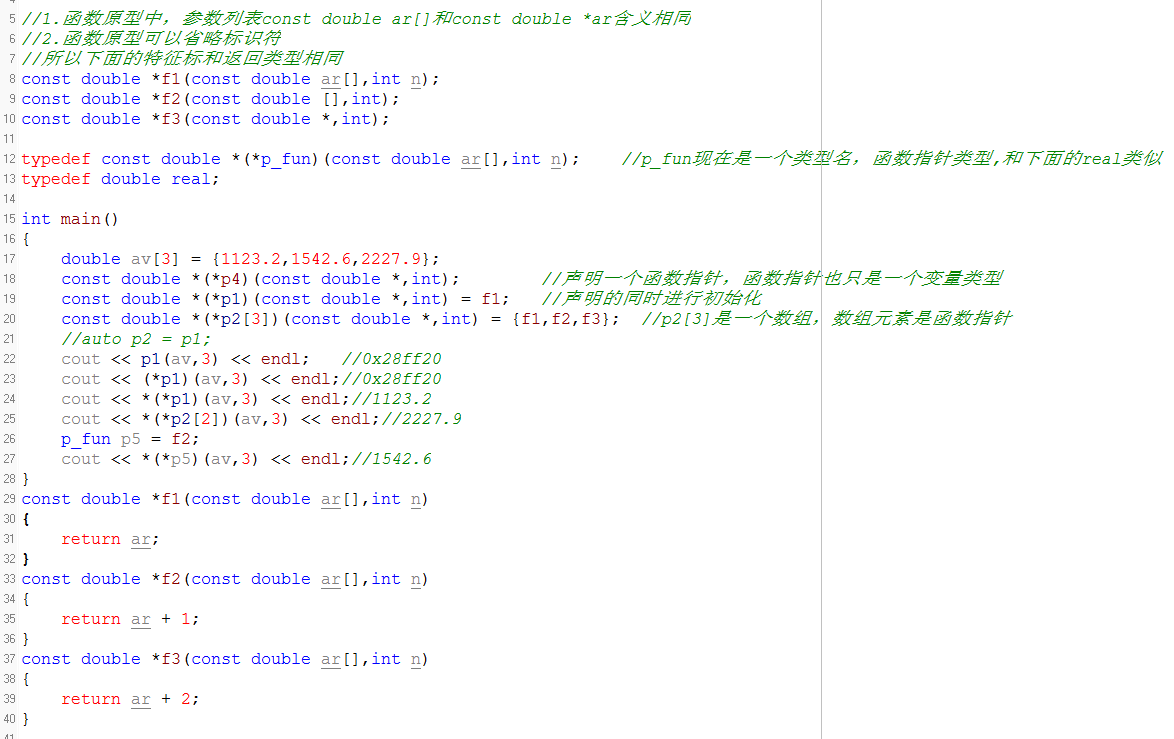


递归



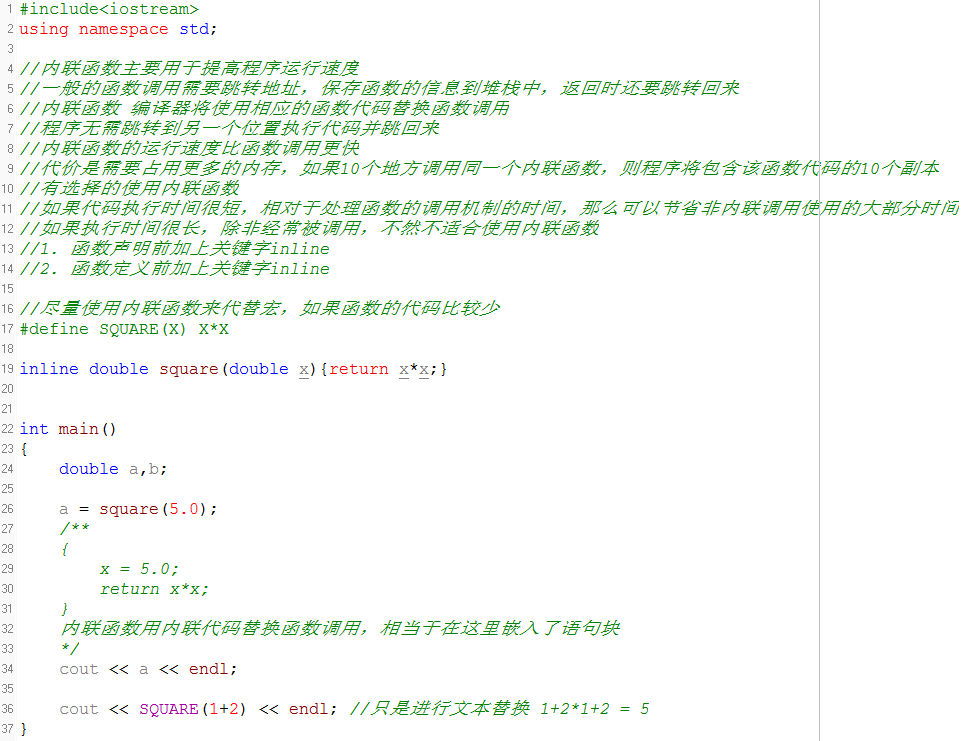
函数指针



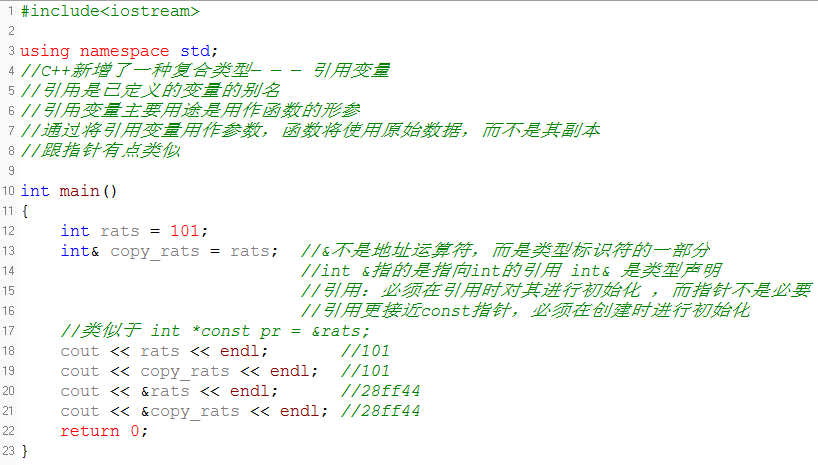


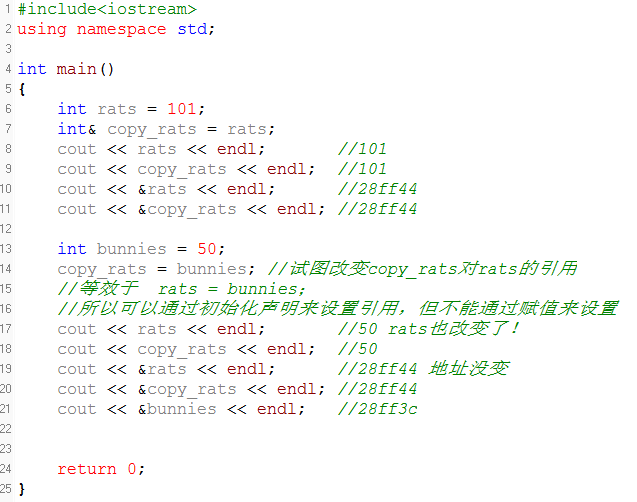
第七章 函数探幽

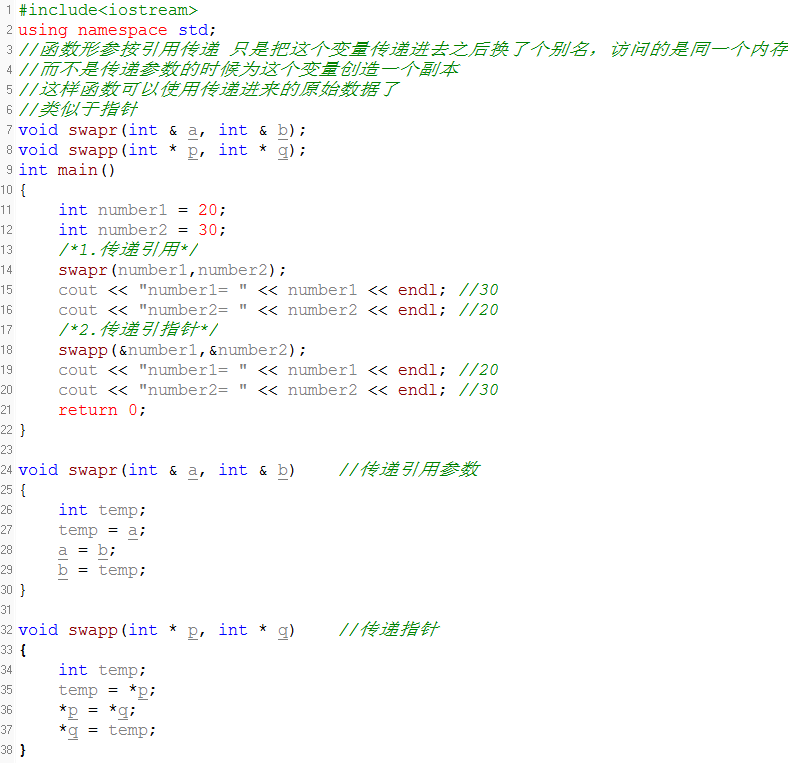
C++内联函数

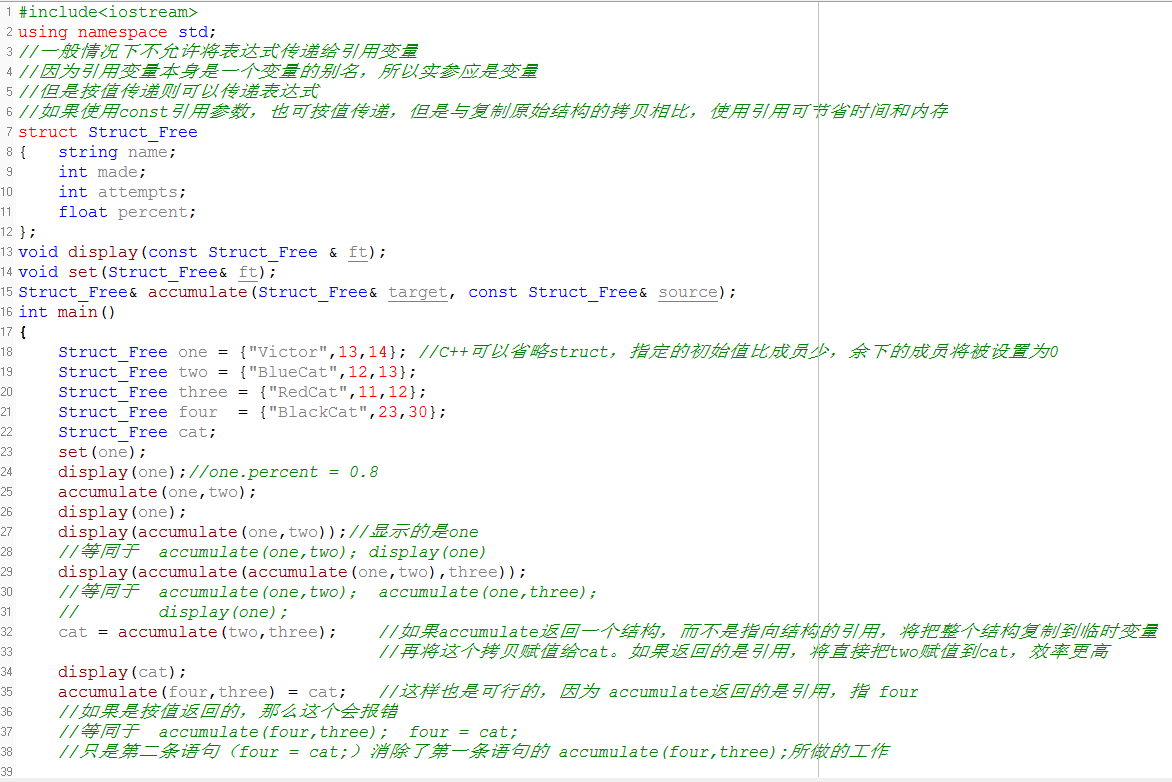


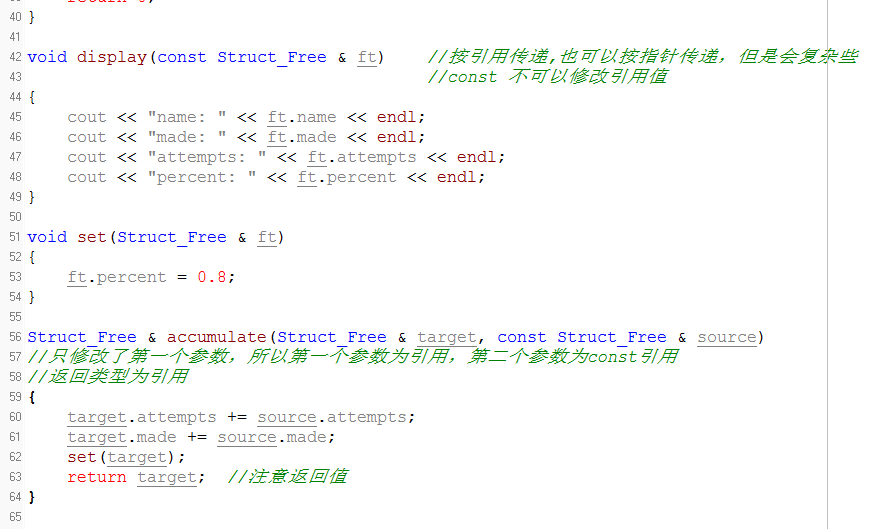
引用变量

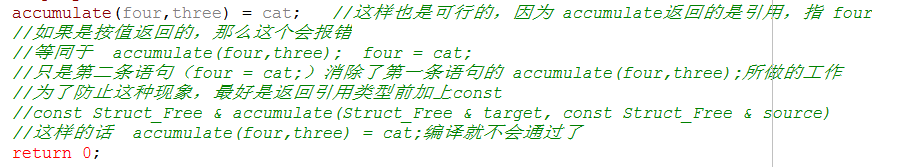












什么时候该使用引用参数，什么时候应使用指针，什么时候应使用按值传递？

一、如果参数不作修改

1.如果数据对象很小，如内置数据类型或小型结构，则按值传递

2.如果数据类型是数组，则使用指针， 这是唯一的选择，并将指针声明为指向const的指针

3.如果是较大的结构，则使用const指针或const引用，以提高程序效率，可以节省复制结构所需的时间和空间

4.如果数据对象是类对象，则使用const引用，传递类对象的标准方式是按引用传递

二、如果需要修改调用函数中的数据的函数

1. 如果数据对象很小，如内置数据类型或小型结构，则使用指针

2. 如果数据对象是数组，则只能使用指针

3. 如果数据对象是结构，则使用引用或指针

4. 如果数据对象是类对象，则使用引用

默认参数

