# 2.3 其他C++语句

程序清单2.3中的程序对前一个程序进行了扩展，要求在程序运行时输入一个值。它使用了cin，这是与cout对应的用于输入的对象。另外，该程序还演示了cout对象的多功能性。

**程序清单2.3 getinfo.cpp**

// getinfo.cpp -- input and output

#include <iostream>

int main()

{

 using namespace std;

 int carrots;

 cout << "How many carrots do you have?" << endl;

 cin >> carrots; // C++ input

 cout << "Here are two more. ";

 carrots = carrots + 2;

// the next line concatenates output

 cout << "Now you have " << carrots << " carrots." << endl;

 // cin.get();

 // cin.get();

 return 0;

}

在以前的程序清单中需要添加cin.get()，在这个程序清单中，需要添加两条cin.get()语句，这样才能在屏幕上看到输出。第一务cin.get()语句在输入数字并按Enter键时读取输入，而第二条cin.get()语句让程序暂停，直到您按Enter键。

下面是该程序的运行情况：

How many carrots do you have?

12

Here are two more. Now you have 14 carrots.

该程序包含两项新特性：用cin来读取键盘输入以及将四条输出语句组合成一条。

## 2.3.1使用cin

上面的输出表明，从键盘输入的值(12)最终被赋给变量carrots。下面就是执行这项功能的语句：

cin >> carrots;

从这条语句可知，信息从cin流向carrots。显然，对这一过程有更为正式的描述。就像C++将输出看作是流出程序的字符流一样，它也将输入看作是流入程序的字符流。iostream文件将cin定义为一个表示这种流的对象。输出时，<<运算符将字符串插入到输出流中；输入时，cin使用>>运算符从输入流中抽取字符。通常，需要在运算符右侧提供一个变景，以接收抽取的信息（符号<<和>>被选择用来指示信息流的方向）。

与cout一样，cin也是一个智能对象。它可以将通过键盘输入的一系列字符（即输入）转换为接收信息的变量能够接受的形式。在这个例子中，程序将carrots声明为一个整型变量，因此输入被转换为计算机用来存储整数的数字形式。

## 2.3.2使用cout进行拼接

getinfo.cpp中的另一项新特性是将4条输出语句合并为一条。iostream文件定义了<<运算符，以便可以像下面这样合并（拼接）输出：

cout << “Now you have “ << carrots << “carrots” << endl;

这样能够将字符串输出和整数输出合并为一条语句。得到的输出与下述代码生成的相似：

cout << "Now you have”;

cout << carrots;

cout << “ carrots";

cout << endl;

根据有关cout的建议，也可以按照这样的方式重写拼接版本，即将一条语句放在4行上：

cout << "Now you have”

<< carrots

 << “ carrots.”

 << endl;

这是由于C++的自由格式规则将标记间的换行符和空格看作是可相互替换的。当代码行很长，限制输出的显示风格时，最后一种技术很方便。

需要注意的另一点是：

Now you have 14 carrots.

和

Here are two more.

在同一行中。

这是因为前面指出过的，cout语句的输出紧跟在前一条cout语句的输出后面。即使两条cout语句之前有其他语句，情况也将如此。