**第五章**

C++提供了3种循环: for循环、while循环和do while循环。如果循环测试条件为true或非零，则循环将重复执行一组指令；如果测试条件为false或0则结束循环。for循环和while循环都是入口条件循环，这意味着程序将在执行循环体中的语句之前检查测试条件。do while循环是出口条件循环，这意味着其将在执行循环体中的语句之后检查条件。

每种循环的句法都要求循环体由一条语句组成。然而，这条语句可以是复合语句，也可以是语句块（由花括号括起的多条语句）。

关系表达式对两个值进行比较，常被用作循环测试条件。关系表达式是通过使用6种关系运算符之一构成的：<、<=、= =、>=、>或!=。关系表达式的结果为bool类型，值为true或false。

许多程序都逐字节地读取文本输入或文本文件，istream类提供了多种可完成这种工作的方法。如果ch是一个char变量，则下面的语句将输入中的下一个字符读入到ch中：

cin >> ch;

然而，它将忽略空格、换行符和制表符。下面的成员函数调用读取输入中的下一个字符（而不管该字符是什么）并将其存储到ch中：

cin.get (ch);

成员函数调用cin.get( )返回下一个输入字符一包括空格、换行符和制表符，因此，可以这样使用它：

ch = cin.get( );

cin.get (char)成员函数调用通过返回转换为false的bool值来指出已到达 EOF，而cin.get( )成员函数调用则通过返回BOF值来指出已到达EOF，EOF是在文件iostream中定义的。

嵌套循环是循环中的循环，适合用于处理二维数组。

**第六章**

使用引导程序选择不同操作的语句后，程序和编程将更有趣。C++提供了if语句、if else语句和switch语句来管理选项。if语句使程序有条件地执行语句或语句块，也就是说，如果满足特定的条件，程序将执行特定的语句或语句块。if else 语句程序选择执行两个语句或语句块之一。可以在这条语句后再加上if else，以提供一系列的选项。switch语句引导程序执行一系列选项之一。

C++还提供了帮助决策的运算符。第5章讨论了关系表达式，这种表达式对两个值进行比较。if和if else语句通常使用关系表达式作为测试条件。通过使用逻辑运算符（&&、||和!），可以组合或修改关系表达式创建更细致的测试。条件运算符（?:）提供了一种选择两个值之一的简洁方式。

cctype字符函数库提供了一组方便的、功能强大的工具，可用于分析字符输入。

对于文件I/O来说，循环和选择语句是很有用的工具；文件I/O与控制台I/O极其相似。声明ifstream和ofstream对象，并将它们同文件关联起来后，便可以像使用cin和cout那样使用这些对象。

使用循环和决策语句，便可以编写有趣的、智能的、功能强大的程序。

**第七章**

函数是C++的编程模块。要使用函数，必须提供定义和原型，并调用该函数。函数定义是实现函数功能的代码；函数原型描述了函数的接口：传递给函数的值的数目和种类以及函数的返回类型。函数调用使得程序将参数传递给函数，并执行函数的代码。

在默认情况下，C++函数按值传递参数。这意味着函数定义中的形参是新的变量，它们被初始化为函数调用所提供的值。因此，C++函数通过使用拷贝，保护了原始数据的完整性。

C++将数组名参数视为数组第一个元素的地址。从技术上讲，这仍然是按值传递的，因为指针是原始地址的拷贝，但函数将使用指针来访问原始数组的内容。当且仅当声明函数的形参时，下面两个声明才是等价的:

typeName arr[ ];

typeName \* arr;

这两个声明都表明，arr是指向typeName的指针，但在编写函数代码时，可以像使用数组名那样使用arr来访问元素：arr[i]。即使在传递指针时，也可以将形参声明为const指针，来保护原始数据的完整性由于传递数据的地址时，并不会传输有关数组长度的信息，因此通常将数组长度作为独立的参数来传递。另外，也可传递两个指针（其中一个指向数组开头，另一个指向数组末尾的下一个元素），以指定一个范围，就像STL使用的算法一样。

C++提供了3种表示C-风格字符串的方法：字符数组、字符串常量和字符串指针。它们的类型都是char\*（char指针），因此被作为char\*类型参数传递给函数。C++使用空值字符（\0）来结束字符串，因此字符函数检测空值字符来确定字符串的结尾。

C++还提供了string类，用于表示字符串。函数可以接受string对象作为参数以及将string对象作为返回值。string类的方法size( )可用于判断其存储的字符串的长度。

C++处理结构的方式与基本类型完全相同，这意味着可以按值传递结构，并将其用作函数返回类型。然而，如果结构非常大，则传递结构指针的效率将更高，同时函数能够使用原始数据。这些考虑因素也适用于类对象。

C++函数可以是递归的，也就是说，函数代码中可以包括对函数本身的调用。

C++函数名与函数地址的作用相同。通过将函数指针作为参数，可以传递要调用的函数的名称。

**第八章**

C++扩展了C语言的函数功能。通过将inline 关键字用于函数定义，并在首次调用该函数前提供其函数定义，可以使得 C++编译器将该函数视为内联函数。也就是说，编译器不是让程序跳到独立的代码段，以执行函数，而是用相应的代码替换函数调用。只有在函数很短时才能采用内联方式。

引用变量是一种伪装指针，它允许为变量创建别名（另一个名称）。引用变量主要被用作处理结构和类对象的函数的参数。通常，被声明为特定类型引用的标识符只能指向这种类型的数据：然而，如果一个类（如ofstream）是从另一个类（如ostream）派生出来的，则基类引用可以指向派生类对象。

C++原型让您能够定义参数的默认值。如果函数调用省略了相应的参数，则程序将使用默认值；如果函数调用提供了参数值，则程序将使用这个值（而不是默认值）。只能在参数列表中从右到左提供默认参数。因此，如果为某个参数提供了默认值，则必须为该参数右边所有的参数提供默认值。

函数的特征标是其参数列表。程序员可以定义两个同名函数，只要其特征标不同。这被称为函数多态或函数重载。通常，通过重载函数来为不同的数据类型提供相同的服务。

函数模板自动完成重载函数的过程。只需使用泛型和具体算法来定义函数，编译器将为程序中使用的特定参数类型生成正确的函数定义。

**第九章**

C++鼓励程序员在开发程序时使用多个文件。一种有效的组织策略是，使用头文件来定义用户类型，为操纵用户类型的函数提供函数原型，并将函数定义放在一个独立的源代码文件中。头文件和源代码文件起定义和实现了用户定义的类型及其使用方式。最后，将main( )和其他使用这些函数的函数放在第三个文件中。

C++的存储方案决定了变量保留在内存中的时间（储存持续性）以及程序的哪一部分可以访问它（作用域和链接性）。自动变量是在代码块（如函数体或函数体中的代码块）中定义的变量，仅当程序执行到包含定义的代码块时，它们才存在，并且可见。自动变量可以通过使用存储类型说明符register或根本不使用说明符来声明，没有使用说明符时，变量将默认为自动的。register说明符提示编译器，该变量的使用频率很高，但C++11摒弃了这种用法。

静态变量在整个程序执行期间都存在。对于在函数外面定义的变量，其所属文件中位于该变量的定义后面的所有函数都可以使用它（文件作用域），并可在程序的其他文件中使用（外部链接性）。另一个文件要使用这种变量，必须使用 extern关键字来声明它。对于文件间共享的变量，应在一个文件中包含其定义声明（无需使用 extern但如果同时进行初始化也可使用它）并在其他文件中包含引用声明（使用 extern且不初始化）。在函数的外面使用关键字static定义的变量的作用域为整个文件，但是不能用于其他文件（内部链接性）。在代码块中使用关键字static定义的变量被限制在该代码块内（局部作用域、无链接性），但在整个程序执行期间，它都一直存在并且保持原值。

在默认情况下，C++函数的链接性为外部，因此可在文件间共享；但使用关键字static限定的函数的链接性为内部的，被限制在定义它的文件中。

动态内存分配和释放是使用new和delete进行的，它使用自由存储区或堆来存储数据。调用new占用内存，而调用delete释放内存。程序使用指针来跟踪这些内存单元。

名称空间允许定义一个可在其中声明标识符的命名区域。这样做的目的是减少名称冲突，尤其当程序非常大，并使用多个厂商的代码时。可以通过使用作用域解析运算符、using声明或using编译指令，来使名称空间中的标识符可用。