国外经典教材·计算机科学与技术

PEARSON
Addison Wesley

Lab Manual to Accompany Problem Solving with C++: The Object of Programming

C++面向对象程序设计：上机指导

(美) Rahman Tashakkori 著
周 靖 译

清华大学出版社

国外经典教材·计算机科学与技术

C++面向对象程序设计

上 机 指 导

（美） Rahman Tashakkori 著

周 靖 译

清华大学出版社

北 京

内 容 简 介

作为经典教材《C++面向对象程序设计——基础、数据结构与编程思想》的上机指导，本书针对教材，设计了53个有意思的上机活动。通过这些上机活动，学生可更进一步掌握C++编程技巧。

EISBN: 0-321-17359-7

北京市版权局著作权合同登记号 图字：01-2004-3184

图书在版编目（CIP）数据

C++面向对象程序设计：上机指导/（美）塔萨科里（Tashakkori, R.）著；周靖译.—北京：清华大学出版社，2005.3
（国外经典教材·计算机科学与技术）
书名原文：Lab Manual to Accompany Problem Solving with C++: The Object of Programming, 4th Edition
ISBN 7-302-10570-7
Ⅰ.C… Ⅱ.①塔… ②周… Ⅲ.C语言—程序设计 Ⅳ.TP312

中国版本图书馆CIP数据核字（2005）第014345号

出 版 者：清华大学出版社 **地 址**：北京清华大学学研大厦
http://www.tup.com.cn **邮 编**：100084
社 总 机：010-62770175 **客户服务**：010-62776969
文稿编辑：文开棋
封面设计：久久度文化
印 刷 者：北京市世界知识印刷厂
装 订 者：北京鑫海金澳胶印有限公司
发 行 者：新华书店总店北京发行所
开 本：185×260 **印张**：13.25 **字数**：314千字
版 次：2005年3月第1版 2005年3月第1次印刷
书 号：ISBN 7-302-10570-7/TP·7168
印 数：1～4000
定 价：24.00元

出 版 前 言

在《C++面向对象程序设计：上机指导》出版之际，简要介绍一下这本书及其配套教材的情况。

这本上机指导是国外经典教材《C++面向对象程序设计——基础、数据结构与编程思想》（ISBN 号为 7-302-07594-8）的配套用书。后者原书名为 *Problem Solving with C++: The Object of Programming*，作者是美国加州大学教授 Walter Savitch。Savitch 教授是是计算机课程的主要设计者之一。他是一名享有盛名并受人尊敬的教育专家和作者。很多教师和学生非常喜欢他的授课和写作风格。他编写过多部非常畅销的计算机类教材。

《C++面向对象程序设计——基础、数据结构与编程思想》便是其中之一。原著首次出版于 1995 年，距今已经有十年，在世界各地都拥有大批读者。在我国台湾地区，国立东华大学资讯工程学系，中原大学资讯工程系和国立交通大学应用数学系均采用本书作为教材。在这十年当中，Savitch 教授一直关注着编程领域的发展，并陆续更新和修订这本教材（现在已经有第 5 版了），因而这本教材始终有大批忠实的读者。

有读者如此评价：

“作为初学编程的大一学生，我必须要清楚而快速地理解基本的编程概念。老师分配的作业需要很快提交。事实证明，Walter Savitch 编写的这本书帮助我迅速而高效地掌握了基本概念以及编程组件与参加课堂训练的关系。作为入门书，本书针对没有任何编程背景的读者，提供了基本的、严谨的理论知识，把读者引入 C++动态编程环境中，并用易于理解的图示解释了源代码执行期间发生了什么，并说明为什么以及何时会发生这些行为。全书覆盖面广（同时适合作教材和自学），其中附有答案的自测题尤其受欢迎。程序高手（上帝保佑他们）也许认为这本书很浅，对它不屑一顾，但说实在的，遍布书中各处的“编程提示”将伴随着我们的整个编程生涯。”

也有读者这样说：

“我在卡梅隆大学主修计算机和数学。幸运的是，我们在大一时选用了本书作为教材。在我看来，本书是最棒的 C++教材，没有哪本书（Dietel 编著的 C++大学教材除外）能像她这样清楚地解释 C++基础知识。对刚接触编程的新手而言，有些较难的主题，比如类、指针和虚函数，都因为良好的范例变得‘浅显易懂’。书中采用的范例比其他任何书都简单易懂，但同样清楚地说明了主题。

本书的编排顺序可能不太好。数组出人意料地放在第 10 章进行介绍，而类放在第 6 章介绍，这与其他 C++教材显然不同。这可能有点脱离传统，但先介绍类能让我们直接进入 C++编程。尽管我们的老师是按顺序介绍各章的，但书中的材料非常灵活，可随意调整（但有人建议最好采用‘前言’部分推荐的顺序）。”

清华大学出版社在 2003 年引进了这本教材的第 4 版。在甄选译者的过程中，我们选择了具有多年技术背景和翻译经验的周靖。在翻译过程中，他将原书勘误结合在其中，并把原作者疏漏的个别问题进行了补充。与此同时，还考虑到初学者的需要，将所有注释转换为中文，并在适当的地方添加译者注，以帮助读者迅速消化 C++基础知识。此外，他还对所有代码进行了编译，并在个别地方提供他自己的修改版本。译者严谨、认真的态度，使我们的《C++面向对象程序设计》上升到一个新的台阶。

这本教材出版以来，陆续收到读者来信，或者希望获得编程项目题的答案，或者希望解决编程过程中碰到的问题，甚至有读者问是否能帮他们找到配套的上机指导。经过一些努力，在这个春暖花开的日子里，终于迎来了上机指导的出版。

上机指导是阿巴拉契亚州立大学 Rahman Tashakkori 教授为这本教材设计的。所有上机活动都经过精心设计，有助于读者开发自己的编程和解题技能。作者治学严谨，要求学生参加所有上机活动，并重点强调“缺 3 次上机活动以上的学生，将被评为不及格”。

在这些上机活动中，课前准备部分必须先于后续活动完成。其中都有一个练习。其目的是帮助学生更好地理解上机活动，更好地利用动手机会。

最后，希望本上机指导能帮助广大的学生从实践中领会编程的精髓。选用《C++面向对象程序设计》（第 4 版）及其上机指导的教师，则可发邮件索取上机指导的电子文稿，以便更好地安排上机活动，邮件地址 coo@netease.com。

清华大学出版社

2005 年春

目　录

第 1 章　计算机和 C++编程入门

姓名：　　课程编号/Section：

登录 ID：　　日期：

目标：

- 了解计算机系统基础知识
- 了解问题求解和编程
- 了解软件生存期
- 问题求解和编程——一个简单的 C++程序

指导内容	分数	需要完成	成绩
课前准备		在上机之前完成	
1-1　简介		练习 1.1	
1-2　程序设计过程			
1-3　一个简单的 C++程序		练习 1.2 练习 1.3 练习 1.4	
课后练习			
总分		总成绩	
学生评语：	教师评语：		

感谢 Appalachian State University 计算机科学系学生 Monica Verma 和 Scott A. Barlowe 协助测试和完善这次上机活动。

课前准备

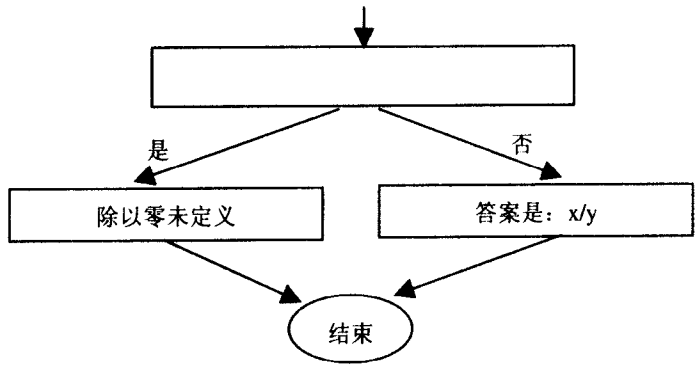
开始上机之前，先阅读教材第 1 章，并回答以下问题：

1）您为这门课程使用的是什么操作系统？

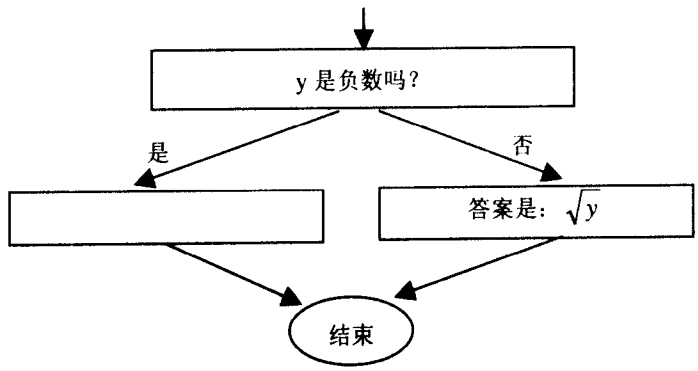
2）教师推荐为这门课程使用什么文本编辑器？

3）创建一个程序需要哪些步骤？

4）在下图中，在空白方框内填写一个问句，使其符合之后的陈述句。假设在空白方框之前，您已经读取了 x 和 y 的值。



5）在下图中，在空白方框内填写一个陈述句，使其符合之前的问句。



6）问题求解和编程相比，哪一个更难？请详细说明。

7）为解决一个难题，您准备如何与计算机沟通？

1-1　简　介

在这门课中，您将学习写 C++程序来解决不同的问题。有些时候，这个任务可能会让人迷惑不解，让人犯难。但是，只要您有耐心，并不断练习，就能获得更多经验，学起来会越来越轻松。用计算机来解决问题之所以困难，是因为我们需要指挥计算机为自己干活儿。想像一下，我们坐在计算机面前，要求它对两个数进行相加："嘿，2 和 4 相加，等于多少？"但就目前而言，我们还不能如此直接与计算机交谈。只能通过一个操作系统（OS）来和它交流。和操作系统的交流是通过您写的一个程序来完成的。所谓程序，是计算机将要执行的一组指令。这些指令不是用日常语言写的，但我们能很好地理解它们，因为它们是用高级语言来写的。机器语言是低级语言。本章末尾的补充内容对此进行了更详细的说明。

一般说来，在得到程序输出之前，要经历若干个步骤：1）首先要用一个编辑器来敲入程序；2）其次用一个编译器（这里用的是 C++编译器）来编译程序，并得到一个可执行文件；3）最后运行这个可执行文件，得到输出。

程序员的工作中，最重要的部分首先是解决问题。相较于把自己的解题思路变成程序，问题求解过程要难得多。因此，一个人应该首先思考一个方法，并开发一个算法来解决问题。算法是一系列准确的指令，它们最终形成一个解决方案。算法的关键在于"准确"和"清晰"。算法有歧义，就得不到正确答案。

练习 1.1

假设您要帮助学校注册处处理学生注册过程。您的任务是把学生们送到 6 个不同的大厅（Hall），具体送到哪一个大厅要取决于学生姓氏（Last Name）首字母及其手上的缴费单上的余额。区分学生的条件如下所示：

学生余额为 0，首字母：

A-E 去大厅 3，F-J 去 2 号大厅，L-O 去 18 号大厅，P-R 去 10 号大厅，S-Z 去 12 号大厅。

学生余额尚不为 0 的学生去 18 号大厅缴费。一旦这些学生的缴费单上的余额为 0，就可以回去注册。

通过文字陈述或者画示意图的方式，描述您的算法为解决这个问题而采取的步骤。本章末尾的补充内容给出了几个提示（参见提示 1，提示 2 和提示 3）。

1-2　程序设计过程

解决问题一般分为两大阶段：

1）问题求解阶段；2）实现阶段。

阶段(1)——在第一个阶段，您要采取 3 个步骤：

步骤 1：清楚定义出要解决的问题。

步骤 2：设计一个算法，这个算法要准确，要经过深思熟虑。

步骤 3：在纸上演算您的算法。只有在确定算法能正确工作之后，才能着手为它写一个程序。

阶段(2)——在这个阶段，要采取 2 个步骤：

步骤 1：把算法转换成 C++语言。如果您已经有了一个正确和准确（无歧义）的算法，那么几乎能够逐行进行翻译（算法的一行对应于程序的一行）。转换必须正确，而且没有以下错误：

- 语法错误，也就是因为不正确使用编程语言的语法或违反语法规则而造成的错误。
- 不可能实现的计算，比如除以零。
- 程序员造成的错误，比如错用符号或算术操作符而造成的错误。

步骤 2：测试程序，确定它能得出正确结果。确定您使用了多种不同的测试条件。正确测试程序的惟一方法就是使用多种不同的测试条件。

准备写一个简单的 C++程序

在编写、编译和运行第一个程序之前，需要了解几个小问题。

问题 1：操作系统（OS）

C++编译器通常是在一个操作系统上运行的。在本书的上机活动中，我们假定您使用的是 Unix 操作系统或者 Unix 风格的操作系统。本书包含一个简短的 Unix 命令手册，为您解释了数量有限的一系列命令。要了解更多的信息，建议您参考 Unix 图书。另外，也可以阅读网上的 Unix man page，方法是在命令行上敲入 *man <命令名称>*。

问题 2：编辑器

一旦设计好算法并在纸上完成了演算，就需要用一个文本编辑器敲入程序。有好几个编辑器都可用于敲入程序。最常用的文本编辑器是 vi，pico 和 nedit。您的导师可能要求您使用这三个编辑器中的一个，也可能要求您用其他编辑器。我们提供了两个简单的手册，一个用于 vi，另一个用于 pico，帮助您掌握录入程序的基本技术。如果不知道如何使用编辑器，建议您先阅读我们提供的简单手册或者由老师为您提供的手册，以便熟悉自己即将使用的编辑器。

问题 3：编译器

取决于您所使用的计算机类型，所用的 C++编译器也可能不同。GNU C++编译器是一个免费的编译器，很多人都在用它。为了编译 C++代码，GNU C++编译器会要求您敲入 g++ <程序名>。如果您的机器上的命令与此不同，就把它更换为适用于您的系统的命令。

问题 4：电子邮件

在整个学期，您需要和老师、同学以及 / 或者助教沟通。Unix 提供了几个电子邮件包。请和老师确定一下要为这门课程使用哪一个包。

现在，我们将开始编写第一个程序。

1-3　一个简单的 C++程序

一家食品杂货店每天都售出许多箱软饮料。每箱 12 瓶，店主每售出一瓶，就获利 20 美分。我们想算出这家店每天出售软饮料的获利。另外还想算出一年出售软饮料的获利。假设一年有 365 天。

1）问题定义：

计算一家商店一天出售软饮料所获的利润。

计算一家商店一年出售软饮料所获的利润。

这个程序涉及比较多的计算，所以我们要写一个 C++程序，在计算机上解决这个问题！至少现在如此。

2）问题设计－算法

在开始写程序之前，首先要开发一个解决该问题的算法。

练习 1.2

为这个问题设计一个算法。您既可以在纸上画出示意图，也可以直接写出步骤。记住，您的算法必须准确，不能存在歧义。如果愿意，您可以自己设计一下，然后将您的设计和我们在后面给出的算法进行比较。算法必须转换成 C++，以便获得实际的程序。

2-A) 桌面测试

得到算法后，要在纸上对其进行演算，检验它的正确性。

3）用 C++实现算法

这个步骤要将算法转换成 C++。下面这个程序是根据前面获得的算法来设计的。请检查这个程序，确定其中的每个步骤已得到正确转换。

//P11.cpp－这个 C++程序将计算通过出售软饮料而获得的利润

```
#include <iostream>
using namespace std;

int main()
{
    int cases_per_day, bottles_per_day;
    int bottles_per_case = 12;
    double profit_per_bottle = 0.2;  // 20 cents per bottle profit
    double profit_per_day, profit_per_year;

   cout << "Press enter after entering each number \n";
   cout << "Enter number of cases \n";
   cin >> cases_per_day;

   profit_per_day = cases_per_day * bottles_per_case * profit_per_bottle;
   profit_per_year = 365 * profit_per_day;

   cout << "The store has a made :";
   cout << profit_per_day;
   cout << "per day. \n";
   cout << "That means the profit for one year will be:";
```

```
    cout << profit_per_year << endl;

    cout << "Good business?! \n";
    return 0;
}
```

练习 1.3

在您为本课程准备的目录下面新建一个 lab1 目录。切换到这个目录。打开一个空白文件 P11.cpp，采取剪切并粘贴的方式，或者将上述程序仔细地录入文件，保存文件后退出。注意在某些计算机上，C++编译器不会编译扩展名为.cpp 的 C++程序，您可能要将文件另存为 P11.C。如果要求用.C 扩展名来保存 C++程序，那么在后续所有上机活动中，都要用.C 来代替.cpp 扩展名。就目前来说，我们假设您用.cpp 扩展名来保存自己的文件。

敲入以下命令编译这个程序：

```
%g++ P11.cpp（如果使用.C 扩展名，则敲入%g++ P11.C）
```

如果您写的程序没有语法错误，就会得到一个新的文件，名为 a.out。要想运行这个程序，敲入下面的命令即可：

```
%a.out
```

4）测试程序

最后要做的事情是测试程序，确定它能产生正确的结果。

假设这家商店每天售出 10 箱软饮料，那么每天能获得多少利润？每年呢？您可以手工算出答案，再和程序结果进行对照。

练习 1.4

修改程序 P11.cpp，使其在计算时，使用每瓶饮料获利 22 美分。在程序输出中，这一次要同时显示一天、一年以及十年售出的瓶数以及相应的利润。将新的程序命名为 P12.cpp。

详细解释简单的 C++程序 P11.cpp

程序第 1 行是：

```
//P11.cpp－这个 C++程序将计算通过出售软饮料而获得的利润
```

双斜杠//向编译器表明这一行是注释，不参与计算。添加注释的目的是为了增强程序的可读性，并且 / 或者对程序的各个部分进行描述。

下一行是：

```
#include <iostream>
```

这一行称为“include 预编译指令”。它向编译器指出，应该到哪里去寻找和程序中使用的部分项目有关的信息。在前面的程序中，这些项目的例子包括 cout，<<，>>和 cin。您需要使用其他库来包容其他项目。注意，预编译指令肯定以符号#开头。

```
using namespace std;
```

这行代码向编译器表明要对 iostream 中定义的名称进行解释，而且要采用标准（std）

方式来解释。注意，没有这行代码，程序也能运行，因为默认设置是使用一个全局命名空间。我们将在“第 10 章”中详细讨论命名空间。

```
int main()
{
```

我们可以简单地认为，上述代码表示“程序从此开始”。实际上，int 代表 integer（整数）类型，main 是一个函数名，而括号()代表参数的边界。我们很快就会对它们进行详细解释。注意，花括号{表示 main 函数从这里开始，而花括号}则表示 main 函数在这里结束。通常，每个{都有一个对应的}。

接下来的 4 行代码是：

```
int cases_per_day, bottles_per_day;
int bottles_per_case = 12;
double profit_per_bottle = 0.2;  //每瓶饮料的利润是 20 美分
double profit_per_day, profit_per_year;
```

在上述代码中，我们声明了要使用的变量，还定义了它们的类型。我们将在以后的活动中接触不同的变量类型。在这 4 行代码中，int 用于声明“整数”类型的变量。在第 2 行，我们在将 bottles_per_case 定义为一个整数的同时，将它初始化为 12。在第 3 行和第 4 行，我们定义了 double 类型的变量。注意，每个语句都以一个分号（;）结束。

接下来的两行是：

```
cout << "Press enter after entering each number \n";
cout << "Enter number of cases \n";
```

我们使用 cout 在屏幕上显示一条消息。cout 语句允许我们将数据从一个变量中定向到屏幕上。

另一方面，在下面这行语句中，cin 将数据从键盘定向到一个变量中：

```
cin >> cases_per_day;
```

记住<<和>>的指向非常重要，同时也非常容易。注意，将数据发送到屏幕时，我们是将它们发送到 cout，所以是 cout <<；使用 cin 语句将数据从键盘发送给变量时，由于 cin 代表键盘，所以是 cin >>。

在示范程序中，我们执行了一些计算：

```
profit_per_day = cases_per_day * bottles_per_case * profit_per_bottle;
profit_per_year = 365 * profit_per_day;
```

第 1 行执行了乘法（*），也就是“cases_per_day 乘以 bottles_per_case 乘以 profit_per_bottle”。另外，我们使用赋值操作符（=），将上述乘法结果保存到 profit_per_day 中。在后面的上机活动中，我们将学到多种不同的算术操作符。请自己解释一下第 2 行做了什么。

祝贺您，通过这个上机活动，您学会了用计算机程序来解决问题的基本步骤。另外，还学习了 C++的一些语法。在最后一部分，我们将简要讨论写程序时可能碰到的一些错误类型。

各种程序错误

用计算机程序解决问题时，可能遇到以下 3 种类型的错误：1）语法错误；2）逻辑错误；3）运行时错误。

1）语法错误

语法错误是违反了编程语言的语法规则而造成的。例如，如果忘记在每一个 C++指令的末尾添加分号（;），编译器就不能正确编译，并在屏幕上报错。为了查看错误类型，您可以试着删除一个分号，再试着编译程序。这类错误的另一个例子是没有使用匹配的花括号｛和｝。

2）逻辑错误

逻辑错误是在您写程序时，错误地转换了算法而造成的。这种错误计算机无法检测。要找出这种错误，惟一的办法就是在写完程序后仔细测试它。例如下面这个语句：

```
profit_per_day = cases_per_day * bottles_per_case * profit_per_bottle; //正确
```

计算 profit_per_day 时，上述语句是正确的。但是，如果您不小心用+代替了*，得到的语句就是：

```
profit_per_day = cases_per_day + bottles_per_case * profit_per_bottle; //错误
```

虽然仍然能得到一个答案，但那个答案是错误的。这个错误是在将算法转换成 C++语句时发生的。在应该用*的地方，我们错用了+。

3）运行时错误

运行时错误是在运行程序的时候侦测到的。这种类型的错误大多与数字运算有关。例如，计算机是无法算出一个负数的平方根的。

课后练习

1. 写一个 C++程序，使其能生成如下所示的输出：

```
My name is YourFirstName YourLastName

        Hello
    Hello   Hello
Hello   Hello   Hello
    Hello   Hello
        Hello
Bye
```

2. 将以下程序复制并粘贴到一个文件中，将文件另存为 PS2.cpp，然后编译程序。屏幕上会报告一些语法错误，请注意错误的类型，以及出现错误的行号。逐一改正这些错误，然后重新编译程序，直到程序中没有错误为止。运行这个程序并确保它能生成一个输出。

```
include<iostream.h>

int main( )
}
   cout < "This is the second program of the post_lab. \n";
   cout << "There were some syntax errors in it that I fix them. \n';
   count >> "Syntax errors are due to the violation of the grammar of C++ \n"

    return0;
}
```

第 1 章—补充内容

计算机是如何解决问题的?

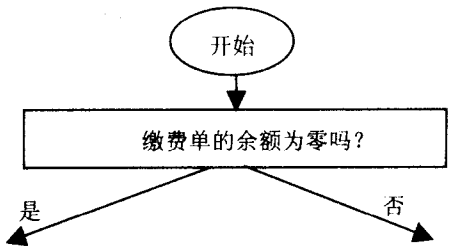
通过下面的描述，您将更容易理解计算机是如何解决问题的。假设您是一名建筑师，知道如何设计房屋，刚刚到日本工作，打算在这里建房子。您不懂日语，但可以找到一个翻译。于是，在一张纸上用英语写下一组指令，并把它交给翻译。第二天，翻译会将这组指令翻译成日语，并将它交给您。如果翻译能完全理解您所说的话，那么一切都会十分顺利。如果有的地方是翻译不能理解的（**语法错误**），他就会中断他的工作，让您知道有些错误必须改正，然后才能得到正确的翻译结果。错误可能是由一组语法不正确的指令造成，或应归咎于缺了某些语法成分。如果没有错误，很快就能得到正确的翻译结果。在看到这组翻译过来的指令时，您看不懂，这也难怪，它毕竟是用另一种语言写的。但不要着急，有人看得懂这些指令，并为您建好房子。下一步是找到一个日本建筑师，请他建房子。幸运的话（这里我们假设您是幸运的），翻译和建筑师恰好是同一个人，而您只需要拿到翻译好的设计指令，然后交给他执行即可。随后，日本建筑师将准备好所有建筑材料，并开始建房子。如果您所给的所有尺寸都是正确的，而且所有要求都是合理的，就能得到一所好房子。有时，您可能会提出一个不可能实现的请求。不合理的请求不一定就是语法错误造成的。例如，您要求承建方建一个高度为 0 的狗舍。尽管这样的尺寸在纸面上是合理的，但实际是不可执行的。换言之，您要求承建方建造一个不可能的东西。在这种情况下，他会在执行指令集的过程中指出这个错误（**运行时错误**）。第三种错误是在您设计算法或者写下指令集时造成的。例如，您写出来的指令可能是“keep the distance between each steps at 1 foot”，但事实上您的想法是“keep the distance between each steps at 1/2 foot”。在这种情况下，承建方会根据指令中指定的距离来设置楼梯的台阶。这类错误称为**逻辑错误**。在日常生活中，您看到过多少这类错误呢？开门的方向错了，楼梯的台阶太宽或太窄了。

我们可以将这个故事的主体和元素与程序设计课程中的元素对应起来：

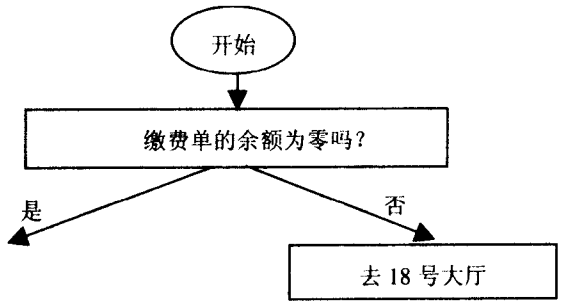
您知道如何设计房子	您知道如何编写 C++程序—语法
您写下的指令集	您的 C++程序
日语译员	C++编译器
您的指令集的翻译版本	编译器生成的可执行文件
日本建筑师	C++编译器
资源	链接器（Linker）、输入数据等等
房子	程序输出结果

除此之外，您也许还能找到其他相似之处。

提示 1



提示 2



提示 3