小鸡和白兔教你明白编程

王士弘
pwang@cs.kent.edu

2022 年 10 月 1 日

计算机给我们带来了信息时代和数字世界。计算机有一种万能的性质。只要有程序它就可以做任何事。因此它对世界及其文明的影响力比任何其他机器都要大得多。

即然如此，我们对计算机编程的了解就是计算思维(CT) 中不可避免的重要部分。在这里我们用一个简单易懂的例子来揭开编程神秘的面纱。

对编程的了解也可启发我们在不同领域里解决问题和寻找答案的方法。甚至在日常生活中都能派上用场。本文针对一般读者，简单易懂，内容丰富有趣。

程序并不稀奇

其实我们对程序并不陌生。任何按步就班有组织有次序的方法就是程序。比如电视和广播节目单, 办公手续, 炒菜做饭的食谱, 等等。其实你可能也写过几个食谱, 药方, 旅游线路之类的东西。要写得清晰易懂不出差错才好。计算机编程也一样。

想不想尝试编程？别害怕，不难的。一看就会。

编程思维

编程就是要把完成一个任务的具体方法和步骤一步接一步写下来。要完整有序不能有漏洞。

编写程序时我们必须小心做到：要让**任何一个人** 都可以准确地理解和执行每个步骤。一个好的程序应该是**万无一失的**。因为程序最终是要由顽固死板的机器（计算机）来执行的。

这就要求程序员采用与外行人不同的思维方式。程序员要确保一切都非常清楚，且没任何出差错的机会或可能。因此，一个程序是一个接一个事先预定的步骤。而且为路上的每一个岔道，都必须提前制定因应的步骤。

计算机程序的定义

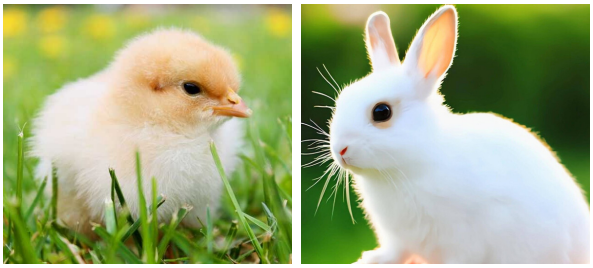
计算机程序是一个可以由计算机执行的程序. 它要符合以下标准:

- **有限性**：该程序的步骤数量是有限的，并且程序运行总会在有限时间内完毕。
- **确定性**：每一步都被精确、严格、明确地指定。
- **输入**：程序可以接受外来数据。输入数值必须在可接受范围内。
- **输出**：程序要能产生结果/答案作为其输出。
- **有效性**：程序中的每个操作都是明显可行的，并且可以通过计算机明确执行。

输入-加工-输出

小鸡和白兔的程序

小明养了许多小鸡和白兔。他的这两种动物一共有 H 个头。一共有 L 只腿。那么小明到底有几只鸡和几只兔呢？这是一个小学或初中的数学题。



如果已知有 12 只鸡和 7 只兔，那么 $H = 19$。因为每只鸡有两条腿，每只兔子有 4 条腿，$L = 24 + 28 = 52$。这很容易。

反过来就有点难了。问题是 “从已知头的总数和腿的总数，我们如何知到鸡和兔子的数量?”。

我们先来看一些简单的例子。如果我们知道 $H = 1$ 和 $L = 2$，那么我们只有一只鸡，没有兔子。如果我们知道 $H = 1$ 和 $L = 4$，那么我们有一只兔子，没有鸡。等等。

现在我们可以编一个程序来解这个题。我们将这个程序命名为 `CRprog`。

首先来看看程序 `CRprog` 的输入输出：

输入：头的总数 H 和腿的总数 L
输出：鸡的总数 c 和兔的的总数 r

当然，输入值 H 和 L 必须是正整数或零。

然后通过一系列步骤，让程序计算 c 和 r 的正确值。最后程序将这些值显示为输出。

计算 c 和 r

现在是时候谈谈如何解决这个问题了。那就是如何用给定的输入 H（头数）和 L（腿数）计算 c（鸡数）和 r（兔数）。

方法不止一种。我们先用 “蛮力法”——逐个尝试 r 所有可能的值：

$$r = 0,\ r = 1,\ r = 2,\ r = 3,\ r = 4,\ ...,\ r = H$$

换句话说，尝试 “没有兔子” 到 “都是兔子”，我们迟早会找到答案。所以 r 的每一个值只是一个**猜测**，需要测试它到底对不对。蛮力法又叫穷举搜索(exhaustive search)。就是系统地列举并检查所有可能的解决方案。这种方法不厌其烦但很有用处。

在我们的解法中，猜测兔子的数量为 r 意味着相应的鸡的数量猜测为 $c = H - r$，其中 H 是已知的头数。

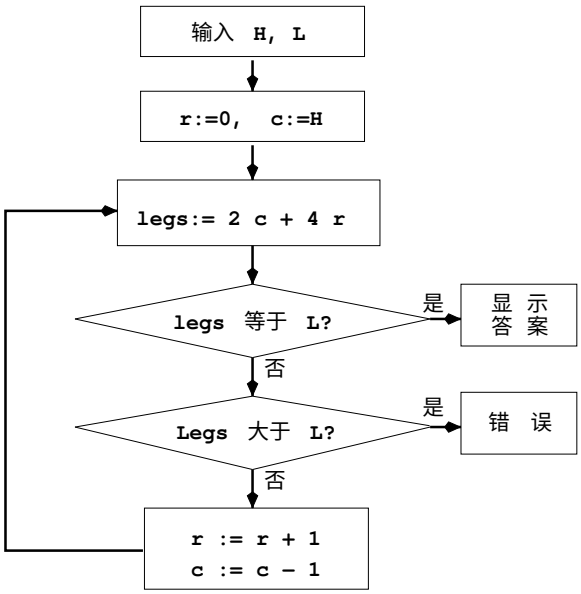
测试任何一对 r 和 c 的正确性，只要查看它们是否符合给定的腿数 L。具体来说，我们测试以下等式是否成立：

$$2 \times c + 4 \times r = L$$

现在让我们给一个 `CRprog` **蛮力版** 的流程图，可以让整个过程更加明白。

流程图

流程图是用来表示工作步骤的。它以不同的框代表不同种类的步骤，步骤之间以箭头连接。这样便于说明解决问题的方法。流程图在分析、设计、及操控等领域都有广泛应用。在编程方面尤其方便。



在流程图中，**赋值号** := 表示左侧的变量得到右侧的值。在图中你也会看到一个迭代 (iteration)，一组连续重复执行的步骤。在程序中迭代也叫循环 (loop)。使用迭代让一个程序简单地尝试所有可能性。

一步一步的程序

现在我们照流程图写下蛮力法程序的步骤。注解用于解释说明，而不是程序指令。

CRprog **蛮力版**:

1. 接收输入值 H 和 L
   注解: H 头数, L 腿数

2. 让 $r := 0$; 让 $c := H$
   注解: r 兔数初始值, c 鸡数初始值

3. **repeat**: 让 $legs := (2 \times c + 4 \times r)$
   注解: 用 r 和 c 的值算出腿数

4. 如果 $legs$ 等于 L 那么跳转到 **answer**
   注解: 如果 $legs$ 等于给定的 L 我们就找到了答案

5. 如果 $legs$ 大于 L 那么跳转到 **error**
   注解: 如果 $legs$ 大于 L 那就出问题了

6. 让 $r := r + 1$; 让 $c := c - 1$
   注解: 下一个猜测: 加一只兔子，减一只鸡

7. 跳转到 **repeat**

8. **answer**: 展示 "c 只鸡和 r 只兔"; 停止

9. **error**: 展示 "没有找到解"; 停止

理解一个程序最好是跟着它的控制流 (control flow) 走一遍: 从头开始一步接着一步的往下走。这样你就很容易了解该程序是如何作用的。

你现在可以从具体的输入值开始按照步骤进行操作，例如 $H = 11$ 和 $L = 28$。看看你是否得到 "8 只鸡和 3 只兔子" 的结果。

如果这是你第一个程序，那么恭喜！因为你已经迈出了关键的一步。

垃圾进垃圾出

一个程序通常要立即验证输入数据。如果输入无效，就不必要继续了。

看看 $H = 9$ 和 $L = 15$ 怎么样? 这个输入无效，因为 L 必须是偶数。当然，任何负或非整数值的 H 或 L 也显然是无效的。任何低于 $2H$ 或高于 $4H$ 的 L 值也是无效的。

但是输入 $H = 0$ 和 $L = 0$ 可以吗? 不妨用 `CRprog` **蛮力版** 试试。这就是一种极端情况。好的程序应该能应付所有极端情况。

你能想到其他的测试吗? 请記得在第 1 步之前添加这些输入有效性的测试。

程序效率

上述蛮力方法虽然可行但真的不太好，因为它可能要经过多次试错才能得到答案。这会花费不少计算力。我们何不看看如何减少试错。例如用 r 大约 H 的一半作为最初的猜测呢?

其实不管最初的猜测 r 是什么，我们都可以计算 $c = H - r$ 和

$$legs = 2 \times c + 4 \times r$$
$$d = legs - L$$

如果 d 等于零，我们就已经找到了答案。如果 d 是正数，那么我们有 $d/2$ 太多的兔子了。如果 d 是负数，我们有 $-d/2$ 太少的兔子了。在任一情况下 $r + d/2$ 就是正确的兔子数量，因而得到了解。这方法只需要一个猜测。

但我们可以做得更好，不用任何猜测就能得到正确的答案。在接下来的两个方程上使用代数解未知数 r 和 s

$$r + c = H$$
$$2 \times r + c = L/2$$

我们直接得到

$$r = L/2 - H$$

这意味着我们实际上可以一**步到位**计算出正确的 r 和 $c = H - r$。这多有效啊！

编程的阶段

我们可以有系统的，分阶段的编写程序来解决既定的问题。

1. 首先想想解决问题的不同方法。制定整体解决策略和方案。
2. 建立流程图来表明设计并梳理逻辑。从一个步骤流到下一个步骤。
3. 写下一个程序，检查输入、实际步骤、对付不同的情况，并产生答案。
4. 用各种输入值在纸上试运行程序，注意极端情况。
5. 更正、改进和完善。

程序的改进

程序的建立多半都会有一个渐进的过程，逐步的精进和完善。现在让我们看看鸡和兔子程序的精簡版。可以一步到位。

`CRprog` **精簡版**:

1. 接收输入值 H 和 L

2. 检查 H 和 L 的有效性，如果无效跳转到**错误**

3. 让 $r := L/2 - H$; 让 $c := H - r$

4. 显示 "c 只鸡和 r 只兔子"；停止

5. **错误**: 显示 "无效输入—未找到解决方案"；停止

程序编码

当我们把程序弄清楚并准确表述后，就可以用不同的程序语言来进行编码。

流行的编程语言包括 C/C++, Java, Perl, Javascript, Python, basic, PhP, 等等。编码是一种类似翻译的活动: 将用中文, 英文或其他自然语言给出的程序，翻译成 C++ 或 Python 编码语言。

程序一旦变成所要的编码型式它便可以在目标计算机上运行。然后就可以进行测试、调整、修订和更新。准备就绪后，该程序可以发布以供一般使用, 直到下一个版本取代它。

最后: 一个好的开始

小鸡和白兔为我们介绍了程序编写的完整例子。也说明了编程如何涉及解决问题的策略、计划、发现问题、准备预案、精确的步骤、迭代的力量、逻辑和效率思维。

这简单易懂的例子还展示了一种系统的编程方法：确定解决方案策略，创建流程图，列出逐步过程，测试，修改，用编程语言编写代码，最后运行代码，调试和发布。

这好像没什么大不了。但请不要误认编程只是个人的努力，或者它是如此简单。其实编程是一门经常需要团队合作的**艺术**。实在讲，编程是计算和计算机科学的一个广阔而深入的领域，包括软件工程、编程语言、编译器、算法设计和分析、应用程序接口 (API) 设计、协议、数据结构、数据库等等。精通编程在职场是非常有价值的。

一个简单的例子不能让任何人成为程序员。但至少你可以窥见那个世界。不是每个人都会喜欢编程。但是如果你喜欢，希望这篇文章能给你起个头。至少它可以增加你的计算思维。

顺便说一句，这个程序有一个 Javascript 版本可以在线尝试和下载。有兴趣请联系作者 (请到 www.cs.kent.edu/~pwang)。