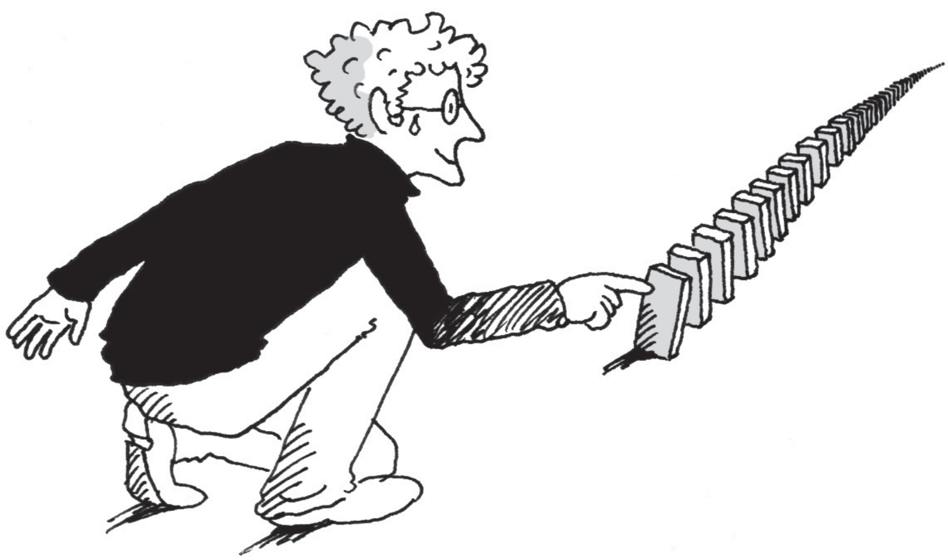
CHAPTER 4

第 4 章

数学归纳法
——如何征服无穷数列



◎ 课前对话

老师：假设现在有一排多米诺骨牌。如何将它们全部推倒呢?

学生：这个简单！只要将它们排列成其中一个一倒就能顺次带倒下一个的形状就行了。

老师：这样还不够噢!

学生：啊？为什么呢?

老师：因为还需要推倒第一个多米诺骨牌。

学生：那不是理所当然的嘛!

老师：正是！这样你就能理解数学归纳法的两个步骤了。

本章学习内容

本章我们要学习的是数学归纳法。数学归纳法是证明某断言对于 0 以上的所有整数（$0, 1, 2, 3, \cdots$）都成立的方法。0 以上的整数 $0, 1, 2, 3, \cdots$ 有无穷个，但若使用数学归纳法，只需要经过“两个步骤”，就能证明有关无穷的命题。

首先，我们以求出 1 到 100 之和为例介绍数学归纳法。接着会穿插几道思考题来看一下数学归纳法的具体实例。最后，我们会讨论数学归纳法和编程的关系，一起了解一下循环不变式。

高斯求和

思考题（存钱罐里的钱）

在你面前有一个空存钱罐。

- 第 1 天，往存钱罐里投入 1 元。存钱罐中总金额为 1 元
- 第 2 天，往存钱罐里投入 2 元。存钱罐中总金额为 $1 + 2 = 3$ 元
- 第 3 天，往存钱罐里投入 3 元。存钱罐中总金额为 $1 + 2 + 3 = 6$ 元
- 第 4 天，往存钱罐里投入 4 元。存钱罐中总金额为 $1 + 2 + 3 + 4 = 10$ 元

那么，每天都这样往存钱罐里投入硬币的话，第 100 天时的总金额为多少呢?

思考一下

本题要求算出第 100 天时存钱罐的总金额。要求出第 100 天的金额，只要计算 $1+2+3+\cdots+100$ 的值就行了。那么，具体应如何计算呢？

一般来说，最先想到的肯定是机械地将它们逐个相加。1 加 2, 再加 3, 再加 4, ⋯, 再加 99, 再加 100。只要这样加起来就能得出答案了吧。如果说笔算比较花时间的话，也可以使用计算器或编程来计算。

不过，德国数学家高斯在 9 岁时遇到了同样的问题，却马上得出了答案。当时他既没用计算器也没用计算机。那么，他究竟是如何做到的呢？

小高斯的解答

小高斯是这么考虑的。

$1+2+3+\cdots+100$ 顺次计算的结果和 $100+99+98+\cdots+1$ 逆向计算的结果应该是相等的。那么，就将这两串数字像下面那样纵向地相加。

$$
\begin{array}{rrrrrrr}
 & 1\ + & 2\ + & 3\ + & \cdots + & 99\ + & 100 \\
+) & 100\ + & 99\ + & 98\ + & \cdots + & 2\ + & 1 \\
\hline
 & \multicolumn{6}{c}{\underbrace{101+101+101+\cdots+101+101}_{\text{有 100 个 101}}}
\end{array}
$$

如此一来，就变成了 $101+101+101+\cdots+101$ 那样 100 个 101 相加的结果。这样的计算就非常简单了。只要将 101 乘以 100 即可，结果为 10 100。不过 10 100 是要求的数的 2 倍，因此还得除以 2，答案为 5050。

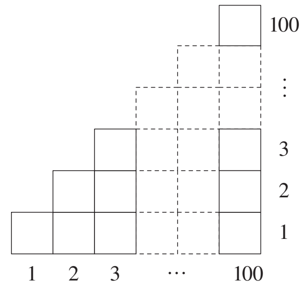
答案：5050 元。

讨论一下小高斯的解答

小高斯的方法可谓绝妙非凡！

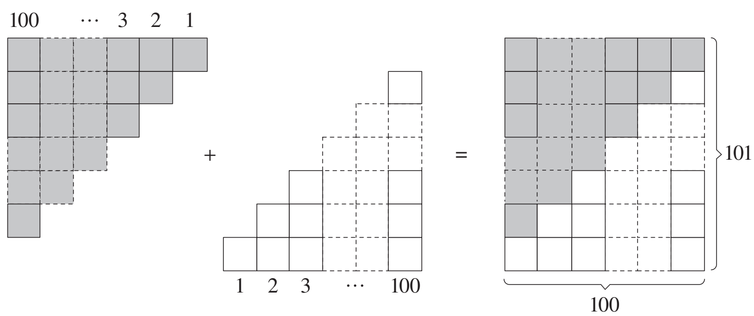
为了便于大家理解，我们将小高斯的方法用图来表示。求 $1+2+3+\cdots+100$ 的结果，相当于计算图 4-1 所示的排列成阶梯型的瓷砖块数。

图 4-1 将小高斯的方法图形化



小高斯则又做了一个一模一样的阶梯，并将两者合二为一，组成了一个长方形（图 4-2）。

图 4-2 将两个阶梯组合成一个长方形



由两个阶梯组合而成的长方形，纵向有 101 块瓷砖，横向有 100 块瓷砖。因此，该长方形由 101 × 100 = 10 100 块瓷砖构成。而所求的瓷砖块数就是 10 100 的一半，即 5050。

我们来说一说小高斯的计算效率。使用他的方法不需要花费力气逐个相加。只要将两端的 1 和 100 相加，结果乘以 100 再除以 2 就行了。

现在，假设我们不是从 1 加到 100，而是从 1 加到 10 000 000 000（100 亿）。这次我们就不能采用逐一相加的方法了。因为即使计算器 1 秒能完成 1 次加法计算，加到 100 亿也得花 300 年以上的时间。

不过，如果使用小高斯的方法，那么从 1 加到 100 亿也只要 1 次加法、1 次乘法、1 次

除法运算即可完事。我们来实际计算一下。

$$\frac{(10\,000\,000\,000+1)\times 10\,000\,000\,000}{2}=50\,000\,000\,005\,000\,000\,000$$

高斯（Karl Friedrich Gauss，1777—1855）后来成为了历史上著名的数学家。

归纳

小高斯运用了以下等式。

$$1+2+3+\cdots+100=\frac{(100+1)\times 100}{2}$$

这里，使用变量 n，将“1 到 100”归纳为“0 到 n”。这样，上面的等式就变为如下形式。

$$0+1+2+3+\cdots+n=\frac{(n+1)\times n}{2}$$

那么，这个等式对于 0 以上的任意整数 n 都成立吗？即 n 为 100、200，或者 100 万、100 亿时该等式也都成立吗？如果成立的话，又如何来证明呢？

这种时候就要用到数学归纳法了。数学归纳法是证明“断言对于 0 以上的所有整数 n 都成立”的方法。

学生：“对于所有整数 n”，总觉得这种说法别扭。

老师：别扭？

学生：会感觉头脑中充满了整数。

老师：那么，改为“对于任一整数 n”怎么样？

学生：啊！那样感觉稍微舒服些。

老师：其实说的是一回事呢！

数学归纳法——如何征服无穷数列

本节，我们就来讨论一下数学归纳法的相关内容。首先，从“0 以上的整数的断言”开始学起，然后使用数学归纳法来证明小高斯的断言。

0 以上的整数的断言

“0 以上的整数 n 的断言”，就是能够判定 0, 1, 2 等各个整数为“真”或“假”的断言。这样说明或许难以理解，下面就举几个例子。

●例 1

- 断言 $A(n)$：$n \times 2$ 为偶数

$A(n)$，即“$n \times 2$ 为偶数”的断言。由于 n 为 0 时，$0 \times 2 = 0$ 为偶数，所以 $A(0)$ 为真。

$A(1)$ 又怎么样呢？因为 $1 \times 2 = 2$ 为偶数，所以 $A(1)$ 也为真。

那是否可以说断言 $A(n)$，对于 0 以上的所有整数 n 都为真呢？

对！可以这么说。因为 0 以上的任意整数乘以 2 的结果都为偶数，所以对于 0 以上的所有整数，断言 $A(n)$ 都为真。

●例 2

- 断言 $B(n)$：$n \times 3$ 为奇数

那么，断言 $B(n)$ 又将如何呢？该断言对于 0 以上的所有整数 n 都成立吗？

例如，假设 n 为 1，则断言 $B(1)$ 就是“1×3 为奇数”，这个结果为真。但不能说对于 0 以上的所有整数 n，断言 $B(n)$ 都为真。因为假设 n 为 2，则 $n \times 3$ 的值为 $2 \times 3 = 6$。而 6 是偶数，所以断言 $B(2)$ 不为真（为假）。

$n = 2$ 是推翻“断言 $B(n)$ 对于 0 以上的所有整数 n 都成立”的反例之一。

●其他例子

那么请思考一下，在下面 4 个断言中，对于 0 以上的所有整数 n 都成立的有哪些。

- 断言 $C(n)$：$n + 1$ 为 0 以上的整数
- 断言 $D(n)$：$n - 1$ 为 0 以上的整数
- 断言 $E(n)$：$n \times 2$ 为 0 以上的整数
- 断言 $F(n)$：$n \div 2$ 为 0 以上的整数

断言 $C(n)$，对于 0 以上的所有整数 n 都成立。因为若 n 为 0 以上的整数，则 $n + 1$ 肯定是 0 以上的整数。

断言 $D(n)$，对于 0 以上的所有整数 n 不成立。例如，断言 $D(0)$ 为假。因为 $0-1=-1$，不是 0 以上的整数。$n=0$ 是唯一的反例。

断言 $E(n)$，对于 0 以上的所有整数 n 都成立。

断言 $F(n)$，对于 0 以上的所有整数 n 不成立。因为当 n 为奇数时，$n\div 2$ 的结果不是整数。

小高斯的断言

在讨论了“0 以上的整数 n 的断言”之后，我们将话题转回小高斯的断言。

可以使用下述有关 n 的断言形式来表现小高斯的观点。

- 断言 $G(n)$：0 到 n 的整数之和为 $\frac{n\times(n+1)}{2}$

接下来要证明的是，“$G(n)$ 对于 0 以上的所有整数 n 都成立”。可以通过描画前面的阶梯状的图（图 4-1）来证明，但是有人可能会有这样的疑问：0 以上的整数有 $0,1,2,3$ 等**无穷个**数，而图中表现的只是其中一种情况。当 $G(1\,000\,000)$ 时也成立吗?

确实，0 以上的整数有无穷个。这就要通过引入“数学归纳法”来证明了。使用数学归纳法能够进行 0 以上的所有整数的相关证明。

什么是数学归纳法

数学归纳法是证明有关整数的断言对于 0 以上的所有整数（$0,1,2,3,\cdots$）是否成立时所用的方法。

假设现在要用数学归纳法来证明“断言 $P(n)$ 对于 0 以上的所有整数 n 都成立”。

数学归纳法要经过以下两个步骤进行证明。这是本章的核心内容，请大家仔细阅读。

- 步骤 1

  证明“$P(0)$ 成立”
- 步骤 2

  证明不论 k 为 0 以上的哪个整数，“若 $P(k)$ 成立，则 $P(k+1)$ 也成立”

在步骤 1 中，要证明当 k 为 0 时断言 $P(0)$ 成立。我们将步骤 1 称作**基底**（base）。

在步骤 2 中，要证明无论 k 为 0 以上的哪个整数，“若 $P(k)$ 成立，则 $P(k+1)$ 也成立”。

我们将步骤 2 称作**归纳**（induction）。该步骤证明断言若对于 0 以上的某个整数成立，则对于下一个整数也成立。

若步骤 1 和步骤 2 都能得到证明，就证明了“断言 $P(n)$ 对于 0 以上的所有整数 n 都成立”。

以上就是数学归纳法的证明方法。

试着征服无穷数列

数学归纳法通过步骤 1（基底）和步骤 2（归纳）两个步骤，证明断言 $P(n)$ 对于 0 以上的所有整数 n 都成立。

为什么只通过两个步骤的证明，就能证明无穷的 n 呢？请作如下思考。

- 断言 $P(0)$ 成立

  理由：步骤 1 中已经证明。
- 断言 $P(1)$ 成立

  理由：$P(0)$ 已经成立，并且步骤 2 中已证明若 $P(0)$ 成立，则 $P(1)$ 也成立。
- 断言 $P(2)$ 成立

  理由：$P(1)$ 已经成立，并且步骤 2 中已证明若 $P(1)$ 成立，则 $P(2)$ 也成立。
- 断言 $P(3)$ 成立

  理由：$P(2)$ 已经成立，并且步骤 2 中已证明若 $P(2)$ 成立，则 $P(3)$ 也成立。

这样循环往复，可以说断言 $P(n)$ 对于任意整数 n 都成立。无论 n 为多大的整数都没关系。因为即使设 n 为 10 000 000 000 000 000，经过机械式地反复执行步骤 2，终究可以证明 $P(10\,000\,000\,000\,000\,000)$ 成立。

这种数学归纳法的思路可以比喻为“推倒多米诺骨牌”。

假设现在有很多多米诺骨牌排成一列。只要保证以下两个步骤，那么无论多米诺骨牌排得有多长最终都能倒下。

- 步骤 1

  确保让第 0 个多米诺骨牌（排头的多米诺骨牌）倒下
- 步骤 2

  确保只要推倒第 k 个多米诺骨牌，那么第 $k+1$ 个多米诺骨牌也会倒下

推倒多米诺骨牌的两个步骤和数学归纳法的两个步骤一一对应。

数学归纳法并不像“推倒多米诺骨牌”那样关注所用的时间。数学归纳法和编程不同，往往使用的是忽略时间的方法。这就是数学和编程之间最大的差异。

用数学归纳法证明小高斯的断言

下面我们就以证明小高斯的断言 $G(n)$ 为例具体看看数学归纳法。首先讨论断言 $G(n)$。

- 断言 $G(n)$：0 到 n 的整数之和与 $\frac{n\times(n+1)}{2}$ 相等

使用数学归纳法就需要通过步骤 1（基底）和步骤 2（归纳）来证明。

●步骤 1：基底的证明

证明 $G(0)$ 成立。

$G(0)$ 就是“0 到 0 的整数之和与 $\frac{0\times(0+1)}{2}$ 相等”。

这可以通过直接计算证明。0 到 0 的整数之和是 0，$\frac{0\times(0+1)}{2}$ 也是 0。

至此，步骤 1 证明完毕。

●步骤 2：归纳的证明

证明当 k 为 0 以上的任一整数时，“若 $G(k)$ 成立，则 $G(k+1)$ 也成立”。

现假设 $G(k)$ 成立。即假设“0 到 k 的整数之和与 $\frac{k\times(k+1)}{2}$ 相等”。这时，以下等式成立。

假设成立的等式 $G(k)$

$$0+1+2+\cdots+k=\frac{k\times(k+1)}{2}$$

下面，我们来证明 $G(k+1)$ 成立。

要证明的等式 $G(k+1)$

$$0+1+2+\cdots+k+(k+1)=\frac{(k+1)\times((k+1)+1)}{2}$$

$G(k+1)$ 的左边使用假设的等式 $G(k)$ 可以进行如下计算。

$$
\begin{aligned}
G(k+1)\text{ 的左边} &= \underbrace{0+1+2+\cdots+k}_{G(k)\text{ 的左边}}+(k+1) \\
&= \underbrace{\frac{k\times(k+1)}{2}}_{G(k)\text{ 的右边}}+(k+1) && \text{将 } G(k) \text{ 的左边替换为 } G(k) \text{ 的右边} \\
&= \frac{k\times(k+1)}{2}+\frac{2\times(k+1)}{2} && \text{将 } (k+1) \text{ 转换为分数形式} \\
&= \frac{k\times(k+1)+2\times(k+1)}{2} && \text{分母相同，分子相加} \\
&= \frac{(k+1)\times(k+2)}{2} && \text{合并同类项 } (k+1)
\end{aligned}
$$

而 $G(k+1)$ 的右边可以进行如下计算。

$$
\begin{aligned}
G(k+1)\text{ 的右边} &= \frac{(k+1)\times((k+1)+1)}{2} \\
&= \frac{(k+1)\times(k+2)}{2} && \text{算出 } ((k+1)+1) \text{ 的结果}
\end{aligned}
$$

$G(k+1)$ 的左边和右边的计算结果相同。

由此，从 $G(k)$ 到 $G(k+1)$ 推导成功，步骤 2 得到了证明。

至此，通过数学归纳法的步骤 1 和步骤 2 证明了断言 $G(n)$。也就是说通过数学归纳法证明了断言 $G(n)$ 对于 0 以上的任意整数 n 都成立。

求出奇数的和——数学归纳法实例

本节，我们使用数学归纳法来证明另一个断言。

通过数学归纳法证明

请证明以下断言 $Q(n)$ 对于 1 以上的所有整数 n 都成立。

- 断言 $Q(n)$：$1+3+5+7+\cdots+(2\times n-1)=n^2$

$Q(n)$ 是比较有意思的断言。按从小到大的顺序将 n 个奇数相加，得到 n^2，即平方数 $n \times n$。这对吗？在证明之前，先通过较小的数 $n = 1, 2, 3, 4, 5$ 判断 $Q(n)$ 的真假。

- 断言 $Q(1)$：$1 = 1^2$
- 断言 $Q(2)$：$1 + 3 = 2^2$
- 断言 $Q(3)$：$1 + 3 + 5 = 3^2$
- 断言 $Q(4)$：$1 + 3 + 5 + 7 = 4^2$
- 断言 $Q(5)$：$1 + 3 + 5 + 7 + 9 = 5^2$

通过以上计算发现断言确实是成立的。

通过数学归纳法证明

下面我们来证明“断言 $Q(n)$ 对于 1 以上的所有整数 n 都成立”。为此，需要通过数学归纳法的两个步骤进行证明。

虽然这次要证明的不是“0 以上的……”，而是“1 以上的……”，但只要将 0 换成 1 来进行基底的证明就可以使用数学归纳法了。

●步骤 1：基底的证明

证明 $Q(1)$ 成立。

因为 $Q(1) = 1^2$，所以确实成立。

步骤 1 证明完毕。

●步骤 2：归纳的证明

证明 k 为 1 以上的任意整数时，“若 $Q(k)$ 成立，则 $Q(k+1)$ 也成立”。现假设 $Q(k)$ 成立，即以下等式成立。

假设成立的等式 $Q(k)$

$$1 + 3 + 5 + 7 + \cdots + (2 \times k - 1) = k^2$$

下面证明 $Q(k+1)$ 等式成立。

要证明的等式 $Q(k+1)$

$$1 + 3 + 5 + 7 + \cdots + (2 \times k - 1) + (2 \times (k+1) - 1) = (k+1)^2$$

$Q(k+1)$ 的左边使用假设的等式 $Q(k)$ 可以进行如下计算。

$$
\begin{aligned}
Q(k+1)\text{ 的左边} &= \underbrace{1+3+5+7+\cdots+(2\times k-1)}_{Q(k)\text{ 的左边}}+(2\times(k+1)-1) \\
&= \underbrace{k^2}_{Q(k)\text{ 的右边}}+(2\times(k+1)-1) && \text{将 } Q(k) \text{ 的左边替换为 } Q(k) \text{ 的右边} \\
&= k^2+2\times k+2-1 && \text{展开 } 2\times(k+1) \\
&= k^2+2\times k+1 && \text{计算 } 2-1
\end{aligned}
$$

而 $Q(k+1)$ 的右边可以进行如下计算。

$$
\begin{aligned}
Q(k+1)\text{ 的右边} &= (k+1)^2 \\
&= k^2+2\times k+1 && \text{展开 } (k+1)^2
\end{aligned}
$$

$Q(k+1)$ 的左边和右边计算结果相同。

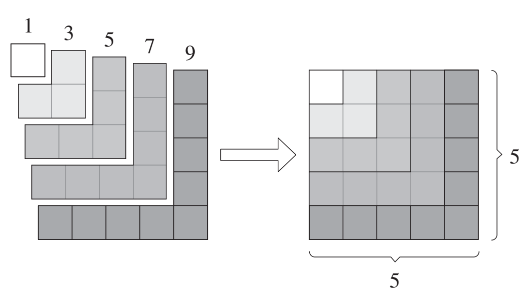
由此，从 $Q(k)$ 到 $Q(k+1)$ 推导成功，步骤 2 得到了证明。

至此，通过数学归纳法的步骤 1 和步骤 2 证明了断言 $Q(n)$。也就是说，通过数学归纳法，证明了断言 $Q(n)$ 对于 1 以上的任意整数 n 都成立。

图形化说明

断言 $Q(n)$ 也可以用图来进行说明。下面我们来看看 $Q(5)$ 的图示（图 4-3）。

图 4-3 $Q(5)$ 的图示



1 块瓷砖、3 块瓷砖、5 块瓷砖、7 块瓷砖、9 块瓷砖可以构成 5×5 的正方形。这正好相当于断言 $Q(5)$。

通过图示来进行说明是直观易懂的。但是过于依赖图就有问题了。下一节我会举出几个容易为图所惑的例子，一起看看吧。

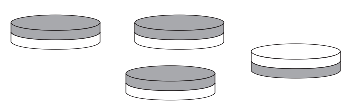
黑白棋思考题——错误的数学归纳法

本节，我们来看几个使用数学归纳法时被图干扰的例子。问题已经准备好了，我们来找出证明过程中的错误吧。

思考题（黑白棋子的颜色）

黑白棋一面是白色，一面是黑色（图 4-4）。现在，我们往棋盘上随便扔几枚棋子。有时会碰巧都是白色或都是黑色。但有时既有白棋，也有黑棋。

图 4-4　黑白棋的颜色（一面是白色，另一面是黑色）



使用数学归纳法可以“证明”投掷的黑白棋的颜色一定相同。然而现实中这却是不可能的。

那么，请找出下述“证明”中的错误之处。

假设 n 为 1 以上的整数，用数学归纳法证明以下断言 $T(n)$ 对于 1 以上的所有整数 n 都成立。

- 断言 $T(n)$：投掷 n 枚黑白棋，所有棋子的颜色一定相同

● 步骤 1：基底的证明

证明 $T(1)$ 成立。

断言 $T(1)$ 即“投掷 1 枚黑白棋子时，所有棋子的颜色一定相同”。棋子只有 1 个，颜色当然只有 1 种，因此 $T(1)$ 成立。

这样，步骤 1 就得到了证明。

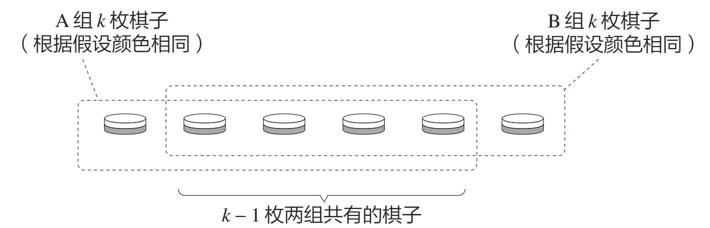
●步骤 2：归纳的证明

证明当 k 为 1 以上的任意整数时，“若 $T(k)$ 成立，则 $T(k+1)$ 也成立”。

首先假设“投掷 k 枚黑白棋子时，所有棋子的颜色一定相同”成立。现假设投掷 k 枚棋子后，再投掷一枚黑白棋。那么投掷的棋子总数为 $k+1$ 枚。

这里，将投掷的棋子以每 k 枚为单位分为两组，分别将这两组称为 A 和 B（图 4-5）。

图 4-5　将投掷的棋子以每 k 枚为单位分为两组



因为“投掷 k 枚黑白棋子时，所有的棋子的颜色一定相同”的假设成立，所以 A 组的棋子（k 枚）和 B 组的棋子（k 枚），分别都是相同色。而通过图 4-5 可见，两组共有的棋子为 $k-1$ 枚。因为各组的棋子颜色相同，又有两组共有的棋子，所以 $k+1$ 枚棋子颜色相同。这就是断言 $T(k+1)$。

这样，步骤 2 就得到了证明。

通过数学归纳法，证明了断言 $T(n)$ 对于 1 以上的所有整数 n 都成立。这个证明有什么不对的地方呢?

提示：不要为图所惑

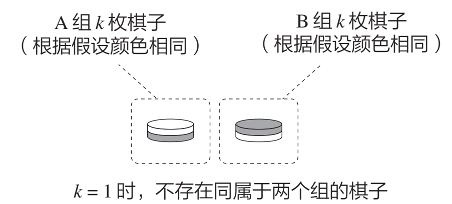
数学归纳法由两个步骤组成。我们依次看看步骤 1 和步骤 2，找找错在哪里。请注意不要为图所惑。

思考题答案

步骤 1 没有问题。若棋子只有 1 枚，那么就只有 1 种颜色。

问题在步骤 2 的图（图 4-5）中。实际上，该图在 $k = 1$ 时不成立。$k = 1$ 时，两组棋子分别都只有 1 枚。双方共有的棋子为 $k - 1$ 枚，而 $k - 1 = 0$，所以不存在同属于两个组的棋子（图 4-6）。

图 4-6　$k = 1$ 的情况



因此在数学归纳法的两个步骤中，步骤 2 是无法得到证明的。

图虽然方便，但是通过本例可知，光靠图来解题是可能存在问题的。

编程和数学归纳法

下面我们站在程序员的角度来思考数学归纳法。

通过循环表示数学归纳法

程序员朋友在学习数学归纳法时，将证明当作编程来考虑可能更容易理解。例如，代码清单 4-1 所示的程序是一个 C 语言函数，功能是“证明断言 $P(n)$ 对于给定的 0 以上的整数 n 都成立”。如果完成了步骤 1 和步骤 2 的证明，那么只要调用该函数就能将“对于任意整数 n，$P(n)$ 成立”的证明过程显示出来。

代码清单 4-1 prove 函数，证明 $P(n)$ 成立

```
void prove(int n)
{
    int k;

    printf(" 现在开始证明 P(%d) 成立。\n", n);
    k = 0;
    printf(" 根据步骤 1 得出 P(%d) 成立。\n", k);
    while (k < n) {
        printf(" 根据步骤 2 可以说“若 P(%d) 成立，则 P(%d) 也成立”。\n", k, k + 1);
        printf(" 因此，可以说“P(%d) 是成立的”。\n", k + 1);
        k = k + 1;
    }
    printf(" 证明结束。\n");
}
```

传入实际的参数，调用 prove(n) 函数，会输出断言 $P(n)$ 成立的证明过程。

例如，调用 prove(0)，会输出下述断言 $P(0)$ 的证明过程。

```
现在开始证明 P(0) 成立。
根据步骤 1 得出 P(0) 成立。
证明结束。
```

而调用 prove(1)，会输出下述断言 $P(1)$ 的证明过程。

```
现在开始证明 P(1) 成立。
根据步骤 1 得出 P(0) 成立。
根据步骤 2 可以说“若 P(0) 成立，则 P(1) 也成立”。
因此，可以说“P(1) 是成立的”。
证明结束。
```

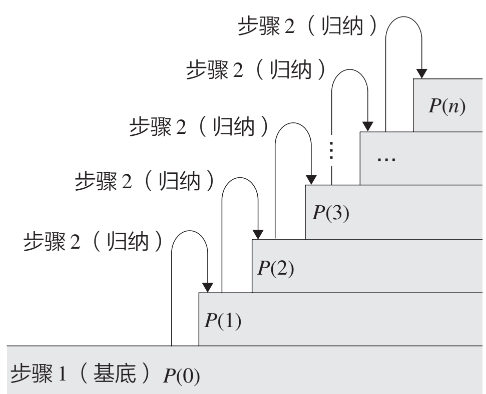
我们再调用 prove(2)，会输出下述断言 $P(2)$ 的证明过程。

现在开始证明 $P(2)$ 成立。
根据步骤 1 得出 $P(0)$ 成立。
根据步骤 2 可以说“若 $P(0)$ 成立，则 $P(1)$ 也成立”。
因此，可以说“$P(1)$ 是成立的”。
根据步骤 2，可以说“若 $P(1)$ 成立，则 $P(2)$ 也成立”。
因此，可以说“$P(2)$ 是成立的”。
证明结束。

从 prove 函数的运行结果可以发现，首先在步骤 1 中证明了出发点，然后让 k 逐次递增 1，每次都进行步骤 2 的证明。由于 C 语言的 int 类型有大小限制，实际上不能进行无穷数的证明。不过从其结构可以看出，如果反复进行步骤 2 的证明，是可以证明 $P(0)$ 到 $P(n)$ 的。

阅读这段代码之后，大家就能够理解“只通过步骤 1 和步骤 2，就证明了 0 以上的任意整数 n”这一数学归纳法的思路了吧。这就像逐层递增的阶梯（图 4-7）。

图 4-7　prove 函数的行为



在学校学习数学归纳法之初，我不是很理解这个结构。虽说等式的计算并没有那么难，但我不认为数学归纳法是有效的证明方法。当初我搞不明白的是步骤 2。在步骤 2 中，要假设 $P(k)$ 成立，推导出 $P(k+1)$。我当时却想：“$P(k)$ 不是现在要证明的式子吗？如果这样假设就谈不上证明了吧。”现在想起来，我是把 prove 函数的输入**参数 n**（目标阶梯）和

prove 函数中使用的**本地变量 k**（途经阶梯）混为一谈了。

循环不变式

熟练掌握数学归纳法的思路对于程序员来说是相当重要的。例如，要在程序中编写循环处理（loop）时数学归纳法是非常有用的。

在编写循环时，找到让每次循环都成立的逻辑表达式很重要。这种逻辑表达式称为**循环不变式**（loop invariant）。循环不变式相当于用数学归纳法证明的“断言”。

循环不变式用于证明程序的正确性。在编写循环时，思考一下“这个循环的循环不变式是什么”就能减少错误。

光这么说也许不容易理解。我还是以一个非常简单的例子来讲解循环不变式吧。

代码清单 4-2 是用 C 语言写的 sum 函数，功能是求出数组元素之和。参数 array[] 是待求和的数组，*size* 是这个数组的元素数。调用 sum 函数，会获得 array[0] 至 array[*size* − 1] 的 *size* 个元素之和。

代码清单 4-2 sum 函数，求出数组的元素之和

```
int sum(int array[], int size)
{
    int k = 0;
    int s = 0;
    while (k < size) {
        s = s + array[k];
        k = k + 1;
    }
    return s;
}
```

在 sum 函数中使用了简单的 while 循环语句。我们从数学归纳法的角度来看这个循环，得出下述断言 $M(n)$。这个断言就是循环不变式。

- 断言 $M(n)$：数组 array 的前 n 个元素之和，等于变量 s 的值

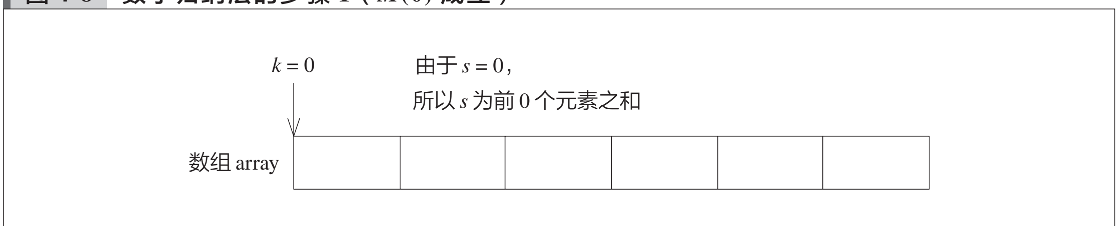
我们在程序中成立的断言上标注注释，形成代码清单 4-3 所示的代码。

代码清单 4-3　在代码清单 4-2 中成立的断言上标注注释

```
1:  int sum(int array[], int size)
2:  {
3:      int k = 0;
4:      int s = 0;
5:      /* M(0) */
6:      while (k < size) {
7:          /* M(k) */
8:          s = s + array[k];
9:          /* M(k+1) */
10:         k = k + 1;
11:         /* M(k) */
12:     }
13:     /* M(size) */
14:     return s;
15: }
```

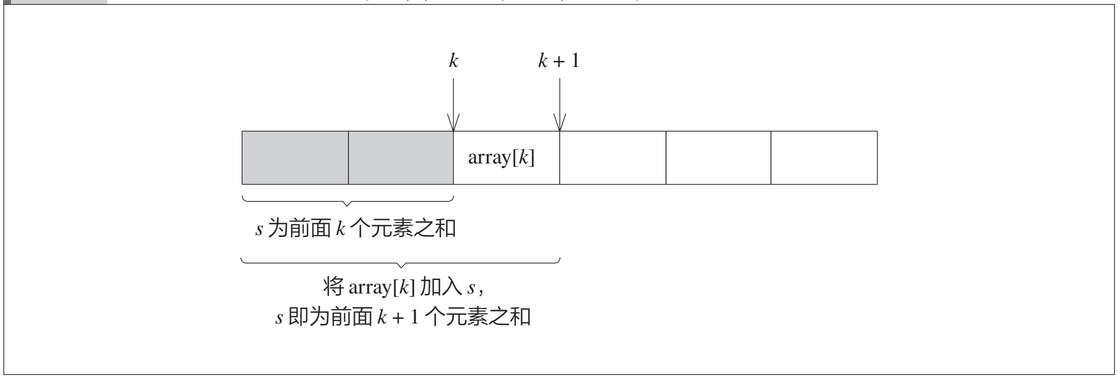
在代码清单 4-3 的第 4 行，s 初始化为 0。由此，第 5 行的 $M(0)$ 成立。$M(0)$ 即为“数组 array 的前 0 个元素之和等于变量 s 的值”。这相当于数学归纳法的步骤 1（图 4-8）。

图 4-8　数学归纳法的步骤 1（$M(0)$ 成立）



第 7 行中，$M(k)$ 成立。然后进行第 8 行的处理，将数组 array[k] 的值加入 s，因此 $M(k + 1)$ 成立。这相当于数学归纳法的步骤 2（图 4-9）。

图 4-9　数学归纳法的步骤 2（$M(k) \Rightarrow M(k+1)$ 成立）



请一定要理解第 8 行，

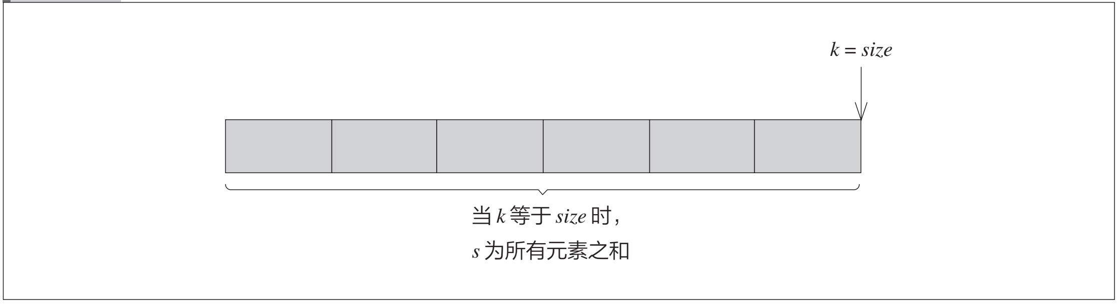
```
s = s + array[k];
```

意为"在 $M(k)$ 成立的前提下，$M(k+1)$ 成立"。

第 10 行中 k 递增 1，所以第 11 行的 $M(k)$ 成立。这里是为了下一步处理而设定变量 k 的值。

最后，第 13 行的 $M(size)$ 成立（图 4-10）。因为 while 语句中的 k 递增了 1，而这时一直满足 $M(k)$，走到第 13 行时 k 和 $size$ 的值相等。$M(size)$ 成立说明 sum 函数是没有问题的。因此，第 14 行 return 返回结果。

图 4-10　$M(size)$ 成立



综上所述，这个循环在 k 从 0 增加到 $size$ 的过程中一直保持循环不变式 $M(k)$ 成立。编写循环时，有两个注意点。一个是"达到目的"，还有一个是"适时结束循环"。循环不变式 $M(k)$ 就是为了确保"达到目的"。而 k 从 0 到 $size$ 递增确保了"适时结束循环"。

代码清单 4-4 中，写明了 $M(k)$ 成立的同时 k 递增的情形。（$\wedge$ 表示“并且”）

代码清单 4-4　$M(k)$ 成立的同时 k 递增

```
int sum(int array[], int size)
{
    int k = 0;
    int s = 0;
    /* M(k) ∧ k == 0 */
    while (k < size) {
        /* M(k) ∧ k < size */
        s = s + array[k];
        /* M(k+1) ∧ k < size */
        k = k + 1;
        /* M(k) ∧ k <= size */
    }
    /* M(k) ∧ k == size */
    return s;
}
```

看了以上循环不变 $M(k)$ 在每次循环时都成立的情形之后，大家是否都掌握了呢？

本章小结

本章我们学习了数学归纳法。数学归纳法是证明断言对于 0 以上的所有整数 n 都成立的方法。只需要两个步骤就能够证明无穷数的断言。非常有意思吧！

用数学归纳法进行证明，说起来就像是推倒有关整数的多米诺骨牌。步骤 2 的证明，就是让“下一张多米诺骨牌”倒下。为此，必须弄清楚“$P(k)$ 推进到 $P(k+1)$ 的过程”。这种数学归纳法的思路在程序员编写循环时也是非常重要的。

下一章，我们学习计数方法。

◎ 课后对话

老师：首先假设一条腿可以往前迈一步。

学生：嗯。

老师：然后假设另一条腿无论什么情况都能迈出去。

学生：那会怎样？

老师：那样的话，就能够行进到无限的远方。这就是数学归纳法。