似的链表。由于链表末尾的元素没有后续的数据，因此就需要用别的值（在这里是 −1）来填充（图 4-10）。

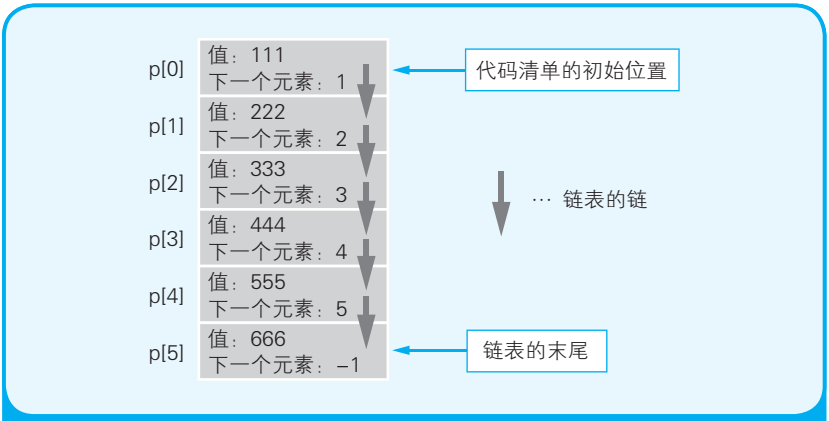

图 4-10　链表的示例（初始状态）

在需要追加或删除数据的情况下，使用链表是很高效的。首先，让我们来看一下删除的情况。在图 4-10 表示的链表中，假设要删除从起始位置开始的第 3 个元素。此时，我们只需要把第 2 个元素的“下一个元素：2”变成“下一个元素：3”即可。由于数组的元素通常是按照索引顺序来引用的，因此当我们需要引用构成链表的数组的某一个元素时，通过该元素的索引信息就可以找到下一个元素。当第 2 个元素的下一个元素变成第 4 个元素后，那么第 3 个元素就被删除了。虽然第 3 个元素在物理内存上还残留着，但在逻辑上则确实被删除了（图 4-11）。

接下来就让我们来看一下如何往链表中追加数据。假设要在图 4-10 的链表的第 5 位前追加一个新数据。此时，我们只需要在刚才消除的第 3 个元素的位置中保存新的数据，并将第 4 个元素的“下一个元素：5”变更成“下一个元素：2”，以使新追加的元素的索引信息变成

“下一个元素：5”即可。虽然新追加的元素在物理上是第 3 个，但从逻辑上看来则是第 5 个（图 4-12）。

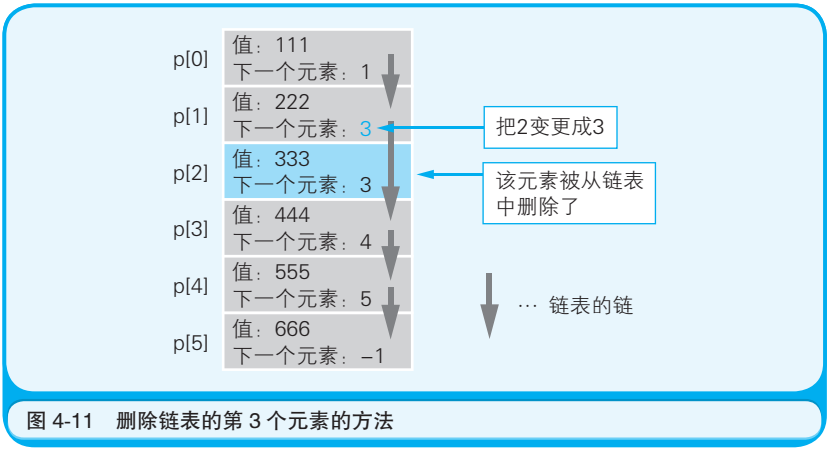

图 4-11　删除链表的第 3 个元素的方法

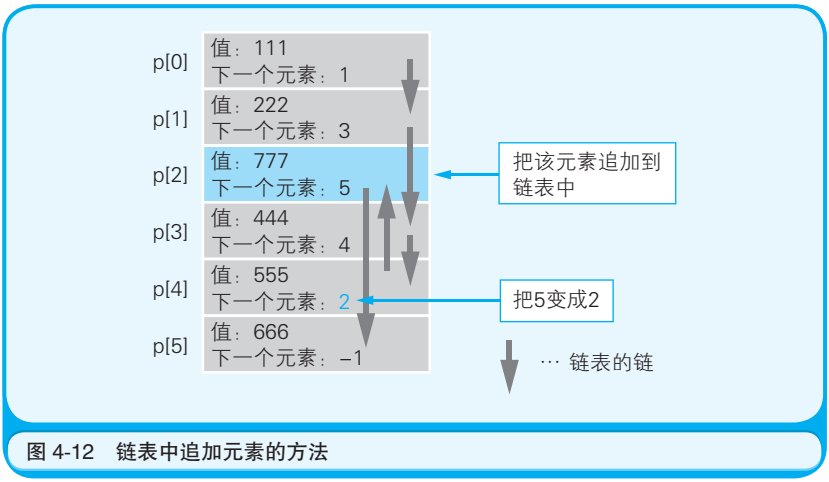

图 4-12　链表中追加元素的方法

如果不使用链表数组，那么中途删除或追加元素时，其后的元素就必须要全部移动。示例中数组的元素只有 6 个，处理起来不会花费较多时间。而在实际的程序中，有时需要对包含数千至数万个元素的

数组进行频繁的数据追加或删除操作。如果每次都需要移动数千至数万个元素，那么哪怕是高速计算机也会花费很长时间（图 4-13、图 4-14）。反之，使用代码清单来追加或删除数据则毫不费事。

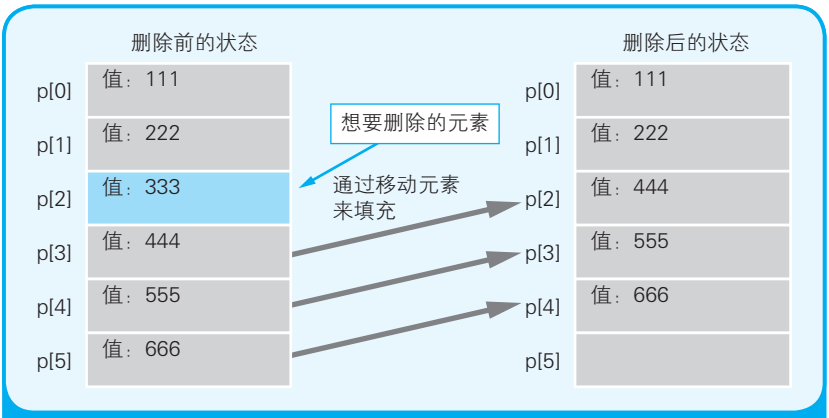

图 4-13　单纯使用数组的情况下的元素删除

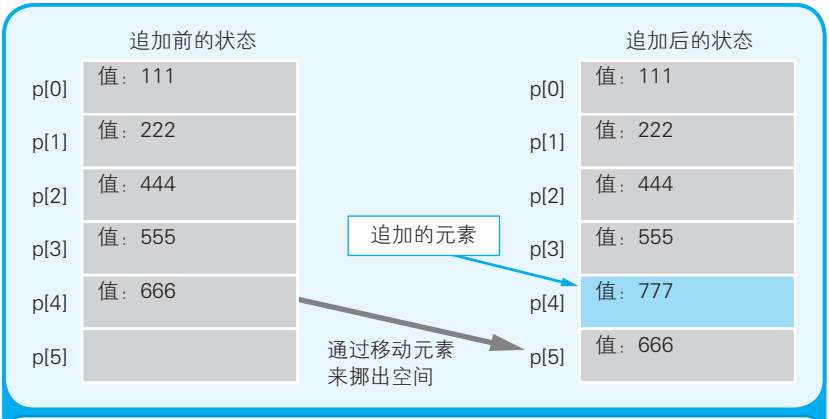

图 4-14　单纯使用数组的情况下的元素追加

4.7 二叉查找树使数据搜索更有效

二叉查找树[①]是指在链表的基础上往数组中追加元素时，考虑到数据的大小关系，将其分成左右两个方向的表现形式。例如，假设我们事先把 50 这个值保存到了数组中。那么，如果接下来的值比先前保存的数值大的话，就要将其放到右边，反之如果小的话就放在左边。但实际的内存并不会分成两个方向，这是在程序逻辑上实现的（图 4-15）。

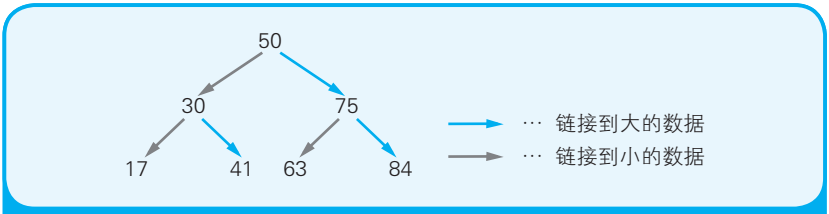

图 4-15 二叉查找树的模型（将树颠倒后的形状）

为了实现二叉查找树，怎么处理比较好呢？其实数组的每个元素中只要有数据的值和两个索引信息就可以了。图 4-16 向我们展示了如何用数组来实现图 4-14 中的二叉查找树。二叉查找树是由链表构造发展而来的表现形式，因此在追加或删除元素方面也同样是有效的。

使用二叉查找树的便利之处在于可以使数据的搜索等更有效率。在使用一般的数组时，必须从数组的开头按照索引顺序来查找目标数据。而使用二叉查找树时，当目标数据比现在读出来的数据小时就可以转到左侧，反之目标数据较大时即可转到链表的右侧，这样就加快了找到目标数据的速度。

① 树（tree）构造指的是数据像树一样分叉连接的方式。二叉查找树也是树构造的一种。

只要在程序开发中多花一些心思，我们就可以熟练地使用内存、实现栈处理、链表处理、二叉查找树处理等，这一点想必大家都清楚了。不过，大家还必须理解为什么要进行这些处理。另外，请大家牢记数组是进行这些处理的基础。

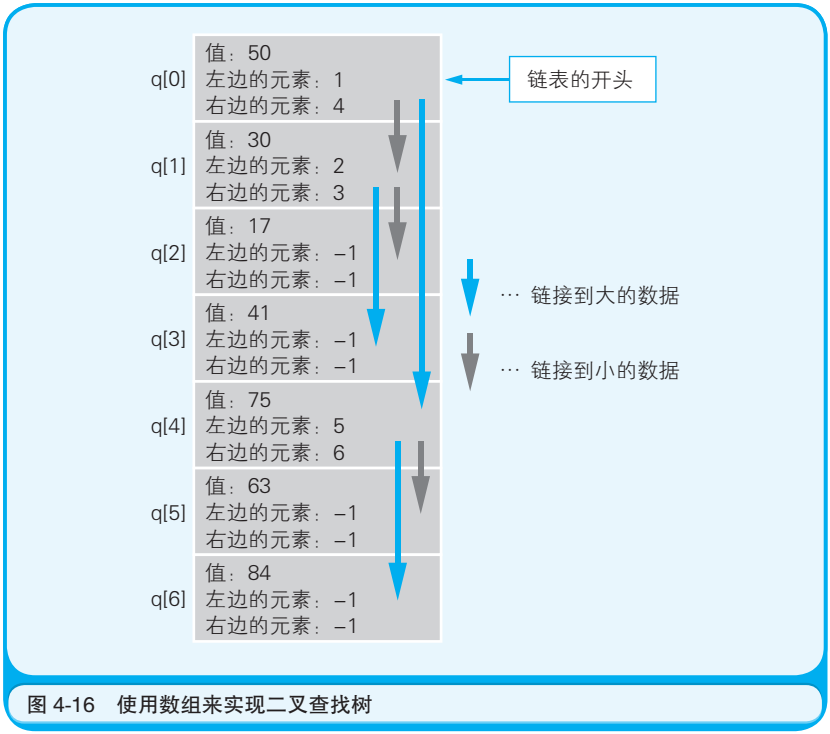

图 4-16　使用数组来实现二叉查找树

下一章，我们将会介绍磁盘。和内存一样，磁盘也是用于存储数据的。磁盘虽然在物理方面只能以扇区为单位进行读写，但通过在程序中多花一些心思，磁盘也可以以各种形态来使用。此外，我们也会对用磁盘替代内存来使用的虚拟内存进行说明。

第5章 内存和磁盘的亲密关系

热身问答

阅读正文前，让我们先回答下面的问题来热热身吧。

1. 存储程序方式指的是什么?
2. 通过使用内存来提高磁盘访问速度的机制称为什么?
3. 把磁盘的一部分作为假想内存来使用的机制称为什么?
4. Windows 中，在程序运行时，存储着可以动态加载调用的函数和数据的文件称为什么?
5. 在 EXE 程序文件中，静态加载函数的方式称为什么?
6. 在 Windows 计算机中，一般磁盘的 1 个扇区是多少字节?

怎么样？是不是发现有一些问题无法简单地解释清楚呢？下面是笔者的答案和解析，供大家参考。

1. 在存储装置中保存程序，并逐一运行的方式
2. Disk Cache（磁盘缓存）
3. 虚拟内存（virtual memory）
4. DLL（DLL 文件）
5. 静态链接
6. 512 字节

解析

1. 现在计算机采用的是存储程序方式。
2. 磁盘缓存是指，把从磁盘中读出的数据存储在内存中，当该数据再次被读取时，不是从磁盘而是直接从内存中高速读出。
3. 借助虚拟内存，哪怕是内存容量不足的计算机，也可以运行很大的程序。
4. DLL 是 Dynamic Link Liabrary 的略称。
5. 函数的加载方式有静态链接和动态链接两种。
6. 扇区是磁盘保存数据的物理单位。

本章重点

从都具有存储程序命令和数据这点来看，内存和磁盘的功能是相同的。在计算机的 5 大部件[①] 中，内存和磁盘也都被归类为存储部件。不过，利用电流来实现存储的内存，同利用磁效应来实现存储的磁盘，还是有差异的。而从存储容量来看，内存是高速高价，而磁盘则是低速廉价。

大家平时使用的计算机，至少都配备了 512M 大小的内存和 80GB 大小的磁盘。在计算机这个系统中，高速小容量的内存与低速高容量的磁盘进行协同作业。本章就让我们来看一下内存和磁盘的亲密关系。在下文中，内存主要是指主内存（负责存储 CPU 中运行的程序指令和数据的内存），磁盘主要是指硬盘。

5.1 不读入内存就无法运行

考虑内存和磁盘的关系之前，我们首先来看一个前提性的问题。

程序保存在存储设备中，通过有序地被读出来实现运行，这一点大家都很清楚。这一机制称为**存储程序方式**（程序内置方式），现在看来这是理所当然的，但在当时它的提出可以说是一个里程碑。为什么这么说呢？因为在此以前的程序都是通过改变计算机的布线等来变更程序的。

计算机中主要的存储部件是内存和磁盘。磁盘中存储的程序，必须要加载到内存后才能运行。在磁盘中保存的原始程序是无法直接运行的。这是因为，负责解析和运行程序内容的 CPU，需要通过内部程

① 一般把输入装置、输出装置、存储器、运算器和控制器这 5 种部件设备称为计算机的 5 大部件。

序计数器来指定内存地址，然后才能读出程序[①]。即使 CPU 可以直接读出并运行磁盘中保存的程序，由于磁盘读取速度慢，程序的运行速度还是会降低。总之，存储在磁盘中的程序需要读入到内存后才能运行。在考虑内存和磁盘的关系之前，大家一定要了解这个前提（图 5-1）。

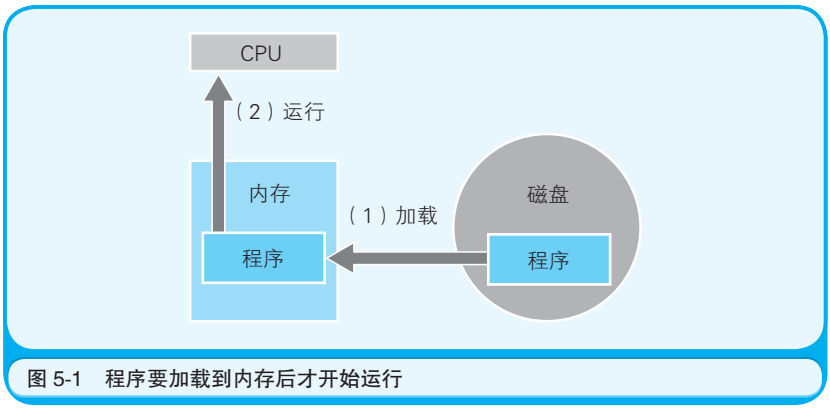

图 5-1　程序要加载到内存后才开始运行

在这个大前提的基础上，内存和磁盘之间存在着许多亲密关系。接下来我们逐一说明。

5.2　磁盘缓存加快了磁盘访问速度

作为体现内存和磁盘亲密关系的第一个示例，首先让我们来看一下磁盘缓存（disk cache）[②]。**磁盘缓存**指的是把从磁盘中读出的数据存储到内存空间中的方式。这样一来，当接下来需要读取同一数据时，就不用通过实际的磁盘，而是从磁盘缓存中把内容读出。使用磁盘缓存可以大大改善磁盘数据的访问速度（图 5-2）。

① 详情请参考第 1 章。

② 磁盘缓存的缓存（cache）是高速缓存、仓库的意思。

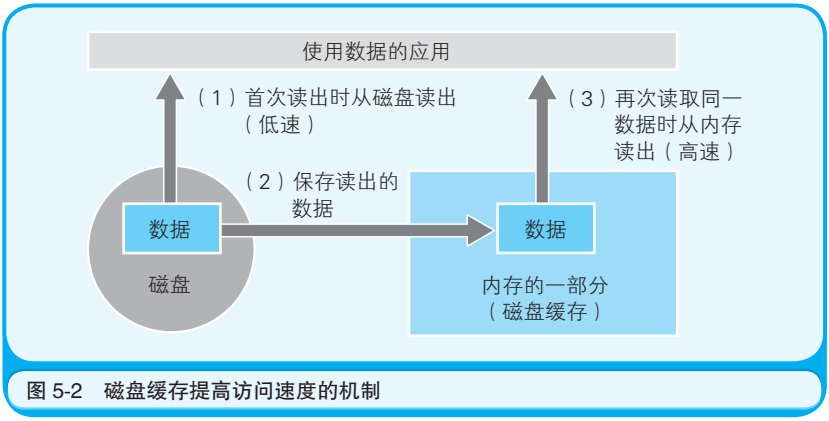

图 5-2 磁盘缓存提高访问速度的机制

Windows 提供了磁盘缓存机制作为操作系统。不过，对普通用户来说，磁盘缓存发挥显著效果的时代只延续到 Windows 95/98。现在，随着硬盘访问速度的大幅改善，磁盘缓存的效果也没有之前那么明显了。

把低速设备的数据保存在高速设备中，需要时可以直接将其从高速设备中读出，这种**缓存**的方式在其他情况下也会用到。其中的一个实例就是在 Web 浏览器中的使用。由于 Web 浏览器是通过网络来获取远程 Web 服务器的数据并将其显示出来的。因此，在显示较大的图片等文件时，会花费不少时间。于是，Web 浏览器就可以把获取的数据暂时保存在磁盘中，然后在需要时再显示磁盘中的数据。也就是说，把低速的网络数据保存到相对高速的磁盘中。

5.3 虚拟内存把磁盘作为部分内存来使用

接下来就让我们来看一下体现内存和磁盘亲密关系的第二个示例，即虚拟内存（virtual memory）。**虚拟内存**是指把磁盘的一部分作为假想的内存来使用。这与磁盘缓存是假想的磁盘（实际上是内存）相对，虚拟内存是假想的内存（实际上是磁盘）。