图解数据结构与算法

汪建 著

中国工信出版集团
人民邮电出版社
POSTS & TELECOM PRESS

异步图书
www.epubit.com

图解数据结构与算法

汪建 著

人民邮电出版社
北京

图书在版编目（CIP）数据

图解数据结构与算法 / 汪建著. -- 北京 : 人民邮电出版社, 2020.2
ISBN 978-7-115-49828-1

Ⅰ. ①图… Ⅱ. ①汪… Ⅲ. ①数据结构②算法分析
Ⅳ. ①TP311.12

中国版本图书馆CIP数据核字(2019)第267848号

内 容 提 要

这是一本“少字多图”、以图描述原理、形象且易于理解的数据结构与算法图书。全书共分为 7 章，首先介绍了一些基础的数据结构，包括数组、链表、栈和队列等；然后通过例子来讲解递归和动态规划的算法思想；接着对树进行了讲解，包括二叉树、二叉搜索树、AVL 树、红黑树、2-3 树、B 树以及 Trie 树等不同用途的树；在树的基础上讲解了堆，包括二叉堆、二项堆和斐波那契堆三种堆结构；还讲解了图结构，主要包括图的表示方式、图的遍历、图的最短路径以及最小生成树；最后讲解了比较排序和非比较排序，其中，比较排序包括选择排序、冒泡排序、插入排序、快速排序、希尔排序、合并排序和堆排序等，而非比较排序则包括计数排序、基数排序和桶排序等。

本书适合对数据结构和算法感兴趣并且想要通过一种轻松的方式学习和掌握数据结构与算法的读者阅读。无论他们是否有编程基础，均可看懂本书。

◆ 著　　汪　建
责任编辑　傅道坤
责任印制　王　郁　焦志炜

◆ 人民邮电出版社出版发行　　北京市丰台区成寿寺路 11 号
邮编　100164　　电子邮件　315@ptpress.com.cn
网址　http://www.ptpress.com.cn
天津画中画印刷有限公司印刷

◆ 开本：800×1000　1/16
印张：17.25
字数：341 千字　　2020 年 2 月第 1 版
印数：1－2 500 册　　2020 年 2 月天津第 1 次印刷

定价：89.00 元

读者服务热线：(010)81055410　印装质量热线：(010)81055316
反盗版热线：(010)81055315
广告经营许可证：京东工商广登字 20170147 号

作者简介

汪建（seaboat），毕业于广东工业大学光信息科学与技术专业，毕业后从事各类业务系统、中间件、基础架构和人工智能系统等方向的研发工作，目前致力于用 AI 来提升企业业务系统效率并节约人力成本。擅长工程算法、人工智能算法、自然语言处理、架构、分布式、高并发、大数据和搜索引擎等方面的技术，大多数编程语言都会使用，但更擅长 Java、Python 和 C++。平时喜欢看书、写作和运动，擅长篮球、跑步、游泳、健身和羽毛球等运动项目。崇尚开源，崇尚技术自由，更崇尚思想自由。

个人博客为 blog.csdn.net/wangyangzhizhou。大家也可以扫描下方二维码或在微信中搜索“远洋号”关注作者的个人公众号。

致谢

首先，感谢读者，您的阅读让本书变得更有价值。

其次，感谢在本书编写过程中帮助过我的人，特别是我的好兄弟“牛哥”，他对本书进行了详细认真的审阅，进而修正了一些错误并提出了很多宝贵的意见；感谢公司提供的平台让我得到了很多学习和成长的机会；感谢人民邮电出版社的傅道坤编辑一直以来给我提供的帮助和机会，并对本书提出了专业的意见。

再次，感谢旧金山大学的 David Galles 教授提供的 visualization 开源项目，该项目基于 FreeBSD 许可协议，本书很多示例都是基于该开源项目进行二次开发的。

最后，感谢一直鼓励、支持我的家人，他们让我的世界拥有更多的色彩。我的第一本书给了我帅气的儿子，而这本书给我可爱的女儿。

前言

Linux 内核之父林纳斯·托瓦兹（Linus Torvalds）曾说过："我认为一个优秀的程序员和一个不合格的程序员之间的区别在于，他是否将数据结构看得比代码更重要。不合格的程序员总是在关注代码，而优秀的程序员则会更看重数据结构。"Pascal 之父及图灵奖获得者尼古拉斯·沃斯（Niklaus Wirth）也曾提出了一个著名的公式：算法 + 数据结构 = 程序。

由此可见，数据结构和算法才是程序的核心。对于程序而言，数据结构和算法都是缺一不可的，一个具有合适数据结构和优异算法的程序就像是一个艺术品。虽然数据结构和算法都无比重要，但现实情况却是很多工作了三五年的程序员并不重视数据结构和算法，他们所实现的程序往往运行效率低且难以维护。

随着面向对象思想的兴起，面向对象的编程语言将众多常用的数据结构和算法封装成了各种对象，比如不同种类的列表、集合、队列、树等。那么它们到底是如何实现的呢？实现的原理是什么？可能多数程序员都会回答："不知道，反正编程语言已经提供了相关类，直接使用就行了。"对于编程语言，封装常用的数据结构和算法固然有很多好处，比如降低了使用成本，也大大降低了编码出错的概率。但封装同样会导致黑盒现象的产生，造成很多人在并不知道所用对象的具体实现原理的情况下，胡乱使用。

那么，数据结构是什么？算法又是什么？根据维基百科的定义，数据结构是一种数据组织、管理和存储的格式，它能提供有效的数据访问和修改操作。更准确地说，数据结构就是数据的集合，描述了各数据之间的关系，并提供了对这些数据的各种操作。而算法是一组定义清晰的指令集合，用于解决某类问题或执行某种运算任务。算法应该在有限的空间和时间内进行表达，其运行从初始状态和初始输入开始，经过一系列有限而定义清晰的指令操作后，最终产生输出并终止于某个状态。

尽管目前已经存在几本经典的算法书籍，但这几本书籍都很厚，而且内容和要点相当多。对于不是专门写算法的程序员来说，他们更需要通过一种形象且更加容易理解的方式来学习数据结构和算法。对于常见的数据结构和算法的核心思想，程序员更应该从感性的角度来理解把握，从而能够在不同的场景中知道要使用怎样的数据结构和算法。作者希望不管是刚入行的程序员还是经验丰富的工程师，都能轻

松领悟常见的数据结构和算法的精髓及思想。这也是本书的写作意图。

本书具备如下特点。

- 本书不与任何具体的编程语言绑定，而是注重用图解的方式来描述数据结构和算法的思想。所以不管是前端程序员还是后端程序员，不管使用的编程语言是 JavaScript、Java、C++ 还是 Python，都能够轻松阅读本书。
- 本书以读者能够轻松地从感性的角度来理解并掌握常用的数据结构和算法为目标，摒弃繁杂的学术式陈述，以通俗易懂的方式进行讲解，跟着执行示意图便能理解核心思想。
- 本书采用大量图解，对讲解的每个数据结构和算法的执行过程都给出了详细的步骤图，使读者能轻松理解各类数据结构和算法的思想。
- 本书所讲解的都是常见的数据结构和算法，能够帮助读者在实际项目开发中理解相关的实现原理。
- 本书脉络结构比较清晰，由简单到复杂，循序渐进。各知识点的连贯性比较强，对于基础知识也有补充说明，便于读者阅读。

组织结构

本书通篇紧扣“少字多图”的写作方针，以图来描述与数据结构和算法相关的机制原理，目的是让读者能够轻松地从感性的角度来理解并掌握常用的数据结构和算法。全书共分为 7 章，具体介绍如下。

- **第 1 章　基础数据结构**，介绍了一些基础的数据结构，包括数组、链表、栈和队列等，其中数组包括一维数组、二维数组、三维及更高维数组，链表包括单向链表和双向链表，栈包括基于数组的栈和基于链表的栈，队列包括基于数组的队列和基于链表的队列。
- **第 2 章　递归与动态规划**，介绍了递归和动态规划的算法思想。递归思想主要以阶乘和斐波那契数列为例进行讲解，动态规划思想则主要以斐波那契数列和最长公共子序列为例进行讲解。
- **第 3 章　树**，介绍了树这种常见的数据结构。根据不同的用途，树衍生出了多种不同的结构，包括二叉树、二叉搜索树、AVL 树、红黑树、2-3 树、B 树以及 Trie 树等，其中分别介绍了这些树的结构、性质及与其相关的主要操作。
- **第 4 章　堆**，介绍了堆数据结构，主要包括二叉堆、二项堆和斐波那契堆等。除了介绍各种堆的结构和性质外，还介绍了堆的相关操作。
- **第 5 章　图**，介绍了图数据结构，主要包括图的表示方式、图的遍历、图

的最短路径以及最小生成树。在图的表示方式中，主要介绍了邻接表和邻接矩阵两种方式；在图的遍历中，主要介绍了广度优先搜索和深度优先搜索；在图的最短路径中，主要介绍了 Dijkstra 和 Floyd 两种算法；在最小生成树中，主要介绍了 Prim 和 Kruskal 两种算法。

- **第 6 章　比较排序**，介绍了多种常见的基于比较的排序算法，包括选择排序、冒泡排序、插入排序、快速排序、希尔排序、合并排序和堆排序等，其中分别介绍了它们的排序要点、排序性能、排序过程。
- **第 7 章　非比较排序**，介绍了三种常见的非比较排序算法，包括计数排序、基数排序和桶排序，分别介绍了它们的排序要点、排序性能以及排序过程。其中，计数排序主要介绍了考虑稳定性的情况和不考虑稳定性的情况，基数排序主要介绍了基于计数排序的实现、基于桶排序的实现和 MSD 排序方式等。

读者对象

- 假如你对数据结构和算法感兴趣，那么这本书适合你。
- 假如你已经从事编程工作好几年，却没有熟练掌握常见的数据结构和算法，那么这本书适合你。
- 假如你想轻松学习并理解数据结构和算法，那么这本书适合你。
- 假如你想顺利解答面试中常见的算法题，那么这本书适合你。
- 本书侧重于用图讲解数据结构和算法的核心思想，并未与具体的某种编程语言绑定在一起，因此没有编程基础或者只会某一门编程语言的读者也可以顺利地阅读本书。
- 对数据结构和算法已经很熟悉了还有必要看吗？有必要。本书附带了常见的数据结构和算法的要点，同时通过图片形象地描述了它们的执行过程，能使读者快速抓住要点，可作为常用数据结构和算法的查阅手册。

反馈

在本书交稿时，我仍在担心是否遗漏了某些知识点，本书的内容是否详细完整，是否能让读者有所收获，是否会因为自己理解的偏差而误导读者。受写作水平和写作时间所限，本书中难免存在谬误，恳请读者批评指正。

读者可将任何意见及建议发送到邮箱 wyzz8888@foxmail.com，本书相关的勘误也会发布到我的博客上，欢迎读者通过邮件或博客与我交流。

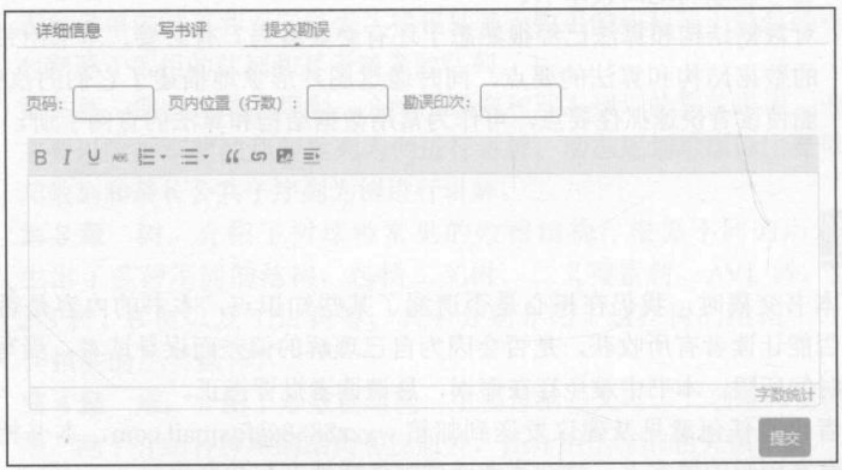
资源与支持

本书由异步社区出品，社区（https://www.epubit.com/）为您提供相关资源和后续服务。

提交勘误

作者和编辑尽最大努力来确保书中内容的准确性，但难免会存在疏漏。欢迎您将发现的问题反馈给我们，帮助我们提升图书的质量。

当您发现错误时，请登录异步社区，按书名搜索，进入本书页面，单击“提交勘误”，输入勘误信息，单击“提交”按钮即可。本书的作者和编辑会对您提交的勘误进行审核，确认并接受后，您将获赠异步社区的100积分。积分可用于在异步社区兑换优惠券、样书或奖品。



扫码关注本书

扫描下方二维码，您将会在异步社区微信服务号中看到本书信息及相关的服务提示。

与我们联系

我们的联系邮箱是contact@epubit.com.cn。

如果您对本书有任何疑问或建议，请您发邮件给我们，并请在邮件标题中注明本书书名，以便我们更高效地做出反馈。

如果您有兴趣出版图书、录制教学视频，或者参与图书翻译、技术审校等工作，可以发邮件给我们；有意出版图书的作者也可以到异步社区在线提交投稿（直接访问www.epubit.com/selfpublish/submission即可）。

如果您所在的学校、培训机构或企业，想批量购买本书或异步社区出版的其他图书，也可以发邮件给我们。

如果您在网上发现有针对异步社区出品图书的各种形式的盗版行为，包括对图书全部或部分内容的非授权传播，请您将怀疑有侵权行为的链接发邮件给我们。您的这一举动是对作者权益的保护，也是我们持续为您提供有价值的内容的动力之源。

关于异步社区和异步图书

“异步社区”是人民邮电出版社旗下IT专业图书社区，致力于出版精品IT技术图书和相关学习产品，为作译者提供优质出版服务。异步社区创办于2015年8月，提供大量精品IT技术图书和电子书，以及高品质技术文章和视频课程。更多详情请访问异步社区官网https://www.epubit.com。

“异步图书”是由异步社区编辑团队策划出版的精品IT专业图书的品牌，依托于人民邮电出版社近30年的计算机图书出版积累和专业编辑团队，相关图书在封面上印有异步图书的LOGO。异步图书的出版领域包括软件开发、大数据、AI、测试、前端、网络技术等。

异步社区

微信服务号

目录

第1章 基础数据结构……1

1.1 数组……1

1.1.1 一维数组……1

1.1.2 二维数组……2

1.1.3 三维及更高维数组……3

1.2 链表……4

1.2.1 单向链表……5

1.2.2 双向链表……14

1.3 栈……19

1.3.1 基于数组的栈……19

1.3.2 基于链表的栈……21

1.4 队列……23

1.4.1 基于数组的队列……23

1.4.2 基于链表的队列……26

第2章 递归与动态规划……28

2.1 递归……28

2.1.1 阶乘……28

2.1.2 斐波那契数列……30

2.2 动态规划……32

2.2.1 斐波那契数列……33

2.2.2 最长公共子序列……36

第3章 树……45

3.1 二叉树……46

3.1.1 完全二叉树……47

3.1.2 满二叉树……48

3.1.3 平衡二叉树……48

3.2 二叉搜索树……49

3.2.1 性质……49

3.2.2 插入操作……50

3.2.3 插入顺序性……51

3.2.4 查找操作……53

3.2.5 中序前驱节点……54

3.2.6 中序后继节点……56

3.2.7 删除操作……58

3.3 AVL 树……61

3.3.1 性质……61

3.3.2 二叉搜索树的平衡……62

3.3.3 为什么要旋转……63

3.3.4 插入类型及旋转……63

3.3.5 插入操作……67

3.3.6 查找操作……68

3.3.7 删除操作……69

3.4 红黑树……71

3.4.1 性质……71

3.4.2 旋转和变色……72
3.4.3 插入操作……72
3.4.4 查找操作……79
3.4.5 删除操作……80
3.5 2-3 树……86
3.5.1 性质……86
3.5.2 插入操作……87
3.5.3 查找操作……89
3.5.4 删除操作……91
3.6 B 树……98
3.6.1 性质……98
3.6.2 插入操作……99
3.6.3 查找操作……102
3.6.4 删除操作……102
3.7 Trie 树……109
3.7.1 性质……109
3.7.2 插入操作……109
3.7.3 查找操作……111
3.7.4 删除操作……113
第 4 章 堆……117
4.1 二叉堆……117
4.1.1 二叉堆的性质……117
4.1.2 二叉堆的实现……118
4.1.3 二叉堆的作用……118
4.1.4 插入操作……118
4.1.5 删除操作……121
4.1.6 构建操作……123
4.2 二项堆……125
4.2.1 结构与性质……125
4.2.2 插入与合并……126
4.2.3 查找最大（小）值……129
4.2.4 删除最大（小）值……131
4.3 斐波那契堆……133
4.3.1 结构及性质……133
4.3.2 斐波那契数列……134
4.3.3 插入操作……135
4.3.4 获取最小节点……136
4.3.5 合并两个斐波那契堆……136
4.3.6 删除最小键值节点……136
4.3.7 减小节点键值……139
4.3.8 删除节点……142
第 5 章 图……143
5.1 图的表示方式……144
5.2 图的遍历……145
5.2.1 广度优先搜索……146
5.2.2 深度优先搜索……151
5.3 图的最短路径……158
5.3.1 Dijkstra 算法……158
5.3.2 Floyd 算法……166
5.4 最小生成树……177
5.4.1 Prim 算法……178
5.4.2 并查集……185
5.4.3 Kruskal 算法……190
第 6 章 比较排序……196
6.1 选择排序……196
6.1.1 排序要点……196
6.1.2 排序性能……197
6.1.3 排序过程……197
6.1.4 稳定性……199
6.2 冒泡排序……200

6.2.1 排序要点 …… 201
6.2.2 排序性能 …… 201
6.2.3 排序过程 …… 201
6.3 插入排序 …… 203
6.3.1 排序要点 …… 203
6.3.2 排序性能 …… 204
6.3.3 排序过程 …… 204
6.4 快速排序 …… 207
6.4.1 排序要点 …… 207
6.4.2 排序性能 …… 207
6.4.3 排序过程 …… 208
6.5 希尔排序 …… 213
6.5.1 排序要点 …… 214
6.5.2 排序性能 …… 214
6.5.3 排序过程 …… 214
6.6 合并排序 …… 219
6.6.1 排序要点 …… 220
6.6.2 排序性能 …… 220
6.6.3 排序过程 …… 220
6.7 堆排序 …… 230
6.7.1 排序要点 …… 230
6.7.2 排序性能 …… 231
6.7.3 排序过程 …… 231
第 7 章 非比较排序 …… 236
7.1 计数排序 …… 236
7.1.1 不考虑稳定性的情况 …… 236
7.1.2 考虑稳定性的情况 …… 239
7.1.3 计数排序的局限 …… 247
7.2 基数排序 …… 247
7.2.1 排序性能 …… 247
7.2.2 排序方式 …… 247
7.2.3 基于计数排序的实现 …… 248
7.2.4 基于桶的实现 …… 253
7.2.5 MSD 排序方式 …… 257
7.3 桶排序 …… 260
7.3.1 排序性能 …… 260
7.3.2 排序要点 …… 261
7.3.3 桶的区间 …… 261
7.3.4 排序过程 …… 261

第 1 章

基础数据结构

1.1 数组

在计算机科学中，数组是我们最熟悉也是最基础的一种数据结构。通常地，数组由有限个相同数据类型的元素按顺序排列组合而成。数组的数据是连续的，并且会设定上界和下界，其中的每个元素都有属于它们自己的索引值（也称下标），通过这些下标就能定位到元素。对于绝大多数编程语言来说，数组的索引都是从 0 开始，但不排除有的编程语言从 1 开始。

几乎所有程序都会使用数组结构，而且很多其他的数据结构也可以由数组来实现，比如栈、队列、列表等。

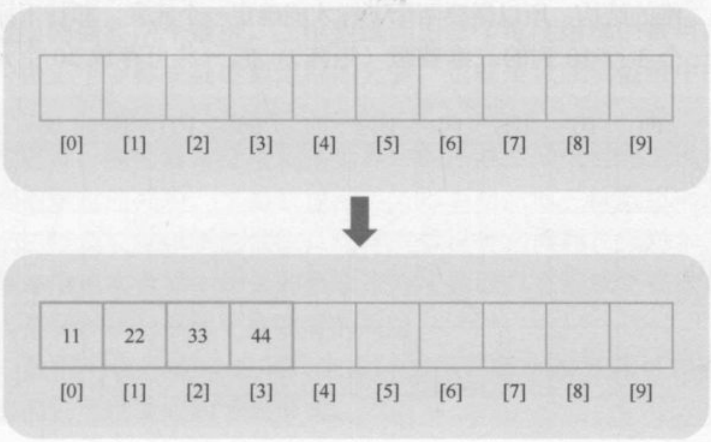
根据维度的不同，数组可以分为一维数组、二维数组、三维数组等，以此类推。

1.1.1 一维数组

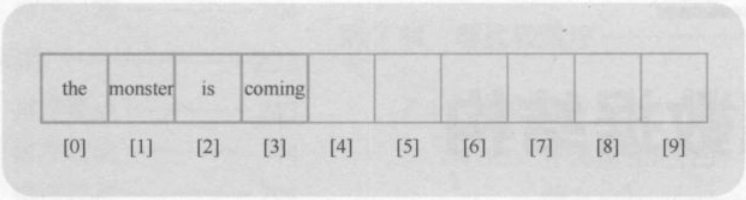
在各种维度的数组中，一维数组是最简单的数组结构，通常它也被称为线性数组。

① 创建一个长度为 10 的数组。

② 将 11、22、33、44 四个数字放到数组中。

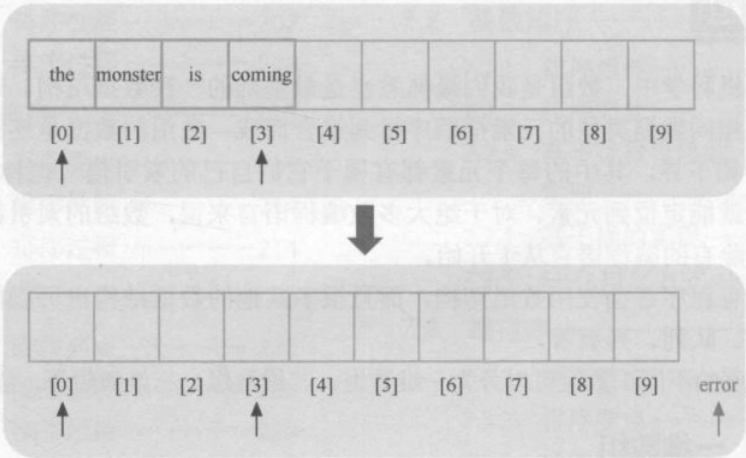


③ 再将“the”“monster”“is”“coming”四个字符串放到数组中，覆盖原来的前四个元素。



④ 获取数组下标为 0 和 3 所对应元素的字符串。

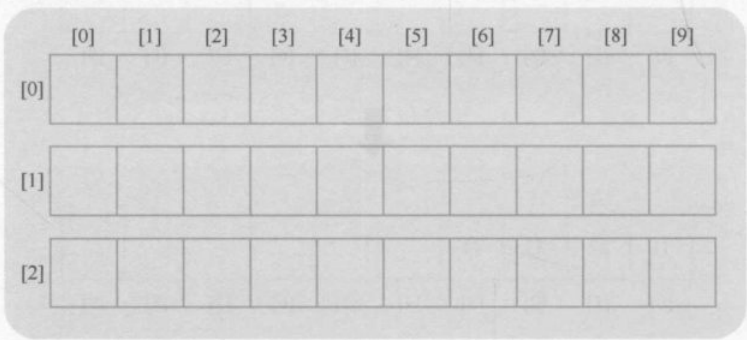
⑤ 数组大小为 10，则下标范围为 0～9，如果超出范围即越界，将会导致错误。



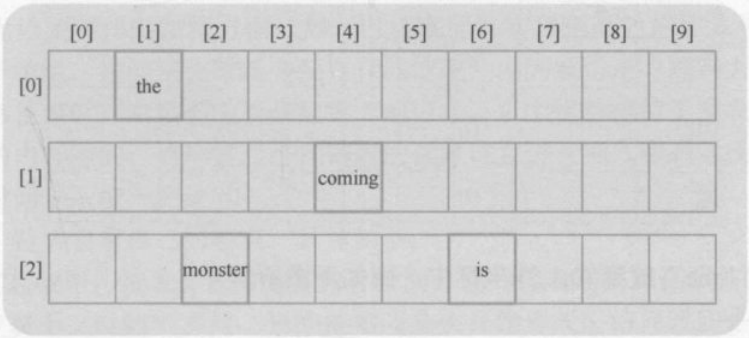
1.1.2 二维数组

由于数学领域中的矩阵可以通过二维数组来表示，因此二维数组也被称为矩阵。此外，因为它是二维的结构，所以需要两个下标才能确定一个元素，即行下标和列下标。

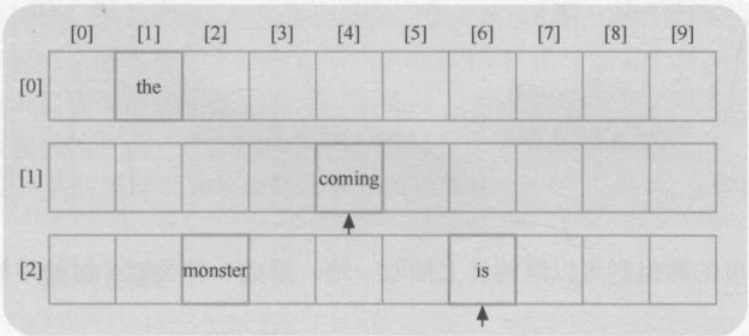
① 创建一个 3 行 10 列的二维数组（矩阵），它一共可存放 30 个元素。



② 将“the”“monster”“is”“coming”四个字符串分别放到数组的(0,1)、(2,2)、(2,6)、(1,4)四个坐标上。



③ 获取二维数组(2,6)、(1,4)坐标中所保存的字符串。

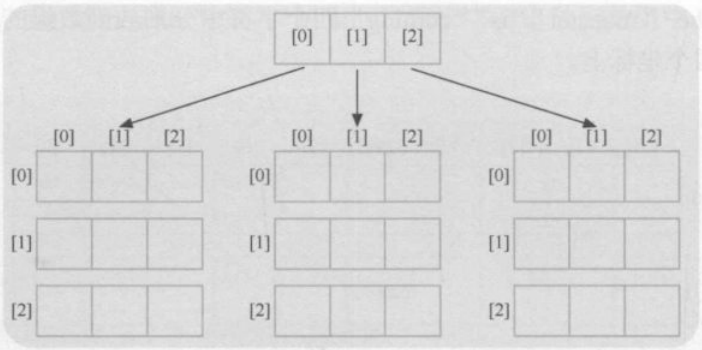


1.1.3 三维及更高维数组

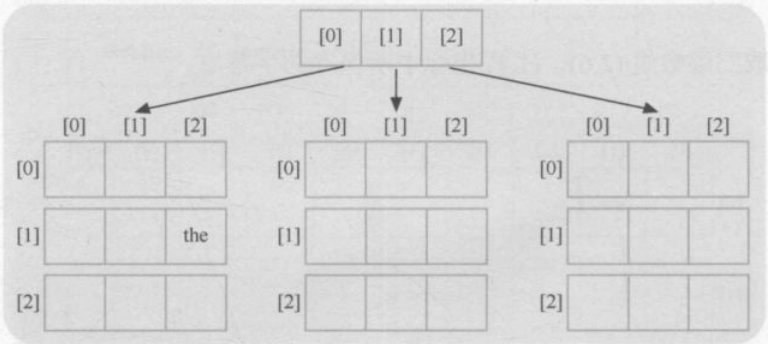
数组的维度数可以看成是获取一个元素所需的索引数，比如一维数组需要一个索引，二维数组则需要两个索引。三维数组即由三个维度组成的数组，它是最常见的多维数组，由三个下标去描述数组中的元素，也就是说获取数组中的一个元素需要三个索引。

按照正常思维，我们常常会用现实世界中的三维空间来对应三维数组以进行理解，比如一维数组对应列表，二维数组对应方形表格，而三维数组对应的是立体空间表格。对于三维及三维以下的数组，这样理解没什么问题，但对于更高维度的数组，我们就很难用现实世界中相关的概念来对应四维、五维或更高维数组，所以这样的思维方式不利于我们理解更高维度的数组。

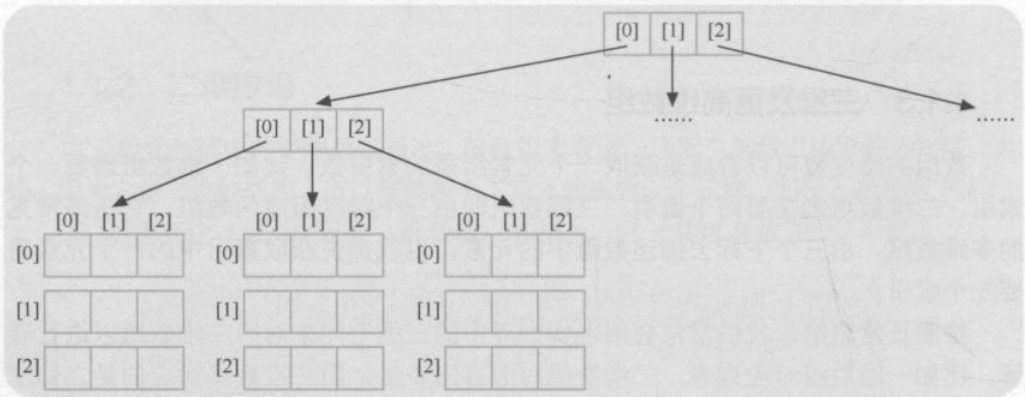
通常地，我们可以以索引的形式来理解数组，每个维度都可以看成是一层索引，三维的情况则可以看成如下的形式。



如果将“the”放到(0,1,2)坐标中，则如下图所示。



更高维度的数组则可以继续往上抽取一维，形成一种类似于树的结构。



1.2 链表

链表是一种线性的数据集合。在链表中，每个节点都指向后继或前驱节点，也