# 第一章 算法排序

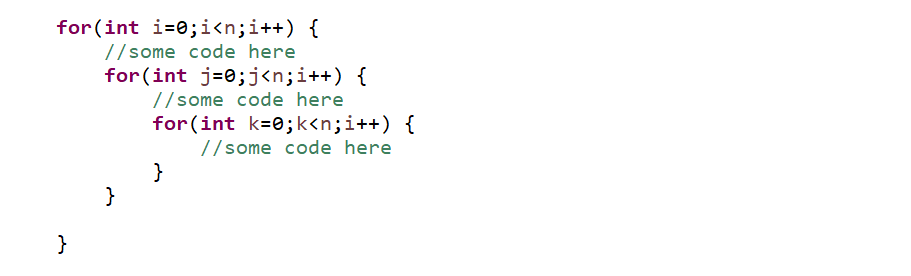
## 1.1算法基础

[算法复杂度](https://baike.baidu.com/item/%E7%AE%97%E6%B3%95%E5%A4%8D%E6%9D%82%E5%BA%A6" \t "https://baike.baidu.com/item/%E6%97%B6%E9%97%B4%E5%A4%8D%E6%9D%82%E6%80%A7/_blank)分为[时间复杂度](https://baike.baidu.com/item/%E6%97%B6%E9%97%B4%E5%A4%8D%E6%9D%82%E5%BA%A6" \t "https://baike.baidu.com/item/%E6%97%B6%E9%97%B4%E5%A4%8D%E6%9D%82%E6%80%A7/_blank)和[空间复杂度](https://baike.baidu.com/item/%E7%A9%BA%E9%97%B4%E5%A4%8D%E6%9D%82%E5%BA%A6" \t "https://baike.baidu.com/item/%E6%97%B6%E9%97%B4%E5%A4%8D%E6%9D%82%E6%80%A7/_blank)。其作用： [时间复杂度](https://baike.baidu.com/item/%E6%97%B6%E9%97%B4%E5%A4%8D%E6%9D%82%E5%BA%A6" \t "https://baike.baidu.com/item/%E6%97%B6%E9%97%B4%E5%A4%8D%E6%9D%82%E6%80%A7/_blank)是指执行算法所需要的计算工作量；而[空间复杂度](https://baike.baidu.com/item/%E7%A9%BA%E9%97%B4%E5%A4%8D%E6%9D%82%E5%BA%A6" \t "https://baike.baidu.com/item/%E6%97%B6%E9%97%B4%E5%A4%8D%E6%9D%82%E6%80%A7/_blank)是指执行这个算法所需要的[内存](https://baike.baidu.com/item/%E5%86%85%E5%AD%98" \t "https://baike.baidu.com/item/%E6%97%B6%E9%97%B4%E5%A4%8D%E6%9D%82%E6%80%A7/_blank)空间。（算法的复杂性体运行该算法时的计算机所需资源的多少上，[计算机](https://baike.baidu.com/item/%E8%AE%A1%E7%AE%97%E6%9C%BA" \t "https://baike.baidu.com/item/%E6%97%B6%E9%97%B4%E5%A4%8D%E6%9D%82%E6%80%A7/_blank)资源最重要的是时间和空间（即[寄存器](https://baike.baidu.com/item/%E5%AF%84%E5%AD%98%E5%99%A8" \t "https://baike.baidu.com/item/%E6%97%B6%E9%97%B4%E5%A4%8D%E6%9D%82%E6%80%A7/_blank)）资源，因此复杂度分为时间和空间复杂度。）

### 时间复杂度

一般的时间复杂度按照性能从差到好有这么几种：O (n3) 、O (n2) 、O(nlogn) 、O (n) 、O(logn) 、O(1)

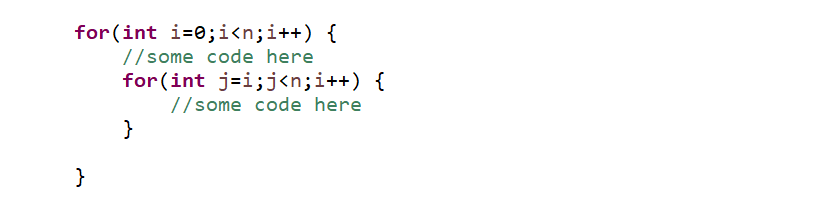
案例1：时间复杂度O (n3)



这段代码是个三重嵌套循环代码，且没重复循环都执行力完整的n编，n一般值算法的规模，很容易判断这段代码的时间复杂度为O (n3)

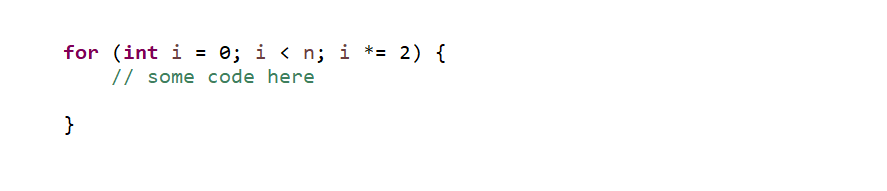
如果是两重嵌套循环代码，那么时间复杂度为O (n2)；如果为一重循环那么时间复杂度为O (n)。

案例2：时间复杂度O(nlogn)



在内层循环嵌套中的起始量是i，而随着每次外层嵌套循环i的增加，j的一层循环执行的次数将会越少。这种情况的时间复杂度为O(nlogn)。

案例3：时间复杂度O(logn)



这种情况称为对数阶，性能要优于O(n)。

案例4：时间复杂度O(1)



实际上，一个算法的执行时间是不可能通过计算得出的，必须到机器上真正执行才能知道，而且每次的运行时间不一样。但是我们没有必须讲每个算法都到机器上运行和测试，但是对于很多算法，我们通过简单的分析就能知道其性能的好坏，而且没有必要详细的写出来，所以时间复杂度的计算还是非常有用的。

### 空间复杂度

空间复杂度(Space Complexity)是对一个算法在运行过程中临时占用存储空间大小的量度，记做S(n)=O(f(n))。比如直接[插入排序](https://baike.baidu.com/item/%E6%8F%92%E5%85%A5%E6%8E%92%E5%BA%8F" \t "https://baike.baidu.com/item/%E7%A9%BA%E9%97%B4%E5%A4%8D%E6%9D%82%E5%BA%A6/_blank)的[时间复杂度](https://baike.baidu.com/item/%E6%97%B6%E9%97%B4%E5%A4%8D%E6%9D%82%E5%BA%A6/1894057" \t "https://baike.baidu.com/item/%E7%A9%BA%E9%97%B4%E5%A4%8D%E6%9D%82%E5%BA%A6/_blank)是O(n^2),空间复杂度是O(1) 。而一般的[递归](https://baike.baidu.com/item/%E9%80%92%E5%BD%92" \t "https://baike.baidu.com/item/%E7%A9%BA%E9%97%B4%E5%A4%8D%E6%9D%82%E5%BA%A6/_blank)算法就要有O(n)的空间复杂度了，因为每次递归都要存储返回信息。一个算法的优劣主要从算法的执行时间和所需要占用的存储空间两个方面[衡量](https://baike.baidu.com/item/%E8%A1%A1%E9%87%8F/483075" \t "https://baike.baidu.com/item/%E7%A9%BA%E9%97%B4%E5%A4%8D%E6%9D%82%E5%BA%A6/_blank)。

### 稳定性

什么是稳定性？在排序算法中，可能在一个列表中存在多个相等的元素，而经过排序只会，这些元素的相对次序保持不变，这时我们称这个算法是稳定的，若经过排序只会次序变了，那么就是不稳定的。

稳定性有什么用呢？如果算法是稳定的，那么第1个元素排序的结果可以被第2个相同值额元素排序所用，也就是说算法是稳定的，那么可以避免多余的比较。

在某些情况下，若是值一样的元素也要保持于原有的相对次序不变，那么这时必须用一个稳定的算法。

## 桶排序

排序在生活中随处可见，比如：买票，超市结账，银行办理业务等等。

案例1：某学期期末考试，老师把大家的分数排序，比如有5个学生分别考5、9、5、1、6（满分10分），从大到小的排序应该是9、6、5、5、1

## 1.4 算法的性能评价

## 1.5 一个算法实例

## 1.6 java程序的基本结构

## 1.7顺序结构

## 1.8 分支结构

## 1.9 循环结构

## 1.10 跳转结构

# 第二章

## 2.1 数据结构概述

## 2.2 线性结构

## 2.3 顺序表结构

## 2.4 链表结构

## 2.5栈结构

### 2.5.1　什么是栈结构

### 2.5.2　准备数据

### 2.5.3　初始化栈结构

### 2.5.4　判断空栈

### 2.5.5　判断满栈

### 2.5.6　清空栈

### 2.5.7　释放空间

### 2.5.8　入栈

### 2.5.9　出栈

### 2.5.10　读结点数据

### 2.5.11　栈结构操作实例

## 2.6　队列结构

### 2.6.1　什么是队列结构

### 2.6.2　准备数据

### 2.6.3　初始化队列结构

### 2.6.4　判断空队列

### 2.6.5　判断满队列

### 2.6.6　清空队列

### 2.6.7　释放空间

### 2.6.8　入队列

### 2.6.9　出队列

### 2.6.10　读结点数据

### 2.6.11　计算队列长度

### 2.6.12　队列结构操作实例

## 2.7　树结构

### 2.7.1　什么是树结构

### 2.7.2　树的基本概念

### 2.7.3　二叉树

### 2.7.4　准备数据

### 2.7.5　初始化二叉树

### 2.7.6　添加结点

### 2.7.7　查找结点

### 2.7.8　获取左子树

### 2.7.9　获取右子树

### 2.7.10　判断空树

### 2.7.11　计算二叉树深度

### 2.7.12　清空二叉树

### 2.7.13　显示结点数据

### 2.7.14　遍历二叉树

### 2.7.15　树结构操作实例

## 2.8　图结构

### 2.8.1　什么是图结构

### 2.8.2　图的基本概念

### 2.8.3　准备数据

### 2.8.4　创建图

### 2.8.5　清空图

### 2.8.6　显示图

### 2.8.7　遍历图

### 2.8.8　图结构操作实例

## 2.9　小结

# 第3章　基本算法思想

## 3.1　常用算法思想概述

## 3.2　穷举算法思想

### 3.2.1　穷举算法基本思想

### 3.2.2　穷举算法实例

## 3.3　递推算法思想

### 3.3.1　递推算法基本思想

### 3.3.2　递推算法实例

## 3.4　递归算法思想

### 3.4.1　递归算法基本思想

### 3.4.2　递归算法实例

## 3.5　分治算法思想

### 3.5.1　分治算法基本思想

### 3.5.2　分治算法实例

## 3.6　概率算法思想

### 3.6.1　概率算法基本思想

### 3.6.2　概率算法实例

## 3.7　小结

# 第4章　排序算法

## 4.1　排序算法概述

## 4.2　冒泡排序算法

### 4.2.1　冒泡排序算法

### 4.2.2　冒泡排序算法实例

## 4.3　选择排序算法

### 4.3.1　选择排序算法

### 4.3.2　选择排序算法实例

## 4.4　插入排序算法

### 4.4.1　插入排序算法

### 4.4.2　插入排序算法实例

## 4.5　Shell排序算法

### 4.5.1　Shell排序算法

### 4.5.2　Shell排序算法实例

## 4.6　快速排序算法

### 4.6.1　快速排序算法

### 4.6.2　快速排序算法实例

## 4.7　堆排序算法

### 4.7.1　堆排序算法

### 4.7.2　堆排序算法实例

## 4.8　合并排序算法

### 4.8.1　合并排序算法

### 4.8.2　合并排序算法实例

## 4.9　排序算法的效率

## 4.10　排序算法的其他应用

### 4.10.1　反序排序

### 4.10.2　字符串数组的排序

### 4.10.3　字符串的排序

## 4.11　小结

# 第5章　查找算法

## 5.1　查找算法概述

## 5.2　顺序查找

### 5.2.1　顺序查找算法

### 5.2.2　顺序查找操作实例

## 5.3　折半查找

### 5.3.1　折半查找算法

### 5.3.2　折半查找操作实例

## 5.4　数据结构中的查找算法

### 5.4.1　顺序表结构中的查找算法

### 5.4.2　链表结构中的查找算法

### 5.4.3　树结构中的查找算法

### 5.4.4　图结构中的查找算法

5**.5　小结**

# 第6章　基本数学问题

## 6.1　判断闰年

## 6.2　多项式计算

### 6.2.1　一维多项式求值

### 6.2.2　二维多项式求值

### 6.2.3　多项式乘法

### 6.2.4　多项式除法

## 6.3　随机数生成算法

### 6.3.1　Java语言中的随机方法

### 6.3.2　[0，1]之间均匀分布的随机数算法

### 6.3.3　产生任意范围的随机数

### 6.3.4　[m，n]之间均匀分布的随机整数算法

### 6.3.5　正态分布的随机数生成算法

## 6.4　复数运算

### 6.4.1　简单的复数运算

### 6.4.2　复数的幂运算

### 6.4.3　复指数运算

### 6.4.4　复对数运算

### 6.4.5　复正弦运算

### 6.4.6　复余弦运算

## 6.5　阶乘

### 6.5.1　使用循环来计算阶乘

### 6.5.2　使用递归来计算阶乘

## 6.6　计算π的近似值

### 6.6.1　割圆术

### 6.6.2　蒙特卡罗算法

### 6.6.3　级数公式

## 6.7　矩阵运算

### 6.7.1　矩阵加法

### 6.7.2　矩阵减法

### 6.7.3　矩阵乘法

## 6.8　方程求解

### 6.8.1　线性方程求解——高斯消元法

### 6.8.2　非线性方程求解——二分法

### 6.8.3　非线性方程求解——牛顿迭代法

### 6.9　小结

# 第7章　数据结构问题

## 7.1　动态数组排序

### 7.1.1　动态数组的存储和排序

### 7.1.2　动态数组排序实例

## 7.2　约瑟夫环

### 7.2.1　简单约瑟夫环算法

### 7.2.2　简单约瑟夫环求解

### 7.2.3　复杂约瑟夫环算法

### 7.2.4　复杂约瑟夫环求解

## 7.3　城市之间的最短总距离

### 7.3.1　最短总距离算法

### 7.3.2　最短总距离求解

## 7.4　最短路径

### 7.4.1　最短路径算法

### 7.4.2　最短路径求解

### 7.5　括号匹配

### 7.5.1　括号匹配算法

### 7.5.2　括号匹配求解

## 7.6　小结

# 第8章　数论问题

## 8.1　数论概述

### 8.1.1　数论概述

### 8.1.2　数论的分类

### 8.1.3　初等数论

### 8.1.4　本章用到的基本概念

## 8.2　完全数

### 8.2.1　什么是完全数

### 8.2.2　计算完全数算法

## 8.3　亲密数

### 8.3.1　什么是亲密数

### 8.3.2　计算亲密数算法

## 8.4　水仙花数

### 8.4.1　什么是水仙花数

### 8.4.2　计算水仙花数算法

## 8.5　自守数

### 8.5.1　什么是自守数

### 8.5.2　计算自守数算法

## 8.6　最大公约数

### 8.6.1　计算最大公约数算法——辗转相除法

### 8.6.2　计算最大公约数算法——Stein算法

### 8.6.3　计算最大公约数示例

## 8.7　最小公倍数

## 8.8　素数

### 8.8.1　什么是素数

### 8.8.2　计算素数算法

## 8.9　回文素数

### 8.9.1　什么是回文素数

### 8.9.2　计算回文素数算法

## 8.10　平方回文数

### 8.10.1　什么是平方回文数

### 8.10.2　计算平方回文数算法

## 8.11　分解质因数

8.12　小结

# 第9章　算法经典趣题

## 9.1　百钱买百鸡

### 9.1.1　百钱买百鸡算法

### 9.1.2　百钱买百鸡求解

## 9.2　五家共井

### 9.2.1　五家共井算法

### 9.2.2　五家共井求解

## 9.3　鸡兔同笼

### 9.3.1　鸡兔同笼算法

### 9.3.2　鸡兔同笼求解

## 9.4　猴子吃桃

9.4.1　猴子吃桃算法

### 9.4.2　猴子吃桃求解

## 9.5　舍罕王赏麦

### 9.5.1　舍罕王赏麦问题

### 9.5.2　舍罕王赏麦求解

## 9.6　汉诺塔

### 9.6.1　汉诺塔算法

### 9.6.2　汉诺塔求解

## 9.7　窃贼问题

### 9.7.1　窃贼问题算法

### 9.7.2　窃贼问题求解

## 9.8　马踏棋盘

### 9.8.1　马踏棋盘算法

### 9.8.2　马踏棋盘求解

## 9.9　八皇后问题

### 9.9.1　八皇后问题算法

### 9.9.2　八皇后问题求解

## 9.10　寻找假银币

### 9.10.1　寻找假银币算法

### 9.10.2　寻找假银币求解

## 9.11　青蛙过河

### 9.11.1　青蛙过河算法

### 9.11.2　青蛙过河求解

## 9.12　三色旗

### 9.12.1　三色旗算法

### 9.12.2　三色旗求解

## 9.13　渔夫捕鱼

### 9.13.1　渔夫捕鱼算法

### 9.13.2　渔夫捕鱼求解

## 9.14　爱因斯坦的阶梯

### 9.14.1　爱因斯坦的阶梯算法

### 9.14.2　爱因斯坦的阶梯求解

## 9.15　兔子产仔

### 9.15.1　兔子产仔算法

### 9.15.2　兔子产仔求解

## 9.16　常胜将军

### 9.16.1　常胜将军算法

### 9.16.2　常胜将军求解

## 9.17　新郎和新娘

### 9.17.1　新郎和新娘算法

### 9.17.2　新郎和新娘求解

## 9.18　三色球

### 9.18.1　三色球算法

### 9.18.2　三色球求解

### 9.19　小结

# 第10章　游戏中的算法

## 10.1　洗扑克牌算法

### 10.1.1　洗扑克牌算法

### 10.1.2　洗扑克牌实例

## 10.2　取火柴游戏算法

### 10.2.1　取火柴游戏算法

### 10.2.2　取火柴游戏实例

## 10.3　十点半算法

### 10.3.1　十点半算法

### 10.3.2　十点半游戏实例

## 10.4　生命游戏

### 10.4.1　生命游戏的原理

### 10.4.2　生命游戏的算法

### 10.4.3　生命游戏实例

10.5　小结

# 第11章　简单Java上机面试题

## 11.1　打印九九乘法口诀表

## 11.2　获得任意一个时间的下一天的时间

## 11.3　将某个时间以固定格式转化成字符串

## 11.4　怎样截取字符串

## 11.5　怎样实现元素互换

## 11.6　怎样实现元素排序

## 11.7　怎样实现Singleton模式编程

## 11.8　怎样实现金额转换

## 11.9　如何判断回文数字

11.10　小结

# 第12章　逻辑推理类面试题

## 12.1　脑筋急转弯

### 12.1.1　中国有多少辆汽车

### 12.1.2　下水道的盖子为什么是圆形的

### 12.1.3　分蛋糕

## 12.2　逻辑推理

### 12.2.1　哪个开关控制哪盏灯

### 12.2.2　戴帽子

### 12.2.3　海盗分金

### 12.2.4　罪犯认罪

### 12.2.5　找出质量不相同的球

### 12.2.6　有多少人及格

### 12.2.7　他说的是真话吗

## 12.3　计算推理

### 12.3.1　倒水问题

### 12.3.2　骗子购物

### 12.3.3　求最大的连续组合值（华为校园招聘笔试题）

### 12.3.4　洗扑克牌（乱数排列）

### 12.3.5　字符移动（金山笔试题）

12.4　小结

# 第13章　数学能力测试

## 13.1　100盏灯

### 13.2　用一笔画出经过9个点的4条直线

### 13.3　时针、分针和秒针重合问题

### 13.4　怎样拿到第100号球

### 13.5　烧绳计时