

## Java仗剑走天涯

专注Java开发，也熟悉前端开发

RSS订阅

### 原 漫画说算法--动态规划算法二（绝对通俗易懂，非常棒）

2017年06月13日 18:57:58

阅读数：2367

在上一篇漫画中，我们分析了一道动态规划相关的算法问题，并归纳出了问题的状态转移方程式。没看过上一篇的朋友可以点击下面的链接：

[漫画说算法-动态规划算法一（绝对通俗易懂，非常棒）](#)

首先，让我们简单回顾一下题目：

- 1 有一座高度是10级台阶的楼梯，从下往上走，每跨一步只能向上1级或者2级台阶。要求用程序来求出一共有多少种走法。

以动态规划的建模思路，我们归纳出的状态转移方程式如下：

**F(1) = 1;**

**F(2) = 2;**

**F(n) = F(n-1)+F(n-2) (n>=3)**

下面，继续我们的故事。

小灰，上次那个动态规划问题的求解，你思考的怎么样了？



[http://blog.csdn.net/baidu\\_37107022](http://blog.csdn.net/baidu_37107022)

那个啊... 我想了一下,很简单,  
不就是一个递归吗?



[http://blog.csdn.net/baidu\\_37107022](http://blog.csdn.net/baidu_37107022)

既然已经归纳出了  $F(N) = F(N-1) + F(N-2)$ ,  
又知道了递归结束的条件,我们就可以直接用递归的思路写程序。看,代码已经写好了:



[http://blog.csdn.net/baidu\\_37107022](http://blog.csdn.net/baidu_37107022)

方法一: 递归求解

```
int getClimbingWays(int n) {  
  
    if(n < 1) {  
        return 0;  
    }  
  
    if(n == 1) {  
        return 1;  
    }  
  
    if(n == 2) {  
        return 2;  
    }  
  
    return getWayNum(n-1) + getWayNum(n-2);  
}
```

[http://blog.csdn.net/baidu\\_37107022](http://blog.csdn.net/baidu_37107022)

由于代码比较简单, 这里就不做过多解释了。



OK, 这样的确可以计算出最终答案。可是, 你想过这个方法的时间复杂度吗?



[http://blog.csdn.net/baidu\\_37107022](http://blog.csdn.net/baidu_37107022)

啊呀, 这我倒是没想过...  
时间复杂度该怎么计算呢?



[http://blog.csdn.net/baidu\\_37107022](http://blog.csdn.net/baidu_37107022)

计算时间复杂度并不太难, 让我们来分析一下递归方法所走过的路径吧。

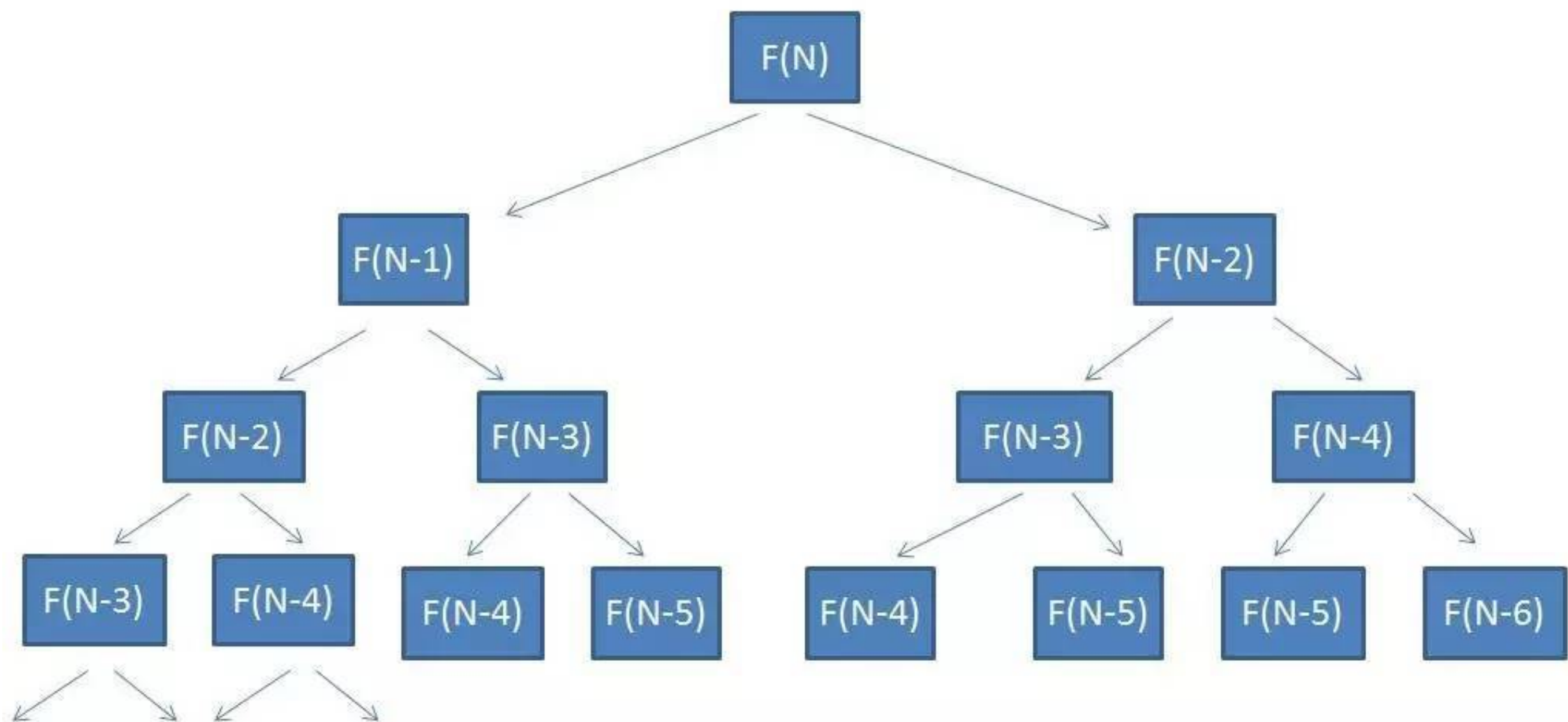


[http://blog.csdn.net/baidu\\_37107022](http://blog.csdn.net/baidu_37107022)

要计算出  $F(N)$ , 就要先得到  $F(N-1)$  和  $F(N-2)$  的值。要计算  $F(N-1)$ , 就要先得到  $F(N-2)$  和  $F(N-3)$  的值... 以此类推, 可以归纳成下面的图:



[http://blog.csdn.net/baidu\\_37107022](http://blog.csdn.net/baidu_37107022)



看起来还挺复杂，像是一棵二叉树。



没错，这就是一颗二叉树，树的节点个数就是我们的递归方法所需要计算的次数。





不难看出，这颗二叉树的高度是  $N$ ，  
节点个数接近  $2$  的  $N$  次方。所以方法  
的时间复杂度近似于  $O(2^N)$ 。



[http://blog.csdn.net/baidu\\_37107022](http://blog.csdn.net/baidu_37107022)

小灰，你想想有什么办法能  
优化一下呢？



[http://blog.csdn.net/baidu\\_37107022](http://blog.csdn.net/baidu_37107022)

这效率确实够低的，让我想想啊.....

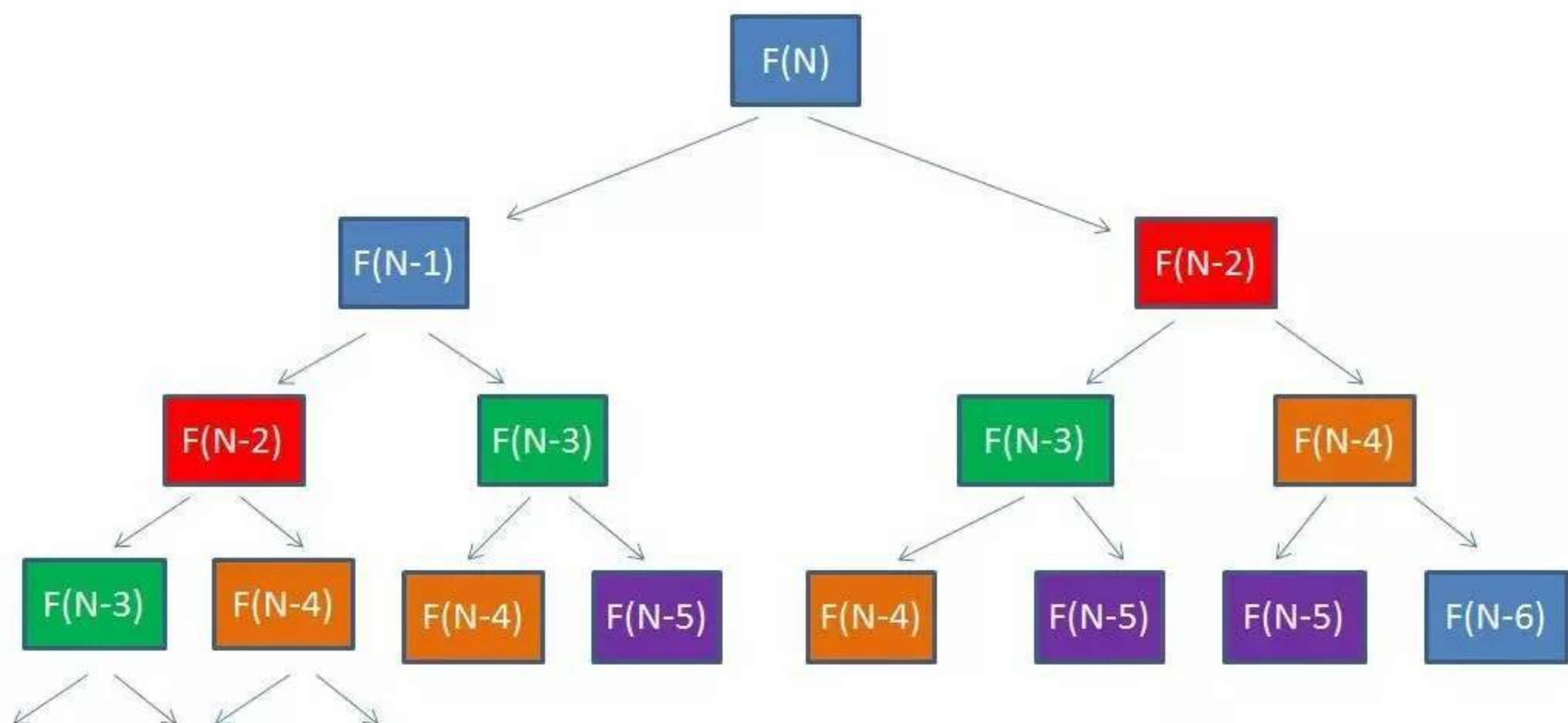


[http://blog.csdn.net/baidu\\_37107022](http://blog.csdn.net/baidu_37107022)

回顾一下刚才的递归图，我觉得  
有些相同的参数被重复计算了。  
越往下走，重复的越多。



[http://blog.csdn.net/baidu\\_37107022](http://blog.csdn.net/baidu_37107022)



.....[http://blog.csdn.net/baidu\\_37107022](http://blog.csdn.net/baidu_37107022)

如图所示，相同的颜色代表了方法被传入相同的参数。

观察的很好！对于这种重复计算的情况，应该怎么避免呢？



[http://blog.csdn.net/baidu\\_37107022](http://blog.csdn.net/baidu_37107022)

我知道了，用缓存！先创建一个哈希表，每次把不同参数的计算结果存入哈希。当遇到相同参数时，再从哈希表里取出，就不用重复计算了。



[http://blog.csdn.net/baidu\\_37107022](http://blog.csdn.net/baidu_37107022)



没错，这种暂存计算结果的方式有一个很贴切的名字，叫做【备忘录算法】。来，把刚才的思路实现一下吧。



[http://blog.csdn.net/baidu\\_37107022](http://blog.csdn.net/baidu_37107022)

方法二：备忘录算法

```
int getClimbingWays(int n, HashMap<Integer, Integer> map) {  
    if(n < 1){  
        return 0;  
    }  
  
    if(n == 1){  
        return 1;  
    }  
  
    if(n == 2){  
        return 2;  
    }  
  
    if(map.containsKey(n)) {  
        return map.get(n);  
    } else {  
        int value = getWayNum(n-1, map) + getWayNum(n-2, map);  
        map.put(n, value);  
        return value;  
    }  
}
```

[http://blog.csdn.net/baidu\\_37107022](http://blog.csdn.net/baidu_37107022)

在以上代码中，集合map是一个备忘录。当每次需要计算F(N)的时候，会首先从map中寻找匹配元素。如果map中存在，就直接返回结果，如果map中不存在，就计算出结果，存入备忘录中。

不错，这就是备忘录算法。你说说这个算法的时间复杂度和空间复杂度分别是多少？



[http://blog.csdn.net/baidu\\_37107022](http://blog.csdn.net/baidu_37107022)

这个容易算。从  $F(1)$  到  $F(N)$  一共有  $N$  个不同的输入，在哈希表里存了  $N-2$  个结果，所以时间复杂度和空间复杂度都  $O(N)$ 。



[http://blog.csdn.net/baidu\\_37107022](http://blog.csdn.net/baidu_37107022)

是的，现在的程序性能已经得到了明显优化，但这样还并不是真正的动态规划实现。



[http://blog.csdn.net/baidu\\_37107022](http://blog.csdn.net/baidu_37107022)

时间复杂度已经不能再小了。小灰，你想一想，咱们还能不能把空间复杂度进一步减小？



[http://blog.csdn.net/baidu\\_37107022](http://blog.csdn.net/baidu_37107022)

我的天，还要继续优化呀...  
让我想想.....



[http://blog.csdn.net/baidu\\_37107022](http://blog.csdn.net/baidu_37107022)



不行了，想不出来.....



[http://blog.csdn.net/baidu\\_37107022](http://blog.csdn.net/baidu_37107022)

哈哈，没关系。借用鲁迅先生的一句名言：  
我们不妨把思路逆转过来！



[http://blog.csdn.net/baidu\\_37107022](http://blog.csdn.net/baidu_37107022)



[http://blog.csdn.net/baidu\\_37107022](http://blog.csdn.net/baidu_37107022)

呸，鲁迅啥时候说过这句话。  
怎么把思路逆转呢？



[http://blog.csdn.net/baidu\\_37107022](http://blog.csdn.net/baidu_37107022)

想想看，我们一定要对  $F(N)$  自顶向下做递归运算吗？可不可以自底向下，用迭代的方式推导出结果？



[http://blog.csdn.net/baidu\\_37107022](http://blog.csdn.net/baidu_37107022)

什么自顶向上，自底向下的...  
听不明白.....



[http://blog.csdn.net/baidu\\_37107022](http://blog.csdn.net/baidu_37107022)

好吧，让我通过一张表格，来说明一下  $F(N)$  自底向上求解的过程。



[http://blog.csdn.net/baidu\\_37107022](http://blog.csdn.net/baidu_37107022)

台阶数	1	2	3	4	5	6	7	8	9
走法数	1	2							

[http://blog.csdn.net/baidu\\_37107022](http://blog.csdn.net/baidu_37107022)



看，表格的第一行代表了楼梯台阶的数目，第二行代表了若干级台阶对应的走法数。 $F(1)=1$ ,  $F(2)=2$ ，这是之前就已经明确过的结果。



[http://blog.csdn.net/baidu\\_37107022](http://blog.csdn.net/baidu_37107022)

台阶数	1	2	3	4	5	6	7	8	9
走法数	1	2	3						

[http://blog.csdn.net/baidu\\_37107022](http://blog.csdn.net/baidu_37107022)

第一次迭代，台阶数等于 3 时，走法数量是 3。这个结果怎么来的呢？是  $F(1)$ ,  $F(2)$  这两个结果相加得到的。所以  $F(3)$  只依赖于  $F(1)$  和  $F(2)$ 。



[http://blog.csdn.net/baidu\\_37107022](http://blog.csdn.net/baidu_37107022)

台阶数	1	2	3	4	5	6	7	8	9
走法数	1	2	3	5					

[http://blog.csdn.net/baidu\\_37107022](http://blog.csdn.net/baidu_37107022)

第二次迭代，台阶数等于 4 时，走法数量是 5。这是  $F(2)$ ,  $F(3)$  这两个结果相加得到的。所以  $F(4)$  只依赖于  $F(2)$  和  $F(3)$ 。



[http://blog.csdn.net/baidu\\_37107022](http://blog.csdn.net/baidu_37107022)

台阶数	1	2	3	4	5	6	7	8	9
走法数	1	2	3	5	8				

[http://blog.csdn.net/baidu\\_37107022](http://blog.csdn.net/baidu_37107022)

同理，在后续的迭代中， $F(5)$  只依赖于  $F(4)$ ,  $F(3)$ ； $F(6)$  只依赖于  $F(5)$ ,  $F(4)$ 。



[http://blog.csdn.net/baidu\\_37107022](http://blog.csdn.net/baidu_37107022)

由此可见，每一次迭代过程中，只要保留之前的两个状态，就可以推导出新的状态。而不需要像备忘录算法那样保留全部子状态。



[http://blog.csdn.net/baidu\\_37107022](http://blog.csdn.net/baidu_37107022)



这样才是真正的动态规划实现，  
让我们一起看看代码吧。



[http://blog.csdn.net/baidu\\_37107022](http://blog.csdn.net/baidu_37107022)

### 方法三：动态规划求解

```
int getClimbingWays(int n){  
    if(n < 1){  
        return 0;  
    }  
  
    if(n == 1){  
        return 1;  
    }  
  
    if(n == 2){  
        return 2;  
    }  
  
    int a = 1;  
    int b = 2;  
    int temp = 0;  
  
    for(int i=3; i<=n; i++){  
        temp = a + b;  
        a = b;  
        b = temp;  
    }  
  
    return temp;  
}
```

[http://blog.csdn.net/baidu\\_37107022](http://blog.csdn.net/baidu_37107022)

程序从 i=3 开始迭代，一直到 i=n 结束。每一次迭代，都会计算出多一级台阶的走法数量。迭代过程中只需保留两个临时变量a和b，分别代表了上一次和上上次迭代的结果。为了便于理解，我引入了temp变量。temp代表了当前迭代的结果值。

原来这才是动态规划的代码实现，  
看起来好简洁，厉害了我的哥！



[http://blog.csdn.net/baidu\\_37107022](http://blog.csdn.net/baidu_37107022)

嘿嘿，你看看这个方法的时间复  
杂度和空间复杂度是多少？



[http://blog.csdn.net/baidu\\_37107022](http://blog.csdn.net/baidu_37107022)

时间复杂度显然是  $O(N)$ ，由于只  
引入了两个或三个变量，所以空  
间复杂度只有  $O(1)$ ！



[http://blog.csdn.net/baidu\\_37107022](http://blog.csdn.net/baidu_37107022)

是的呢，这就是动态规划，利用简  
洁的自底向上的递推方式，实现了  
时间和空间上的最优化。



[http://blog.csdn.net/baidu\\_37107022](http://blog.csdn.net/baidu_37107022)



不过，这道上楼梯的题目仅仅是动态规划领域中最最简单的问题，因为它只有一个变化维度。还有许多问题远比这要复杂得多。



[http://blog.csdn.net/baidu\\_37107022](http://blog.csdn.net/baidu_37107022)

哎呀我滴乖乖，这还只是最简单的？



[http://blog.csdn.net/baidu\\_37107022](http://blog.csdn.net/baidu_37107022)

哈哈，你以为呢？下面给你出一道相对复杂的题目。弄懂了这个问题，才算是真正理解了动态规划。



[http://blog.csdn.net/baidu\\_37107022](http://blog.csdn.net/baidu_37107022)

---

## 题目二：国王和金矿

有一个国家发现了5座金矿，每座金矿的黄金储量不同，需要参与挖掘的工人数也不同，而工人的总数是1000人。要求用程序求解出，要想得到尽可能多的黄金，应该选择挖取哪几座金矿？



总共 1000 工人



天呐，光看着就脑壳子疼 ...

下次再仔细思考吧。



[http://blog.csdn.net/baidu\\_37107022](http://blog.csdn.net/baidu_37107022)

版权声明：本文为博主原创文章，未经博主允许不得转载。 [https://blog.csdn.net/baidu\\_37107022/article/details/73189125](https://blog.csdn.net/baidu_37107022/article/details/73189125)

文章标签：[动态规划](#) [算法](#) [动态规划算法](#) [动态规划逐次逼近](#) [动态规划法](#)

个人分类：[TCB-算法](#)

所属专栏：[漫画说算法](#)

想对作者说点什么？

我来说两句



王红伟 2018-05-16 10:24:27 #3楼

真是很棒那，感谢分享



Soleil-luo 2018-04-20 16:00:23 #2楼

想看 国王和金矿 算法的漫画解



开心乐酷 2018-04-11 08:52:51 #1楼

非常棒，感谢



## 漫画：什么是动态规划？（整合版）

题目：有一座高度是10级台阶的楼梯，从下往上走，每跨一步只能向上1级或者2级台阶。要求用程序来求出一共有多少种走法。比如，每次走1级台阶，一共走10步，这是其中一种走法。我们可以简写成 1,1,1,1...

 dgutliangxuan    2017-12-24 10:49:44    阅读数：291

## 漫画说算法--动态规划算法—（绝对通俗易懂，非常棒）

有一座高度是10级台阶的楼梯，从下往上走，每跨一步只能向上1级或者2级台阶。要求用程序来求出一共有多少种走法。...

 baidu\_37107022    2017-06-13 17:56:05    阅读数：3365

## 【长篇读后】跟着两只仓鼠学算法 漫画算法系列 - CSDN博客

感谢微信公众号“算法爱好者”,以及该漫画系列的出处“程序员小灰” 这里会长期小灰每一期的学习感悟总结,以及全文链接 算法系列 算法系列 漫画算法:最小栈的实现...

2018-2-21

## 【长篇读后】跟着两只仓鼠学算法 漫画算法系列 - CSDN博客

感谢微信公众号“算法爱好者”,以及该漫画系列的出处“程序员小灰” 这里会长期小灰每一期的学习感悟总结。 算法系列 最小栈实现判断2的乘方找出缺失的整数辗转相...

2018-6-10

## 【长篇读后】跟着两只仓鼠学算法 漫画算法系列 - CSDN博客

感谢微信公众号“算法爱好者”,以及该漫画系列的出处“程序员小灰” 这里会长期小灰每一期的学习感悟总结。 算法系列 最小栈实现判断2的乘方找出缺失的整数辗转相...

2018-6-10

## 【长篇读后】跟着两只仓鼠学算法 漫画算法系列 - CSDN博客

感谢微信公众号“算法爱好者”,以及该漫画系列的出处“程序员小灰” 这里会长期小灰每一期的学习感悟总结,以及全文链接 算法系列 算法系列 漫画算法:最小栈的实现...

2018-2-21

## 漫画说算法--动态规划算法二(绝对通俗易懂,非常棒) - CSDN博客

在上一篇漫画中,我们分析了一道动态规划相关的算法问题,并归纳出了问题的状态...感谢微信公众号“算法爱好者”,以及该漫画系列的出处“程序员小灰” 这里会长期...

2018-6-20

## 漫画算法:辗转相除法是什么鬼？ - CSDN博客

辗转相除法, 又名欧几里得算法(Euclidean algorithm),目的是求出两个正整数的最...感谢微信公众号“算法爱好者”,以及该漫画系列的出处“程序员小灰” 这里会长期...

2018-6-21

## 皱纹让你衰老加快，一招教你年轻20岁！

无量影业 · 顶新



## 女性得了静脉曲张变成蚯蚓腿怎么办？用这方法坚持3个月全恢复！

水英电器 · 顶新



## 【长篇读后】跟着两只仓鼠学算法 漫画算法系列

感谢微信公众号“算法爱好者”， 以及该漫画系列的出处“程序员小灰” 这里会长期小灰每一期的学习感悟总结。 算法系列 最小栈实现 判断2的乘方 找出缺失的整数 辗转相除法是什么鬼 Bitmap 算...

 nakiri\_arisu    2018-02-21 00:04:50    阅读数：416

漫画算法:辗转相除法是什么鬼？ - CSDN博客

辗转相除法, 又名欧几里得**算法**(Euclidean algorithm),目的是求出两个正整数的最...感谢微信公众号“**算法**爱好者”,以及该**漫画系列**的出处“程序员小灰” 这里会长期...

2018-6-7

漫画算法:什么是红黑树？ - CSDN博客

感谢微信公众号“**算法**爱好者”,以及该**漫画系列**的出处“程序员小灰” 这里会长期小灰每一期的学习感悟总结。...

2018-6-6

漫画**说算法--动态规划算法二**(绝对通俗易懂,非常棒) - CSDN博客

在上一篇**漫画**中,我们分析了一道动态规划相关的**算法**问题,并归纳出了问题的状态...感谢微信公众号“**算法**爱好者”,以及该**漫画系列**的出处“程序员小灰” 这里会长期...

2018-6-20

漫画算法:什么是红黑树？ - CSDN博客

感谢微信公众号“**算法**爱好者”,以及该**漫画系列**的出处“程序员小灰” 这里会长期小灰每一期的学习感悟总结。...

2018-6-6

漫画算法:什么是 B 树？ - CSDN博客

http://blog.jobbole.com/111757/?utm\_source=blog.jobbole.com&utm\_medium=relatedPosts 伯乐在线补充:本文...

2018-5-22

漫画算法:什么是 B+ 树？ - CSDN博客

https://mp.weixin.qq.com/s/QGepgMbHQ8JeRxVQwAwsxQ **漫画算法**:什么是 B+ 树? 2017-07-14 程序员的那些事 (点击上方公众号,可快速关注) 来源:伯乐专栏...

2018-6-3

漫画**算法**： 什么是红黑树？

（点击上方蓝字，快速关注我们） 来源：伯乐专栏作者/玻璃猫，微信公众号 - 程序员小灰 好文投稿， 请点击 → 这里了解详情 ...

 P5dEyT322JACS    2017-11-02 00:00:00    阅读数：1120

漫画**算法**： 什么是一致性哈希？

2017-07-18 **算法**爱好者 （点击上方公众号，可快速关注） 来源：伯乐专栏作者/玻璃猫，微信公众号 - 梦见（dreamsee321） 如有好文章投稿，请点...

 andyzhaojianhui    2017-07-20 18:43:06    阅读数：509

漫画:什么是HashMap？ - CSDN博客

《**漫画**:什么是MD5**算法**?》 《**漫画**:如何破解MD5**算法**?》 《**漫画**:什么是SHA**系列算法**?》 《**漫画**:什么是AES**算法**...

2018-6-6

undefined

漫画算法:辗转相除法是什么鬼？ - CSDN博客

辗转相除法, 又名欧几里得**算法**(Euclidean algorithm),目的是求出两个正整数的最...感谢微信公众号“**算法**爱好者”,以及该**漫画系列**的出处“程序员小灰” 这里会长期...

2018-6-7

漫画算法:辗转相除法是什么鬼？ - CSDN博客

辗转相除法, 又名欧几里得**算法**(Euclidean algorithm),目的是求出两个正整数的最...感谢微信公众号“**算法**爱好者”,以及该**漫画系列**的出处“程序员小灰” 这里会长期...

2018-6-21

动态规划**算法**之资源分配问题及其空间优化方案

资源分配问题：某厂根据计划安排，拟将n台相同的设备分配给m个车间，各车间获得这种设备后，可以为国家提供盈利Ci j(i台设备提供给j号车间将得到的利润，1≤i≤n，1≤j≤m)。问如何分配，才使国家...

 iamubbTing    2017-01-13 15:53:36    阅读数：7586



饭后吃块它，大量排出胆固醇、血糖再也不高，睡得安稳

海淘车 · 顶新



漫画算法:什么是 B 树? - CSDN博客

http://blog.jobbole.com/111757/?utm\_source=blog.jobbole.com&utm\_medium=relatedPosts 伯乐在线补充:本文...

2018-5-22

算法系列 - CSDN博客

【长篇读后】跟着两只仓鼠学算法 漫画算法系列 感谢微信公众号“算法爱好者”,以及该漫画系列的出处“程序员小灰” 这里会长期小灰每一期的学习感悟总结。 算法系列...

2018-2-28

皱纹让你衰老加快，一招教你年轻20岁！

无量影业 · 顶新



机器学习常用算法（1）最小二乘和k-means聚类

最近复习算法准备校招，顺便写一写算作补上以前欠的债。 1.最小二乘法 有一堆数据点（Xi，Yi），其中i从0到n，那么我现在用一个超平面去拟合这些数据点，这个超平面的方程形式？ whatever...

byplane 2016-08-21 16:28:09 阅读数：1114

动态规划--分段最小二乘法

#include using namespace std; #include const int c = 10; double Min2Method(double \*\*X,int s\_nu...

yuqinh 2017-03-23 22:23:09 阅读数：503

算法系列 - CSDN博客

感谢微信公众号“算法爱好者”,以及该漫画系列的出处“程序员小灰” 这里会长期小灰每一期的学习感悟总结。 算法系列 最小栈实现判断2的乘方找出缺失的整数辗转相...

2018-6-13

undefined

【BUG】交换两个元素的时候遇到的坑！

交换元素的方式无非有这么三种: //愚蠢且多空间的中间量法 temp = a; a = b; b = temp; //异或法 a = a^b;...

nakiri\_arisu 2018-02-26 19:12:20 阅读数：67

漫画算法：辗转相除法是什么鬼？

http://blog.jobbole.com/106315/ 小灰的思路十分简单。他使用暴力枚举的方法，试图寻找到一个合适的整数 i，看看这个整数能否被两个整型参数numberA和numberB...

z69183787 2017-03-21 15:16:33 阅读数：572

漫画：递归

从前有座山，山上有座庙，庙里住着一只老猫，老猫对小猫说，你可知道什么是递归吗?emmmmm....我们首先了解下递归的定义（来自百度百科）程序调用自身的编程技巧称为递归（ recursion）。递归做...

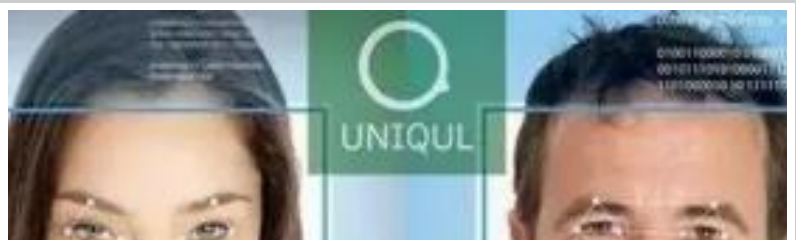
u013211506 2018-05-07 22:50:35 阅读数：46

教你彻底学会动态规划——入门篇

动态规划相信大家都知道，动态规划算法也是新手在刚接触算法设计时很苦恼的问题，有时候觉得难以理解，但是真正理解之后，就会觉得动态规划其实并没有想象中那么难。网上也有很多关于讲解动态规划的文章，大多都是叙...

baidu\_28312631 2015-08-11 13:26:41 阅读数：146612

几个经典的动态规划算法



## 动态规划算法经典案例

动态规划的关键点： 1、最优化原理，也就是最有子结构性质。这指的是一个最优化策略具有这样的性质，无论过去状态和决策如何，对前面的决策所形成的状态而言，余下的决策必须构成最优策略，简单来说就是一个最优化...

## 算法导论-----动态规划是什么

《**算法**导论》中并没有把动态规划的来龙去脉介绍清楚，网上很多讲解都是动态规划的数学模型，感觉没必要系统的学习数学的定义，把人搞晕了。本文更像是一篇科普，方便理解什么是动态规划。一、动态规划概述     动态规...

## 《面试--动态规划》 ---五种经典的**算法**问题

一 动态规划 动态规划问题是面试题中的热门话题，如果要求一个问题的最优解（通常是最大值或者最小值），而且该问题能够分解成若干个子问题，并且小问题之间也存在重叠的子问题，则考虑采用动态规划。使用动...

## 几个经典的动态规划的**算法**

列举几个动态规划经典的**算法**

## 白话**算法**之【动态规划入门】

动态规划入门 什么是动态规划？     动态规划(Dynamic Programming，所以我们简称动态规划为DP)是运筹学的一个分支，是求解决策过程(decision process)最优...

## 便宜云虚拟主机

便宜的虚拟主机一定是好的吗

百度广告



## 五个常用**算法**（一）： 动态规划

1.从01背包问题说起 有一堆宝石一共n个，现在你身上能装宝石的就只有一个背包，背包的容量为C。把n个宝石排成一排并编上号： 0,1,2,...,n-1。第i个宝石对应的体积和价值分别为V[i]和W[...

## 动态规划算法学习

笔试面试中经常会出现一些考察动态规划方面的题目，以前没有接触过，现在初学做个整理。 1. 什么是动态规划？     和分治法一样，动态规划（dynamicprogramming）是通过组...

## 漫画：递归

从前有座山，山上有座庙，庙里住着一只老猫，老猫对小猫说，你可知道什么是递归吗?emmmmm....我们首先了解下递归的定义（来自百度百科）程序调用自身的编程技巧称为递归（ recursion）。递归做...



# 教你彻底学会动态规划——入门篇

动态规划相信大家都知道，**动态规划算法**也是新手在刚接触**算法**设计时很苦恼的问题，有时候觉得难以理解，但是真正理解之后，就会觉得动态规划其实并没有想象中那么难。网上也有很多关于讲解动态规划的文章，大多都是叙...

 baidu\_28312631     2015-08-11 13:26:41     阅读数：146612

## 几个经典的**动态规划算法**

动态规划~ 背包问题 最大子数组和问题

 a20180825     2017-08-05 13:58:15     阅读数：1878

## 动态规划**算法**经典案例

动态规划的关键点： 1、最优化原理，也就是最有子结构性质。这指的是一个最优化策略具有这样的性质，无论过去状态和决策如何，对前面的决策所形成的状态而言，余下的决策必须构成最优策略，简单来说就是一个最优化...

 uestclr     2016-02-29 20:14:52     阅读数：34226

## **算法**导论-----动态规划是什么

《**算法**导论》中并没有把动态规划的来龙去脉介绍清楚，网上很多讲解都是动态规划的数学模型，感觉没必要系统的学习数学的定义，把人搞晕了。本文更像是一篇科普，方便理解什么是动态规划。一、动态规划概述     动态规...

 so\_geili     2016-12-14 21:41:20     阅读数：2238

## 《面试--动态规划》---五种经典的**算法**问题

一 动态规划 动态规划问题是面试题中的热门话题，如果要求一个问题的最优解（通常是最大值或者最小值），而且该问题能够分解成若干个子问题，并且小问题之间也存在重叠的子问题，则考虑采用动态规划。 使用动...

 tongxinhazha     2017-08-19 11:02:49     阅读数：10030

## 几个经典的动态规划的**算法**

列举几个动态规划经典的**算法**

 jpbj\_zb     2016-05-05 09:25:54     阅读数：6442

## 白话**算法**之【动态规划入门】

动态规划入门 什么是动态规划？     动态规划(Dynamic Programming，所以我们简称动态规划为DP)是运筹学的一个分支，是求解决策过程(decision process)最优...

 u013445530     2015-05-11 16:39:09     阅读数：35852

## 开局只有5个小兵,从荒岛到建立帝国！

突然火爆了80后朋友圈,都在攻城建家园,打造自己的帝国时代！

百度广告



## 五个常用**算法**（一）： 动态规划

1.从01背包问题说起 有一堆宝石一共n个，现在你身上能装宝石的就只有一个背包，背包的容量为C。把n个宝石排成一排并编上号： 0,1,2,...,n-1。第i个宝石对应的体积和价值分别为V[i]和W[...

 byplane     2016-10-03 17:22:29     阅读数：4505

## 动态规划**算法**学习

笔试面试中经常会出现一些考察动态规划方面的题目，以前没有接触过，现在初学做个整理。 1. 什么是动态规划？     和分治法一样，动态规划（dynamicprogramming）是通过组...

 nevasun     2011-11-16 16:24:41     阅读数：20959



## Java仗剑走...

关注

原创	粉丝	喜欢	评论
369	180	103	114

等级: 博客 6

访问: 39万+

积分: 7356

排名: 3950

勋章:  

博主专栏		
	HTML5+CSS3	
	阅读量：12494	15 篇
	算法-名企面试	
	阅读量：12995	13 篇
	ORACLE	
	阅读量：15370	14 篇
	JAVA23种设计模式	
	阅读量：1388	3 篇
	MYSQL	
展开		

个人分类	
-----Java-----	3篇
Java基础	56篇
Java设计模式	6篇
Javaweb	24篇
Java常用类	11篇
展开	



归档	
2018年3月	2篇
2018年2月	1篇
2017年11月	17篇
2017年10月	24篇
2017年9月	22篇
展开	

热门文章
<a href="#">eclipse如何导入项目和文件</a> 阅读量： 32704
<a href="#">mysql与Oracle的区别</a> 阅读量： 17352
<a href="#">一步步实现：JPA的基本增删改查CRUD（jpa基于hibernate）</a> 阅读量： 13168
<a href="#">如何写H5/web前端工程简历中的项目经验</a> 阅读量： 13143
<a href="#">如何设置input实现同时选中多个文件并同时上传</a> 阅读量： 13031

最新评论
<a href="#">javaSE_Java第一周总结： ...</a> qq_41992805： public class Test1Change{ public static void ...
<a href="#">javaSE_Java第一周总结： ...</a> qq_41992805： javaSE_Java第一周总结： 有难度题目集合的第一个交换变量那里 c错写为了a
<a href="#">在Eclipse上安装插件spri...</a> kong2660： 是不是只有ecipse4.6.3以上的版本才可以安装
<a href="#">漫画说算法--动态规划算法三（绝对...</a> qq_40764212： 同学，你这里有点问题，不可能出现950的情况 1,4号矿兼得的时候，你少算了一个人
<a href="#">eclipse如何导入项目和文件</a> Sunny5319： 谢谢您的分享



### 韩式床上用品





网络游戏前十名



人脸识别算法



前十名网络游戏



十大端游排行榜



联系我们



请扫描二维码联系客服

✉webmaster@csdn.net

☎400-660-0108

🗨QQ客服 🗨客服论坛

关于 · 招聘 · 广告服务 · 网站地图

©2018 CSDN版权所有 京ICP证09002463号

🐾 百度提供支持

经营性网站备案信息

网络110报警服务

中国互联网举报中心

北京互联网违法和不良信息举报中心

联系我们



请扫描二维码联系客服

✉webmaster@csdn.net

☎400-660-0108

🗨QQ客服 🗨客服论坛

关于 · 招聘 · 广告服务 · 网站地图

©2018 CSDN版权所有 京ICP证09002463号

🐾 百度提供支持

经营性网站备案信息

网络110报警服务

中国互联网举报中心

北京互联网违法和不良信息举报中心