Q 搜索 ど创作 ▼ ♠ CSDN 博客 专栏课程 下载 问答 社区 插件 认证 会员中心 计 收藏 动态 消息 ┛ 背包问题 迷亭1213 2019-02-24 16:18:10 💿 53821 🍁 已收藏 612 版权 分类专栏: 动态规划 Algorithm ACM学习笔记专栏 动... 同时被3个专栏收录▼ 订阅专栏 6 订阅 41 篇文章 本文为个人笔记,这算是我第三次系统的学背包问题。根据老师所写课件,所记为个人感悟以 及知识点梳理,方便日后复习,如有不足之处请斧正,感谢。 状态转移方程 背包问题的方案总数 0/1背包问题 空间优化 有依赖的背包问题 状态转移方程 时间优化 背包问题 分组背包问题 完全背包问题 空间优化 小优化 转化为0/1背包问题 二维费用背包问题 基本算法 混合背包问题 多重背包问题 转化为0/1背包问题 1.背景: **1.1 什么是背包问题:** 背包问题指这样一类问题, 题意往往可以抽象成: 给定一组物品, 每种物品都 有自己的重量和价格,在限定的总重量内,我们如何选择,才能使得物品的总价格最高。(来自百 度百科) 1.2 背包问题的种类:就ACM或者其它算法竞赛而言,背包问题可以分为8种类型,其中最基础的是 **0/1背包问题**。作为动态规划的典型问题,其状态转移方程往往需要认真理解并能自行推出。**这八种** 问题分别为: 0/1背包问题、完全背包问题、多重背包问题、混合三种背包问题、二维费用背包问 题、分组背包问题、有依赖的背包问题、求背包问题的方案总数。 2.0/1背包问题 **2.1 问题描述**:有N件物品和一个容量为V的背包。第i件物品的费用(即体积,下同)是w[i],价值 是val[i]。求解将哪些物品装入背包可使这些物品的费用总和不超过背包容量,且价值总和最大。 2.2 解题思路:用动态规划的思路,阶段就是"物品的件数",状态就是"背包剩下的容量",那么很 显然f[i,v]就设为从前i件物品中选择放入容量为 v 的背包最大的价值。那么状态转移方程为: f[i][v]=max{ f[i-1][v],f[i-1][v-w[i]]+val[i] }. 这个方程可以如下解释:只考虑子问题"将前 i 个物品放入容量为 v 的背包中的最大价值"那么考虑 如果不放入 i ,最大价值就和 i 无关,就是 f[i - 1][v] , 如果放入第 i 个物品,价值就是 f[i - 1][v w[i]] + val[i], 我们只需取最大值即可。 2.3 空间优化:上述状态表示,我们需要用二维数组,但事实上我们只需要一维的滚动数组就可以 递推出最终答案。考虑到用f[v]来保存每层递归的值,由于我们求f[i][v]的时候需要用到的是f[i-1][v] 和 f[i-1][v - w[i]] 于是可以知道,只要我们在求f[v]时不覆盖f[v - w[i]],那么就可以不断 递推至所求答案。所以我们采取倒序循环,即v=m(m为背包总容积)伪代码如下: for i = 1..Nfor v = V..0 $f[v] = max{ f[v], f[v-w[i]] + val[i]};$ 2.4 代码模板: (根据2.1问题作答) 1 #include<iostream> 2 | #include<cstdio> 3 #include<cstring> 4 using namespace std; 5 const int maxn = 1e4; int f[maxn]; int w[maxn], val[maxn]; void solve(int n,int m){ memset(f,0,sizeof f); 9 for(int i = 1;i <= n;i++){ 10 for(int v = m; v > 0; v--){ 11 if(v >= w[i])12 $f[v] = \max(f[v], f[v-w[i]] + val[i]);$ 13 14 15 printf("%d\n",f[m]); 16 17 int main(){ 19 int n,m; while(scanf("%d%d",&n,&m) != EOF){ 20 for(int i = 1;i <= n;i++) scanf("%d%d",w+i,val+i);</pre> 21 solve(n,m); 22 23 24 return 0; 25 } 3.完全背包问题 3.1 问题描述: 有N种物品和一个容量为V的背包, 每种物品都有无限件可用。第i种物品的费用是 w[i],价值是val[i]。求解将哪些物品装入背包可使这些物品的费用总和不超过背包容量,且价值总 和最大。 3.2 解题思路:完全背包问题与0/1背包问题不同之处在于其每个物品是无限的,从每种物品的角度 考虑,与它相关的策略就变成了取0件、1件、2件...。我们可以根据0/1背包的思路,对状态转移方 程进行改进,令 $\mathbf{f[i][v]}$ 表示前 \mathbf{i} 种物品恰放入一个容量为 \mathbf{v} 的背包的最大权值。状态转移方程就变成 了: $f[i][v] = max{f[i-1][v-k*w[i]] + k*val[i] | 0 <= k*w[i] <= v}_{o}$ 我们通过对0/1背包的思路加以改进,就得到了完全背包的一种解法,这种解法时间复杂度为O (n^3) , 空间复杂度为O(n^2) 。 **3.3 时间优化**:根据上述f[i][v]的定义,其为前 i 种物品恰好放入容量为 v 的背包的最大权值。根 据上述状态转移方程可知,我们假设的是**子结果f[i-1][v-k*w[i]]中并没有选入第 i 种物品**,所以我 们需要逆序遍历(像0/1背包一样)来确保该前提;但是我们现在考虑"加选一件第 i 种物品"这种策 略时,正**需要一个可能已经选入第 i 种物品的子结果f[i][v-w[i]]**,于是当我们**顺序遍历**时,就刚好 达到该要求。这种做法,使我们省去了一层循环,即第i种物品放入的件数k,从而时间复杂度优化 为O (n^2)。 **3.4 空间优化:** 正如0/1背包的空间优化,上述状态转移方程已经优化为: f[i][v]=max{f[i-1][v],f[i][v-w[i]]+val[i]} 将这个方程用一维数组实现,便得到了如下伪代码: for i = 1..Nfor v = 0..V $f[v] = max\{f[v], f[v-w[i]] + val[i]\};$ 3.5 小剪枝: 完全背包问题有一个很简单有效的优化, 是这样的: 若两件物品i、j满足w[i] <= w[j]且 val[i] >= val[i],则将物品i去掉,不用考虑。这个优化的正确性显然:任何情况下都可将价值小费用 高的j换成物美价廉的i,得到至少不会更差的方案。对于随机生成的数据,这个方法往往会大大减少 物品的件数,从而加快速度。然而这个并不能改善最坏情况的复杂度,因为有可能特别设计的数据 可以一件物品也去不掉。 3.6 转化为0/1背包问题: 既然01背包问题是最基本的背包问题,那么我们可以考虑把完全背包问题转化为01背包问题 来解。最简单的想法是,考虑到第i种物品最多选V/w[i]件,于是可以把第i种物品转化为V/w[i]件费用 及价值均不变的物品,然后求解这个01背包问题。这样完全没有改进基本思路的时间复杂度,但这 毕竟给了我们将完全背包问题转化为01背包问题的思路:将一种物品拆成多件物品。 更高效的转化方法是:把第i种物品拆成费用为w[i]*2^k、价值为val[i]*2^k的若干件物品,其 中k满足w[i]*2^k<V。这是二进制的思想,因为不管最优策略选几件第i种物品,总可以表示成若干 个2^k件物品的和。这样把每种物品拆成O(log(V/w[i])+1)件物品,是一个很大的改进。 3.7 代码示例:完全背包问题模板 4.多重背包问题 **4.1 问题描述**: N种物品和一个容量为V的背包。第i种物品最多有n[i]件可用,每件费用是w[i],价值 是val[i]。求解将哪些物品装入背包可使这些物品的费用总和不超过背包容量,且价值总和最大。 4.2 解题思路: 这种类型的题目又和完全背包有些相似,不同的就在于其数量不是无限的。基本的 方程只需将完全背包问题的方程略微一改即可,因为对于第i种物品有n[i]+1种策略:取0件,取1 件…… $\mathbf{p}_{[i]}$ 件。令 $\mathbf{f}_{[i]}[v]$ 表示前 $\mathbf{f}_{[i]}$ 种物品恰放入一个容量为 \mathbf{v} 的背包的最大权值,则: $\mathbf{f}_{[i]}[v]$ = $\mathbf{m}_{[i]}$ 1][v-k*w[i]] + k*val[i] | 0<= k <= n[i]}。复杂度是O(V*Σn[i])。 4.3 转化为0/1背包问题: 把第i种物品换成n[i]件01背包中的物品,则得到了物品数为 $\Sigma n[i]$ 的01背包问题,直接求解, 复杂度仍然是O(V*∑n[i])。 但是我们期望将它转化为01背包问题之后能够像完全背包一样降低复杂度。仍然考虑二进制 的思想,我们考虑把第i种物品换成若干件物品,使得原问题中第i种物品可取的每种策略——取 0..n[i]件——均能等价于取若干件代换以后的物品。另外,取超过n[i]件的策略必不能出现。 方法是:将第i种物品分成若干件物品,其中每件物品有一个系数,这件物品的费用和价值均 是原来的费用和价值乘以这个系数。使这些系数分别为 1,2,4,...,2^(k-1),n[i]-2^k+1, 且k是满足 $n[i]-2^k+1>0$ 的最大整数**(注意:这些系数已经可以组合出1\sim n[i]内的所有数字)**。例如,如果n[i]为 13, 就将这种物品分成系数分别为1,2,4,6的四件物品。 分成的这几件物品的系数和为n[i]、表明不可能取多于n[i]件的第i种物品。另外这种方法也能 保证对于0..n[i]间的每一个整数,均可以用若干个系数的和表示,这个证明可以分0..2^k-1和 2^k..n[i]两段来分别讨论得出,并不难,希望你自己思考尝试一下。 这样就将第i种物品分成了O(logn[i])种物品,将原问题转化为了复杂度为O(V*∑logn[i])的O1 **背包问题**,是很大的改进。 4.4 参考模板:例题-庆功会 5.混合三种背包问题 **5.1 问题描述**:如果将01背包、完全背包、多重背包混合起来。也就是说,有的物品只可以取一次 (01背包),有的物品可以取无限次(完全背包),有的物品可以取的次数有一个上限(多重背 包)。应该怎么求解呢? 5.2 0/1背包与完全背包的混合: 考虑到在01背包和完全背包中最后给出的伪代码只有一处不同,故如果只有两类物品:一类物 品只能取一次,另一类物品可以取无限次,那么只需在对每个物品应用转移方程时,根据物品的类 别选用顺序或逆序的循环即可,复杂度是O(VN)。 伪代码如下: for i=1..N if 第i件物品是01背包 for v=V..0 $f[v] = max\{ f[v], f[v-w[i]] + val[i] \};$ else if 第i件物品是完全背包 for v=0..V $f[v] = max\{ f[v], f[v-w[i]] + val[i] \};$ 5.3 再加上多重背包: 如果再加上有的物品最多可以取有限次,那么原则上也可以给出O(VN)的解法:遇到多重背包 类型的物品用单调队列解即可。但如果不考虑超过NOIP范围的算法的话,用多重背包中将每个这类 物品分成O(log n[i])个01背包的物品的方法也已经很优了。 5.4 例题:混合背包 6.二维费用背包问题 **6.1 问题描述**:二维费用的背包问题是指:对于每件物品,具有两种不同的费用;选择这件物品必须 同时付出这两种代价;对于每种代价都有一个可付出的最大值(背包容量)。问怎样选择物品可以 得到最大的价值。设这两种代价分别为代价1和代价2,第i件物品所需的两种代价分别为a[i]和b[i]。 两种代价可付出的最大值(两种背包容量)分别为V和U。物品的价值为c[i]。 6.2 算法: 费用加了一维,只需状态也加一维即可。设f[i][v][u]表示前i件物品付出两种代价分别为v和u 时可获得的最大价值。 状态转移方程就是: f [i][v][u]=max{f[i-1][v][u],f[i-1][v-a[i]][u-b[i]]+c[i]}。如前述方法,可以 只使用二维的数组:当每件物品只可以取一次时变量v和u采用逆序的循环,当物品有如完全背包问 题时采用顺序的循环。当物品有如多重背包问题时拆分物品。 6.3 物品总个数的限制: 有时,"二维费用"的条件是以这样一种隐含的方式给出的:最**多只能取M件物品**。这事实上相 当于每件物品多了一种"件数"的费用,每个物品的件数费用均为1,可以付出的最大件数费用为M。 换句话说,设f[v][m]表示付出费用v、最多选m件时可得到的最大价值,则根据物品的类型(01、 完全、多重)用不同的方法循环更新,最后在f[0..V][0..M]范围内寻找答案。 另外,如果要求"恰取M件物品",则在f[0..V][M]范围内寻找答案。 6.4 例题: 潜水员 7.分组背包问题 7.1 问题描述:有N件物品和一个容量为V的背包。第i件物品的费用是w[i],价值是c[i]。这些物品被 划分为若干组,每组中的物品互相冲突,最多选一件。求解将哪些物品装入背包可使这些物品的费 用总和不超过背包容量,且价值总和最大。 7.2 算法: 这个问题变成了每组物品有若干种策略:是选择本组的某一件,还是一件都不选。也就是说设f[k] [v]表示前k组物品花费费用v能取得的最大权值,则有f[k][v]=max{ f[k-1][v], f[k-1][v-w[i]] + c[i] | 物品i属于第k组 }。 使用一维数组的伪代码如下: for 所有的组k for v=V..0for 所有的i属于组k $f[v]=max\{f[v],f[v-w[i]]+c[i]\}$ 注意这里的三层循环的顺序、"for v=V..0"这一层循环必须在"for 所有的i属于组k"之外。这样才 能保证每一组内的物品最多只有一个会被添加到背包中。 另外,显然可以对每组中的物品应用完全背包中"一个简单有效的优化"(3.5)。 7.3 例题: 分组背包 8.有依赖的背包问题 8.1 简化的问题: 这种背包问题的物品间存在某种"依赖"的关系。也就是说,i依赖于i,表示若选物品i,则必须选 物品j。为了简化起见,我们先设没有某个物品既依赖于别的物品,又被别的物品所依赖;另外,没 有某件物品同时依赖多件物品。 8.2 算法: 这个问题由NOIP2006金明的预算方案一题扩展而来。遵从该题的提法,将不依赖于别的物品 的物品称为"主件",依赖于某主件的物品称为"附件"。由这个问题的简化条件可知所有的物品由若 干主件和依赖于每个主件的一个附件集合组成。 按照背包问题的一般思路,仅考虑一个主件和它的附件集合。可是,可用的策略非常多,包 括:一个也不选,仅选择主件,选择主件后再选择一个附件,选择主件后再选择两个附件.....无法用 状态转移方程来表示如此多的策略。(事实上,设有n个附件,则策略有2^n+1个,为指数级。) 考虑到所有这些策略都是互斥的(也就是说,你只能选择一种策略),所以一个主件和它的附 件集合实际上对应于**分组的背包**中的一个物品组,每个选择了主件又选择了若干个附件的策略对应 于这个物品组中的一个物品,其费用和价值都是这个策略中的物品的值的和。但仅仅是这一步转化 并不能给出一个好的算法,因为物品组中的物品还是像原问题的策略一样多。 再考虑**分组的背包**中的一句话:可以对每组中的物品应用**完全背包**中"一个简单有效的优化"。 这提示我们,对于一个物品组中的物品,所有费用相同的物品只留一个价值最大的,不影响结果。 所以,我们可以对主件i的"附件集合"先进行一次01背包,得到费用依次为0..V-w[i]所有这些值时相 应的最大价值f'[0..V-w[i]]。那么这个主件及它的附件集合相当于V-w[i]+1个物品的物品组、其中费 用为w[i]+k的物品的价值为f'[k]+c[i]。也就是说原来指数级的策略中有很多策略都是冗余的,通过一 次01背包后,将主件i转化为 V-w[i]+1个物品的物品组,就可以直接应用分组的背包的算法解决问题 了。 **更一般的问题是:** 依赖关系以图论中"森林"的形式给出(森林即多叉树的集合),也就是说 主件的附件仍然可以具有自己的附件集合,限制只是每个物品最多只依赖于一个物品(只有一个主 件)目不出现循环依赖。 解决这个问题仍然可以用将每个主件及其附件集合转化为物品组的方式。唯一不同的是,由于 附件可能还有附件,就不能将每个附件都看作一个一般的01 背包中的物品了。若这个附件也有附件 集合,则它必定要被先转化为物品组,然后用分组的背包问题解出主件及其附件集合所对应的附件 组中各个费用的附件所对应的价值。 事实上,这是一种树形DP,其特点是每个父节点都需要对它的各个儿子的属性进行一次DP以求 得自己的相关属性。这已经触及到了"泛化物品"的思想。看完后,你会发现这个"依赖关系树"每一 个子树都等价于一件泛化物品,求某节点为根的子树对应的泛化物品相当于求其所有儿子的对应的 泛化物品之和。 8.3 小结: NOIP2006的那道背包问题,通过引入"物品组"和"依赖"的概念可以加深对这题的理解,还可 以解决它的推广问题。用物品组的思想考虑那题中极其特殊的依赖关系:物品不能既作主件又作附 件、每个主件最多有两个附件,可以发现一个主件和它的两个附件等价于一个由四个物品组成的物 品组,这便揭示了问题的某种本质。 9.求背包问题的方案总数 9.1 问题描述:对于一个给定了背包容量、物品费用、物品间相互关系(分组、依赖等)的背包问 题,除了再给定每个物品的价值后求可得到的最大价值外,还可以得到装满背包或将背包装至某一 指定容量的方案总数。 9.2 算法: 对于这类改变问法的问题,一般只需将状态转移方程中的max改成sum即可。例如若每件物品 均是01背包中的物品,转移方程即为f[i][v] = sum{ f[i-1][v], f[i-1][v-w[i]] + c[i] }, 初始条件 f[0][0] = 1. 事实上,这样做可行的原因在于状态转移方程已经考察了所有可能的背包组成方案。 9.3 例题: 货币系统 PS:本来想写个人心得整理的,但无奈怎么总结都没原文写的易懂,尤其是后面几个没怎么 接触过的背包问题,所以后面基本都是复制课件的,等练习练习再把updata吧。 迷亭1213 (已关注) 150 **1**3 **1**3 **6**12 背包问题(0-1背包,完全背包,多重背包知识概念详解) 06-14 背包问题(0-1背包,完全背包,多重背包知识概念详解)内含实例代码解析,详细讲解了背包的基本概... 【动态规划】01背包问题(通俗易懂,超基础讲解) 热门推荐 Yngz Miao的博客 © 24万+ 问题描述 有n个物品,它们有各自的体积和价值,现有给定容量的背包,如何让背包里装入的物品… 请发表有价值的评论, 博客评论不欢迎灌水, 良好的社区氛围需大家一起维护。 breakankle: 大佬总结很全面呀 6 月前 回复 ••• 迷亭1213 码龄4年 ♥ 暂无认证 ➡️ 小暖心呀: 顶尖 6月前 回复 ••• 3743 2271 107万+ 程序猿从还没入门到放弃: 完全背包问题中for i = 1..N for v = 0..V周排名 总排名 原创 访问 ax{f[v],f[v-w[i]] + val[i] }; 当v<w[i]的时候这个数组f[v-w[i]]不是越界了吗? 1年前 回复 ••• → 迷亭1213 博主 回复: 是的,会越界,所以在实现的时候一般会判断一下,伪代码为 1万+ 2062 2329 817 150 了简洁就没加,因为不是所有语言都会越界 1 年前 回复 ••• 评论 积分 粉丝 获赞 收藏 DAY PAY 石头会说话20: 可以转载吗? 1年前 回复 ••• | 迷亭1213 | 🙀 | 回复: 可以 1 年前 | 回复 🐽 | 已关注 私信 wind_snow_broken: 测试数据错误啊好像 5 10 2 2 6 5 4 6 3 5 4 6 14 1 年前 回复 ••• wind_snow_broken 回复: 不是呢 我的是我表达错了 2对6 2对3 6对5 5对4 4对6 答 案应该是15 1年前 回复 ••• Q 搜博主文章 🥠 迷亭1213 睡 回复: 背包容量为 10,选一个重2价值为2的,两个重四价值为6的, Bug 热门文章 总价值14是最大啊 1年前 回复 ••• 背包问题-笔记整理 ① 53732 skywalker_H: 小编,请问多重背包可以将其转化为01背包问题求解,其中的二进制改进方 Python中xlrd常用用法整理 ① 41482 法,为什么不直接将n[i]用二进制表示,而一定要表示成2的幂与一个数之和的形式?谢谢 2 年前 回复 ••• c语言double类型printf问题 ⊙ 40620 ※ 送亭1213 博主 回复: 因为我们的目的是使得拆分后的物品可以替代任意属于[1, n[i]]

■ ● C++ isalpha isalnum islower 的数量的物品,当且仅当前k位二进制都存在集合内,我们才可以根据(选它或者不选 isupper用法 o 35110 它,01背包)来组合出任意 k 位二进制数,所以前k位二进制,即1,2,4,...,2^k C/C++STL常用容器用法总结 • 都要存在才行。 2年前 回复 ••• 29757 JaceYangg: f[i][v]=max{ f[i-1][v],f[i-1][v-w[i]]+c[i] } 博主01背包这个公式最后写错了吧, c[i] Bug 应该是val[i] 2 年前 回复 ••• 分类专栏 → 送亭1213 博主 回复: 是的,已改正,谢谢 2 年前 回复 ••• 单调栈 1篇 算法每日一练 4篇 动态规划0—1背包问题 逐梦科学 ① 15万+ 动态规划0-1背包问题 Ø 问题描述: 给定n种物品和一背包。物品i的重量是wi,其价值为vi,背... 高性能计算 1篇 在头条干了两年后含泪整理的职场经验,太真实.... 最新发布 weixin 63757250的博客 **②** 63 MPI 1篇 在字节跳动和滴滴干了 2 年后端开发,太真实…先简单交代一下背景吧,某不知名 985 的本硕,17… ACM学习笔记专栏 38篇 【算法设计】背包问题_小魏的修行路_背包问题 10-30 【算法设计】背包问题 研究生课程系列文章参见索引《在信科的那些课》 题目 一个旅行者准备随... Linux 2篇 01背包问题 图解+详细解析 (转载)_可控的事情要谨慎,不... 10-23 根据动态规划解题步骤(问题抽象化、建立模型、寻找约束条件、判断是否满足最优性原理、找大问... Bug 最新评论 【算法设计】背包问题 编程语言小筑 ① 1205 研究生课程系列文章参见索引《在信科的那些课》 题目 一个旅行者准备随身携带一个背包,可以... JAVA用"*"打印出圆形实验案例 冷静388: 是个抽象的圆,有帮助 背包问题讲解 判断搜索树后序遍历是否合法(单... 动态规划(dynamic programming)算法是解决多阶段决策过程最优化问... 动态规划基本思想 Sinova_L: 哥哥好帅 背包问题详解_JYplute的博客 11-1 st表 如果你对比一下01背包问题中的递归解法,就会发现唯一的区别便是这里多了一层循环,因为01背包... SuperBrain0xb: 什么嘛,看不懂qw JAVA用"*"打印出圆形实验案例 九种0-1 背包问题详解_Tyler_Zx的博客 11-13 qq 51330453: 你这代码打出来还是椭 问题9:背包问题求具体方案 0-1 背包是一个经典的问题,之前也整理过一篇关于0-1 背包的博客,当时... 圆圆 背包问题总结 C++ 快速幂运算 前面是转载来的背包9讲,非常详细,后面有几个lintcode上的题目前言本篇文章是我(dd_engi)正... 帅气的Ezio: mod, 通常是10000000 彻底理解0-1背包问题 跑码场 ① 14万+ 您愿意向朋友推荐"博客详情页"吗?19 0-1背包问题 给定n个重量为w1, w2,w3,...,wn, 价值为v1,v2,v3,...,vn的物品和容量为C的背包, 求... 背包问题汇总_maershii_leetcode 背包问题汇总 11-5 0-1背包问题,无价值:https://www.lintcode.com/problem/backpack/description 问题描述:Givennitem... 强烈不推荐 不推荐 一般般 背包问题 懂代码的暖男 ① 1万+ #include <iostream> #include <vector> #include "a_arrayAndMatrix.h" using namespace std; vect... Bug 最新文章 四种常见背包问题整理 月巴虫丰博客 ② 2940 Balance Tree 与 B+树 ① 最优装配 ② 部分背包 ③ 01背包 ④ 完全背包 判断搜索树后序遍历是否合法(单调 背包问题详解: 01背包、完全背包、多重背包 na_beginning的专栏 ① 10万+ 栈) 参考链接: http://www.cnblogs.com/fengty90/p/3768845.html http://blog.csdn.net/mu399/article/de... MPI 点对点通信 背包问题-两个背包问题 qq 43313769的博客 ① 3244 递归解法: public class Main0 { static int n = 5; static int[] v={0,3,7,4,49,2}; static int[] w={0,3,4,7,5... 2021年 4篇 2020年 24篇 一举拿到5份offer的面试资料分享,没有捷径只有艰辛 weixin_42097508的博客 **②** 23 2019年 132篇 2018年 118篇 整理了面试真题送给大家1: jdk1.7 到 jdk1.8 Map 发生了什么变化(底层)?1.8 之后 hashMap 的数... 01背包问题详解(浅显易懂) Iseno_V的博客 ① 1万+ 01背包问题详解 01背包是一种动态规划问题。动态规划的核心就是状态转移方程,本文主要解释0... 用「dyral収減,吸기史 夕凹까大豕 经典背包问题的探讨 weixin 30515513的博客 **②** 45 一、背包问题的描述 背包问题可以有多种形式,下面将对其逐一进行描述: (1) 经典的0-1背包... 使用PayPal跨境收款服务,可以吸引 更多消费者,帮助优先获得买家信赖。 九大背包问题专题--完全背包问题(详解,最优解) gl620321的博客 ① 4733 PayPal 2.完全背包问题 和01背包问题的区别: 01背包1件物品只能选或者不选 完全背包问题: 1件物品可... 注册 最详细动态规划解析——背包问题 oscarwin ^② 4万+ 动态规划的定义要解决一个复杂的<mark>问题</mark>,可以考虑先解决其子<mark>问题</mark>。这便是典型的递归思想,比如… Bug 目录 背包问题简述 mandagod的博客 ② 223 "<mark>背包</mark>题目"的基本描述是:有一个<mark>背包</mark>,能盛放的物品总重量为S,设有N件物品,其重量分别为w1... 1.背景: 0-1背包问题详解(一步一步超详细) qq_42051306的博客 ① 1914 2.0/1背包问题 0-1背包问题详解 1.什么叫背包问题? 背包问题通俗的说,就是假如你面前有5块宝石分别为a, b, c,... 3.完全背包问题 背包九讲 weixin 30924239的博客 **①** 143 背包九讲HTML文件下载 P01: 01背包问题 题目 有N件物品和一个容量为V的背包。第i件物品的费... 4.多重背包问题 ◎2021 CSDN 皮肤主题: 程序猿惹谁了 设计师:白松林 返回首页 5.混合三种背包问题 6) 6.二维费用背包问题 关于我们 招贤纳士 广告服务 开发助手 ☎ 400-660-0108 ☑ kefu@csdn.net ⑤ 在线客服 工作时间 8:30-22:00 7.分组背包问题 公安备案号11010502030143 京ICP备19004658号 京网文〔2020〕1039-165号 经营性网站备案信息 北京互联网违法和不良信息举报中心 网络110报警服务 中国互联网举报中心 家长监护 Chrome商店下载 ©1999-2021北京创新乐知网络技术有限公司 版权与免责声明 版权申诉 8.有依赖的背包问题 出版物许可证 营业执照 不 9.求背包问题的方案总数