



个人资料



寂静山林

访问：332658次

积分：5297

等级：

排名：第3134名

原创：150篇

转载：1篇

译文：0篇

评论：236条

文章搜索

文章分类

入门 (16)

回溯法 (9)

字符串 (8)

排序 (9)

数据结构 (9)

数论 (10)

算术与代数 (9)

组合数学 (8)

日常应用 (4)

图遍历 (8)

图算法 (9)

动态规划 (14)

网络 (11)

几何 (8)

计算几何 (11)

算法总结 (4)

C# (4)

软件工程 (1)

dot Net Framework (1)

C++ (1)

文章存档

2015年02月 (1)

2014年08月 (1)

UVa Problem 10154 Weights and Measures（重量和力量）

标签：[output](#) [input](#) [integer](#) [优化](#) [测试](#) [each](#)

2011-10-11 17:20 2385人阅读 [评论\(2\)](#) [收藏](#) [举报](#)

☰ 分类：[动态规划 \(13\)](#) ▼

版权声明：本文为博主原创文章，未经博主允许不得转载。

[cpp]

01. // Weights and Measures（重量和力量）

02. // PC/UVa IDs: 111103/10154, Popularity: C, Success rate: average Level: 3

03. // Verdict: Programming Challenges - Solved, UVa - Accepted

04. // Submission Date: 2011-10-12

05. // UVa Run Time: 0.080s

06. //

07. // 版权所有（C）2011，邱秋。metaphysis # yeah dot net

08. //

09. // I know, up on top you are seeing great sights,

10. // But down at the bottom, we, too, should have rights.

11. // We turtles can't stand it. Our shells will all crack!

12. // Besides, we need food. We are starving!" groaned Mack.

13. //

14. // Yertle the Turtle, Dr.Seuss

15. //

16. // [Problem Description]

17. // A turtle named Mack, to avoid being cracked, has enlisted your advice as to

18. // the order in which turtles should be stacked to form Yertle the Turtle's throne.

19. // Each of the 5,607 turtles ordered by Yertle has a different weight and strength.

20. // Your task is to build the largest stack of turtles possible.

21. //

22. // [Input]

23. // Standard input consists of several lines, each containing a pair of integers

24. // separated by one or more space characters, specifying the weight and strength

25. // of a turtle. The weight of the turtle is in grams. The strength, also in grams,

26. // is the turtle's overall carrying capacity, including its own weight. That is,

27. // a turtle weighing 300 g with a strength of 1,000 g can carry 700 g of turtles

28. // on its back. There are at most 5,607 turtles.

29. //

30. // [Output]

31. // Your output is a single integer indicating the maximum number of turtles that

32. // can be stacked without exceeding the strength of any one.

33. //

34. // [Sample Input]

35. // 300 1000

36. // 1000 1200

37. // 200 600

38. // 100 101

39. //

40. // [Sample Output]

41. // 3

42. //

43. // [解题方法]

44. // 开始以为是很简单的 DP 题，把乌龟按可承重重量增序排列，若可承重重量相同，则体重轻的排上面，然后

45. // DP 求能得到的最大高度。程序在 Programming Challenges 倒是通过了，但是在 UVa 上是 WA。阅

46. // 读了 Uva BBS 上的帖子，才发现之所以在 PC 上通过是因为其测试数据太弱（汗...）。

47. //

48. // 先分析一下按可承重重量增序排列，若可承重重量相同，体重轻的排上面的方法为什么不可行。可

49. // 承重重量除了给出这只乌龟还能承重的重量外，不能给出更多信息了，如两只乌龟，重量和力量如下：

50. //

2014年07月 (2)

2014年02月 (1)

2014年01月 (2)

展开

阅读排行

《挑战编程:程序设计竞赛

(16976)

"No mapping for the Uni

(9847)

Linux 显示器未正确识别

(9195)

如何制作自动更新程序?

(8514)

《挑战编程:程序设计竞赛

(7947)

UVa Problem 100 The 3i

(7772)

UVa Problem 10181 15-

(4203)

UVa Problem 861 Little I

(3811)

Ubuntu 10.10 天翼3G 华

(3733)

UVa Problem 10189 Mir

(3209)

推荐文章

*EventBus的使用与深入学习

*Android 拍照、选择图片并裁剪

*spark性能调优：开发调优

*浅谈android中图片处理之色

彩特效处理ColorMatrix(三)

*neutron-server的启动流程(一)

*Hadoop中Map端shuffle源码解

析

最新评论

UVa Problem 843 Crypt Kicker

珏: 来看你好几篇博文了，写的

很好！这种问题我一般只有思

路，但是实现时就头大了。。。

谢谢！

如何制作自动更新程序？

qq_34153104: 感谢楼主分享

UVa Problem 10142 Australian'

2D_: 博主你好~你的代码里面我

有一小部分不是很懂//根据min值

剔除选票最少的候选人及其在选

票的位置。...

UVa Problem 10049 Self-descri

寂静山林: 不知道您是否用的是

VC++编译器或者其他编译器。我

使用GCC4.8.4进行编译运行正

常。

UVa Problem 10049 Self-descri

Slow_Wakler: int index; 这个

index没有赋值，还有这个程

序。。。运行会出现错

误：。。。已停止工作

如何制作自动更新程序？

aidisoft: 如何维护服务器端xml文

件与实际版本的一致性，有什么

办法让xml自动获取版本信息？

UVa Problem 10152 ShellSort

qq_29568007: 楼主你的代码在

Programming Challenge 上是

wrong answer的我的也是，...

用 C# 编写 USB 存储设备使用痕

platescp: 先下载看看，学习一下

如何制作自动更新程序？

qq_16001865: 用http的地址提示

无法连接远程服务器,咋修改呢？

UVa Problem 10137 The Trip（

oceanpearl: @satan_1st用贴出

的这组数，楼主的算法的结果是

0.14，用我贴出的算法的结果是

您说的0.1...

评论排行

UVa Problem 100 The 3i

(27)

```
51. // (1) 10 50
52. // (2) 100 120
53. //
54. // 如果按照上述的排序方法乌龟（1）应该排在下方，（2）排在上方，能形成的乌龟塔最高为 1，但是实际上
55. // 若（2）在下，（1）在上，是能形成高度为 2 的乌龟塔的。所以按这种排序方法是不能达到乌龟顺序的最
56. // 优子结构的，所以也就不能保证一定获得最优解。
57. //
58. // 若按重量增序，重量相同按力量增序的顺序排列乌龟，也会得到错误的答案，因为没有考虑到乌龟的可承重
59. // 重量，反例如下：
60. //
61. // (1) 10 1000
62. // (2) 20 1000
63. // (3) 30 40
64. //
65. // 排好后后乌龟塔的最高高度为 2，实际上应该是 3，只需将（3）放在最上面即可。
66. //
67. // 若按力量增序排列，力量相同按体重增序排列，从第一只乌龟开始处理，设当前新增加的乌龟为 i，从 1 到
68. // i - 1 搜索总重量小于乌龟 i 的可承重重量，且高度能增加的乌龟 j，更新乌龟 i 的总承重重量和能堆
69. // 叠的最大高度，使用这种方法，对于一般的测试数据，都能得到正确的答案，但是对于如下测试数据，却不能
70. // 得到正确的答案（之前我就是使用这种方法）：
71. //
72. // (1) 101 101
73. // (2) 100 201
74. // (3) 99 300
75. // (4) 98 301
76. // (5) 5 302
77. //
78. // 前述算法能得到的最大高度是 3，实际上将（2），（3），（4），（5）堆叠起来可以得到高度为 4 的乌
79. // 龟塔，为什么算法失效了？因为在第二步处理乌龟（2）的时候，因为乌龟（1）能放在乌龟（2）上，故乌龟
80. // （2）拥有了最大高度 2，从而在处理后续乌龟时，（2）不能作为单独乌龟放置在其他乌龟上，只能是和（1）
81. // 做为一个整体来放置，所以需要记录能达到高度 k 的乌龟塔时，乌龟的最小总重量，这样在构建乌龟塔时，
82. // 总是选择当前能堆叠最高且总重量最小的乌龟，才有可能构建更高的乌龟塔。
83. //
84. // 那么是否可以按力量来进行排序？答案是肯定的，假设有两只乌龟，重量和力量分别为 w1, s1, w2, s2，
85. // 且有 s1 <= s2，那么排序如下：
86. //
87. // w1 s1
88. // w2 s2
89. //
90. // 若力量为 s1 的乌龟能支撑起包括 w2 在内的乌龟重量，则有 s1 >= (w1 + w2)，因为有 s2 >= s1
91. // 则力量为 s2 的乌龟总是同样能支撑起来，但是反过来就不一定了，即 s2 >= (w1 + w2)，不一定有
92. // s1 >= (w1 + w2)，故按力量来排序总是不会改变最优结构的。
93. //
94. // 综上所述，使用一个二维数组记录使用前 i 只乌龟形成高度为 h 的塔时的最小总重量，设为 minWeight
95. // [h][i]，有以下转移方程：
96. //
97. // minWeight[h][i] = min{minWeight[h][i - 1], minWeight[h - 1][i - 1] + weight[i]}
98. //
99. // 其中第二个是有条件的，即 minWeight[h - 1][i - 1] + weight[i] <= strength[i]。同样的，
100. // 因为较多的乌龟，若使用二维数组因数组很大而会引起段错误，可以使用一维滚动数组来代替优化空间使
101. // 用以避免出现因分配过多内存导致段错误。为什么可以用一维数组来优化，这是题目的转移方程决定的，在
102. // 最开始的时候，乌龟数目为 0，形成高度为 0 的乌龟塔时，最小总重量当然为 0，但是对于从 1 到乌龟
103. // 总数的高度，是不可能堆叠成的，可设其最小总重量为一 MAXINT 值来标记。此时数组元素如下：
104. //
105. // minWeight[0][0] 0
106. // minWeight[1][0] MAXINT
107. // minWeight[2][0] MAXINT
108. // ... ...
109. // minWeight[N][0] MAXINT
110. //
111. // 假设需要计算乌龟数目为 1 时，形成的高度为 1 的乌龟塔最小总重量时，则根据转移方程，有：
112. //
113. // minWeight[1][1] = min{minWeight[1][0], minWeight[0][0] + weight[1]}
114. //
115. // 当然第二个值是有条件的（这里假设满足条件），从数组的元素可以知道，minWeight[1][0] 已经存在于
116. // 数组中，minWeight[0][0] 也存在于数组中，最后计算得到的值 minWeight[1][1]，所存放的位置与
117. // minWeight[1][1] 行标号相同，只不过列标号增加 1，此后列为 0 的元素就不会被使用了，那么可以直
118. // 接省略掉列标号，变成一维数组：
119. //
120. // minWeight[0] minWeight[1] minWeight[2] ... minWeight[N]
121. // 0 MAXINT MAXINT ... MAXINT
122. //
123. // 只不过在计算的时候，为了不覆盖前一次计算的值，每次都从最后一个元素开始使用转移方程往前计算即可。
124. // 前面的 Uva 10069 Distinct Subsequences 也可以使用的同样的方法来优化。
125.
126. #include <iostream>
127. #include <algorithm>
128.
129. using namespace std;
```

http://blog.csdn.net/metaphysis/article/details/6863505

2/4

UVa Problem 10189 Mir	(23)
《挑战编程:程序设计竞赛	(20)
UVa Problem 10137 The	(19)
UVa Problem 843 Crypt	(18)
UVa Problem 10142 Aus	(11)
UVa Problem 10044 Erd	(9)
UVa Problem 10149 Ya	(9)
如何制作自动更新程序?	(8)
UVa Problem 10196 Ch	(7)

```
130.
131. #define MAXN 5610
132. #define MAXINT (1 << 20)
133.
134. class turtle
135. {
136. public:
137.     int weight;
138.     int strength;
139.
140.     bool operator<(const turtle &other) const
141.     {
142.         // 力量大的的在下方。
143.         if (strength != other.strength)
144.             return strength < other.strength;
145.
146.         // 若力量相同, 则重的乌龟在下方。
147.         return weight < other.weight;
148.     };
149. };
150.
151. int main(int ac, char *av[])
152. {
153.     turtle turtles[MAXN];
154.     int minWeight[MAXN];
155.     int nTurtles = 1, weight, strength;
156.
157.     while (cin >> weight >> strength)
158.     {
159.         if (weight > strength)
160.             continue;
161.
162.         turtles[nTurtles++] = (turtle){weight, strength};
163.     }
164.
165.     // 按力量和自身重量排序。
166.     sort(turtles + 1, turtles + nTurtles);
167.     nTurtles--;
168.
169.     // 赋初值, 若为 MAXINT 则不可能形成高度为 h 的乌龟塔。
170.     minWeight[0] = 0;
171.     for (int h = 1; h <= nTurtles; h++)
172.         minWeight[h] = MAXINT;
173.
174.     // DP 找最大高度。
175.     int answer = 1;
176.     for (int i = 1; i <= nTurtles; i++)
177.         for (int h = nTurtles; h >= 1; h--)
178.         {
179.             if (minWeight[h - 1] + turtles[i].weight <= turtles[i].strength)
180.                 minWeight[h] = min(minWeight[h], minWeight[h - 1] + turtles[i].weight);
181.
182.             if (minWeight[h] < MAXINT)
183.                 answer = max(answer, h);
184.
185.         }
186.
187.     cout << answer << endl;
188.
189.     return 0;
190. }
```

顶 4 踩 0

上一篇 UVa Problem 10069 Distinct Subsequences (不同的子序列)
下一篇 UVa Problem 116 Unidirectional TSP (单向旅行商问题)

我的同类文章

动态规划（13）

• UVa Problem 104 - Arbitrage

2011-12-22

阅读 1467

• UVa Problem 111 - History ...

2011-12-02

阅读 692

• UVa Problem 10201 Adven...

2011-10-19

阅读 1243

• UVa Problem 10271 Chop...

2011-10-19

阅读 1465

• UVa Problem 103 - Stackin...

2011-10-17

阅读 660

• UVa Problem 10261 Ferry ...

2011-10-16

阅读 2907

• UVa Problem 10003 Cuttin...

2011-10-14

阅读 1675

• UVa Problem 116 Unidirec...

2011-10-12

阅读 1355

• UVa Problem 10069 Distin...

2011-10-11

阅读 2779


• UVa Problem 10131 Is Big...

2011-10-10

阅读 2649

更多文章

参考知识库



算法与数据结构知识库


2112 关注 | 3864 收录

猜你在找

营销型网站建设	UVA 10154Weights and Measures --dp
iOS8-Swift开发教程	uva10154 - Weights and Measures
软件测试基础	UVA - 10154 Weights and Measures
有趣的算法（数据结构）	UVA 10154 Weights and Measuresdp
从此不求人:自主研发一套PHP前端开发框架	UVA 10154 Weights and Measures


查看评论

2楼 F_THANK 2012-03-15 21:52发表



比较详细~~~

1楼 chchwy 2012-01-02 19:19发表



頂一個，這是我看到網路上最完整最詳細的烏龜塔解說，也是唯一從頭到尾都看得懂的，謝謝你。

您还没有登录,请[\[登录\]](#)或[\[注册\]](#)

* 以上用户言论只代表其个人观点，不代表CSDN网站的观点或立场

核心技术类目

全部主题 Hadoop AWS 移动游戏 Java Android iOS Swift 智能硬件 Docker OpenStack
VPN Spark ERP IE10 Eclipse CRM JavaScript 数据库 Ubuntu NFC WAP jQuery
BI HTML5 Spring Apache .NET API HTML SDK IIS Fedora XML LBS Unity
Splashtop UML components Windows Mobile Rails QEMU KDE Cassandra CloudStack
FTC coremail OPhone CouchBase 云计算 iOS6 Rackspace Web App SpringSide
Maemo Compuware 大数据 aptech Perl Tornado Ruby Hibernate ThinkPHP HBase
Pure Solr Angular Cloud Foundry Redis Scala Django Bootstrap

公司简介 | 招贤纳士 | 广告服务 | 银行汇款帐号 | 联系方式 | 版权声明 | 法律顾问 | 问题报告 | 合作伙伴 | 论坛反馈

网站客服 杂志客服 微博客服 webmaster@csdn.net 400-600-2320 | 北京创新乐知信息技术有限公司 版权所有 | 江苏乐知网络技术有限公司 提供商务支持
京 ICP 证 09002463 号 | Copyright © 1999-2014, CSDN.NET, All Rights Reserved