## 🗩 Proud Merchants HDU - 3466 (与顺序有关的01背包)

2019-09-27 19:20:31 Laaahu 阅读数 8 更多

版权声明:本文为博主原创文章,遵循 CC 4.0 BY-SA 版权协议,转载请附上原文出处链接和本声明。

本文链接: https://blog.csdn.net/laaahu/article/details/101554618

## 题目链接: vj里面的题目

题音

n件商品每件商品有三个属性,p代表的价格,q代表你至少要有多少钱才可以买这个商品q>p, v代表的是这个商品的价值,总共有m的钱,问最多可以值之和是多少。

## 思路:

假设有A B 两个商品要买pa=5,qa=6;pb=5,qb=9,m=10;

如果要从状态转移得到A是否购买,因为条件的限制关系所以 A 只有在 m=10,9,8,7,6时才会发生状态转移。

而对于B来说就是m=10,9时才会有状态转移。

第二个物品要不要买要借助于第一个物品的状态转移之后的结果,所以必须先买A再买B,如果先买B对于A来讲m=6,7,8时转台转移就没有和有所关联, 所以我们要选择合适的顺序去跑背包。

当然也与价格有关系, 当限定相同价格的物品出售的顺序不同时结果也不同,

此题策略: 限制小的价格贵的先选

也就是q-p小的先选,

A: pa,qa;

B : pb,qb;

先选A至少要pa+qb的容量,先选B至少要pb+qa的容量,

如果pa+qb>pb+qa,那么要两个都选你只能先选A再选B。例如上面的AB例子。所以按照 qa-pa<qb-pb(q1-p1<q2-p2)排序之后跑背包即可。

```
1 /* 两物品A(p1,q1) B(p2 q2) 先选A则至少需要p1+q2的容量 先选B则至少需要p2+q1 如果p1+q2>p2+q1
    那么要选两个的话的就要先选A再选B 公式可换成q1-p1<q2-p2*/
 2
 3
    #include <iostream>
 4
    #include <cstdio>
    #include <algorithm>
 6
    #include <cstring>
 7
    #include <cmath>
 8
    using namespace std;
 9
    const int maxn = 510;
10 | const int M = 5e3+10;
    struct node{
11
      int p,q,w;
12
      friend bool operator < (node a, node b){
13
           return (a.q-a.p)<(b.q-b.p);
14
15
      }
16 }arr[maxn];
17 int dp[M];
   int main()
18
19
20
        int n,m;
21
        while(~scanf("%d%d",&n,&m)){
22
            for(int i=1;i<=n;i++){</pre>
23
               scanf("%d%d%d",&arr[i].p,&arr[i].q,&arr[i].w);
24
25
           sort(arr+1,arr+n+1);
26
           memset(dp,0,sizeof(dp));
27
            for(int i=1:i<=n:i++){
28
                for(int j=m;j>=arr[i].q;j--){
                   dp[j]=max(dp[j],dp[j-arr[i].p]+arr[i].w);
29
30
31
32
            printf("%d\n",dp[m]);
33
34
    }
35
```

©2019 CSDN 皮肤主题: 大白 设计师: CSDN官方博客

猿衣醋 专属于程序员的卫衣

关闭

https://blog.csdn.net/laaahu/article/details/101554618