资源分配类动态规划

主讲老师：党东

【资源分配类动态规划】

背包类型动态规划是动态规划中的经典问题，在近几年联赛中，经常出现，甚至在NOIP2006的普及组和提高组中同时出现。

对于一般的情况，如果我们给出了某种资源，那么我们只要考虑将资源如何分配给每个阶段即可。

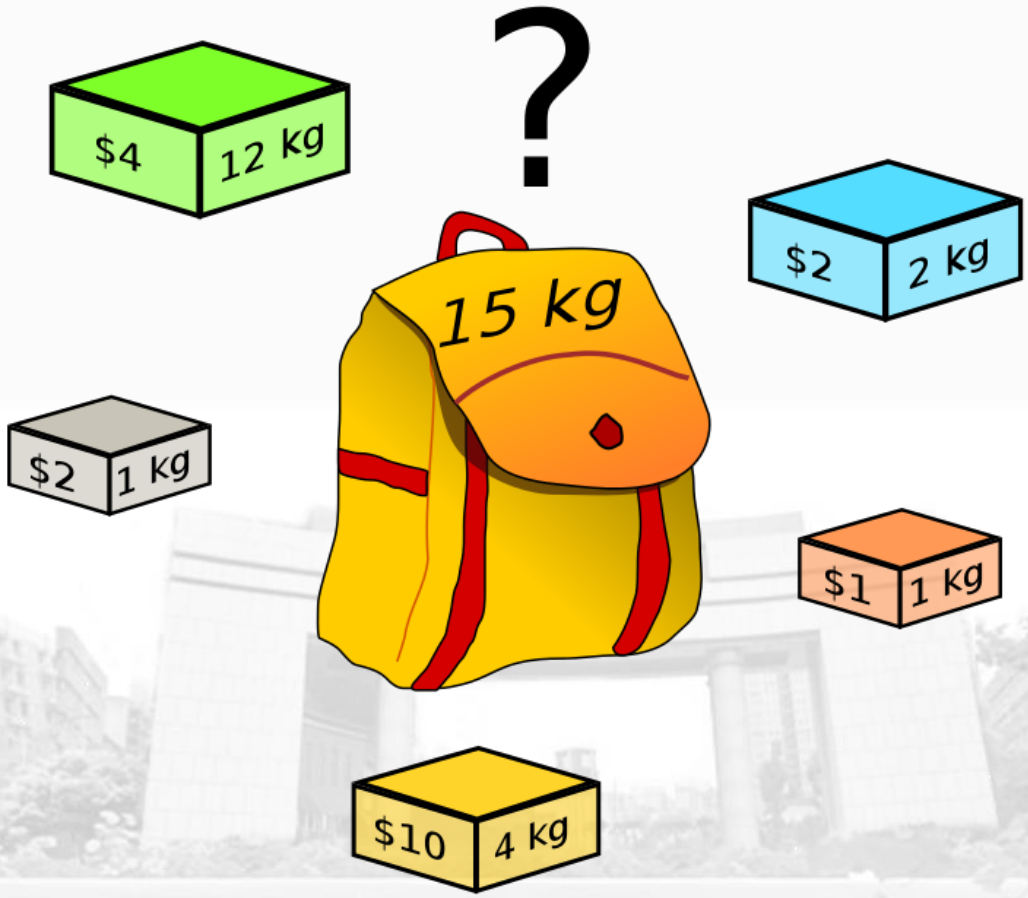
【资源分配类动态规划】

背包类DP

01背包

【什么是背包问题】

【0-1背包】假定背包的最大容量为W，N件物品，每件物品都有自己的价值和重量，将物品放入背包中使得背包内物品的总价值最大。（其他背包：物品信息和所求解不同）



【例1】01背包问题 --1364

【问题描述】在01背包问题中，需对容量为C的背包进行装载。从n个物品中选取装入背包的物品，每件物品i的重量为wi，价值为vi。对于可行的背包装载，背包中物品的总重量不能超过背包的容量，最佳装载是指所装入的物品价值最高，即**v1*x1+v2*x1+…+vi*xi**（其1<=i<=n,x取0或1，取1表示选取物品i）取得最大值。

【文件输入】

第一行：两个整数，M(背包容量，M<=200)和N(物品数量，N<=30)；

第2..N+1行：每行二个整数Wi，Vi，表示每个物品的重量和价值。。

【文件输出】仅一行，一个数，表示最大总价值。

【样例输入】

```
10 4
2 1
3 3
4 5
7 9
```

【样例输出】12

【试题特点】每种物品仅有一件,可以选择放或不放，故称作为0-1背包。

【例1】01背包问题 --1364

【题目分析】

定义：Wi是第i件物品的重量、Vi是第i件物品的价值

状态：F[i][j] 表示选择前i个物品，背包容量为j时的最大价值

F[i][j]从前一状态转移而来的情况分两种。

第 i 件物品不装：F[i-1,j]

第 i 件物品装：F[i-1][j-Wi]+Vi (j>= Wi)

（在背包中给第 i 件物品预留空间Wi，剩下的i-1件物品重新摆放（**F[i-1][j-Wi]再前一阶段计算时，已得到最优解**），最后在加上其的价值Vi）

为了得到f[i][j]的最大价值，选择**两种情况中的最大值**。

状态转移方程：F[i][j] = max(F[i-1][j], F[i-1][j-Wi]+Vi)　(j>=w[i])

【例1】01背包问题 --1364

【题目分析】

定义：Wi是第i件物品的重量、Vi是第i件物品的价值

状态：F[i][j] 表示选择前i个物品，背包容量为j时的最大价值

F[i-1][j]

将前i-1件物品放入容量为j的背包中的最大价值

F[i-1][j-Wi]　(预留放第i个物品的空间)

将前i-1件物品放入容量为j-Wi的背包中的最大价值

不装

装

F[i][j]

将前i件物品放入容量为j的背包中的最大价值

【例1】01背包问题 --1364

【题目分析】

（1）**阶段**：在前i件物品中选择若干件物品。

（2）**状态F[i][j]**：在前i个物品，背包容量为j时的最大价值。

（3）**决策**：装第i件物品或不装第i件物品。

（4）**状态转移方程**：f[i][v]=max{f[i-1][v],f[i-1][v-w[i]]+c[i](j>=w[i])}

（5）**初始条件**：f[i][0]=f[0][i]=0

【例1】01背包问题 --1364

【样例模拟】

若m=10,n=4，物品重量依次为2,3,4,7，价值依次为1,3,5,9。

可以得到所有f[i][j]的值，如下表：

	v=1	v=2	v=3	v=4	v=5	v=6	v=7	v=8	v=9	v=10
i=1										
i=2										
i=3										
i=4										

【例1】01背包问题 --1364

【参考代码】

```
for(i=1;i<=n;i++)f[0][i]=0;//初始化
for(i=1;i<=m;i++)f[i][0]=0; //初始化
for(i=1;i<=n;i++)//阶段,前i个物品
for(j=1;j<=m;j++)//状态j的重量
{       f[i][j]=f[i-1][j];  //不装
        if(j>=w[i]&&f[i][j]<f[i-1][j-w[i]]+v[i])f[i][j]= f[i-1][j-w[i]]+v[i];    //装
}
```

【例1】01背包问题 --1364

【样例模拟】

若m=10,n=4，物品重量依次为2,3,4,7，价值依次为1,3,5,9。

由样例数据可以得到所有f[i][j]的值，如下表：

0	1	2	3	4	5	6	7	8	9	10
0	0	1	1	1	1	1	1	1	1	1
0	0	1	1	7	8	8	8	8	8	8
0	0	1	1	7	8	8	8	8	8	12
0	0	1	1	7	8	8	8	8	8	12
0	0	1	1	7	8	8	8	8	8	12
0	0	1	1	7	8	8	8	8	8	12

【例1】01背包问题 --1364

【二维降一维：滚动数组优化 】

优化依据：

二维时，F[i][j]的最优值来自

F[i-1][j]和F[i-1][j-wi] （上一层）

0	1	2	3	4	5	6	7	8	9	10
0	0	1	1	1	1	1	1	1	1	1
0	0	1	1	7	8	8	8	8	8	8
0	0	1	1	7	8	8	8	8	8	12
0	0	1	1	7	8	8	8	8	8	12
0	0	1	1	7	8	8	8	8	8	12
0	0	1	1	7	8	8	8	8	8	12

优化方法：

背包的空间 J 倒序枚举，利用当前层F[...] 提供上一层的最优值

参考代码：

```
for(i=1;i<=n;i++)
        for(j=m;j>=w[i];j--)   //倒序
            if(f[j-w[i]]+v[i]>f[j])  f[j]=f[j-w[i]]+v[i];
```

【练习】采药 --1196

Description

辰辰是个天资聪颖的孩子，他的梦想是成为世界上最伟大的医师。为此，他想拜附近最有威望的医师为师。医师为了判断他的资质，给他出了一个难题。医师把他带到一个到处都是草药的山洞里对他说：“孩子，这个山洞里有一些不同的草药，采每一株都需要一些时间，每一株也有它自身的价值。我会给你一段时间，在这段时间里，你可以采到一些草药。如果你是一个聪明的孩子，你应该可以让采到的草药的总价值最大。”

如果你是辰辰，你能完成这个任务吗?

Input

输入的第一行有两个整数T（1 <= T <= 1000）和M（1 <= M <= 100），用一个空格隔开，T代表总共能够用来采药的时间，M代表山洞里的草药的数目。接下来的M行每行包括两个在1到100之间（包括1和100）的整数，分别表示采摘某株草药的时间和这株草药的价值。

【练习】采药 --1196

Output

输出包括一行，这一行只包含一个整数，表示在规定的时间内，可以采到的草药的最大总价值。

Sample Input

70 3

71 100

69 1

1 2

Sample Output

3

【练习】开心的金明 --1191

Description

金明今天很开心，家里购置的新房就要领钥匙了，新房里有一间他自己专用的很宽敞的房间。更让他高兴的是，妈妈昨天对他说：“你的房间需要购买哪些物品，怎么布置，你说了算，只要不超过N元钱就行”。今天一早金明就开始做预算，但是他想买的东西太多了，肯定会超过妈妈限定的N元。于是，他把每件物品规定了一个重要度，分为5等：用整数1~5表示，第5等最重要。他还从因特网上查到了每件物品的价格（都是整数元）。他希望在不超过N元（可以等于N元）的前提下，使每件物品的价格与重要度的乘积的总和最大。

设第j件物品的价格为v[j]，重要度为w[j]，共选中了k件物品，编号依次为j1，j2，……，jk，则所求的总和为：

v[j1]*w[j1]+v[j2]*w[j2]+ …+v[jk]*w[jk]。（其中*为乘号）

请你帮助金明设计一个满足要求的购物单。

【练习】开心的金明 --1191

Input

第1行，为两个正整数，用一个空格隔开：

N m（其中N（<30000）表示总钱数，m（<25）为希望购买物品的个数。）

从第2行到第m+1行，第j行给出了编号为j-1的物品的基本数据，每行有2个非负整数

v p （其中v表示该物品的价格(v<=10000)，p表示该物品的重要度(1~5)）

Output

输出不超过总钱数的物品的价格与重要度乘积的总和的最大值（<100000000）。

Sample Input

```
1000 5
800 2
400 5
300 5
400 3
200 2
```

Sample Output

```
3900
```

【练习】小飞侠的游园方案 --1513

Description

经过抽签选择，小智将军第一个进入考场。

菜虫：（身上散射出华贵（?）的光芒）欢迎你，第一位挑战者！！

小智：……（走到菜虫身后，关灯）女王陛下，虽然我们国家现在很富裕，但也请您不要浪费电来用这么大功率的灯泡。

菜虫（汗）：啊啊~~爱卿所言甚是~~那么，你的题目是……我们的情报组织探听到敌人的重要将领——小飞侠星期天会邀他的灵儿妹妹到公园去玩。公园里有很多娱乐项目，可并不是每一项他们都喜欢，所以他们对每一项都进行了“喜欢度”的评分。因为小飞侠也是一个了不起的角色，所以他一定会选择在有限时间内的最好的方案。现在要你做的就是找出在规定时间内他们选择哪几项不同的活动可以使其“喜欢度”之和达到最大，据此我们就可以知道他会在哪些地方出现，从而在那里派人看守了。

小智：（灯泡一亮）每个地方都派人看守不就行了？！

“当~~~”

菜虫：（手执八公分直径炒锅，筋）……你是白痴吗？-_-##（都派人去看守的话我们会有多少桌三缺一？！）听好了，输入格式是第一行一个正整数N（1 <=N<=100）表示总共的娱乐项目数；第二行一个正整数表示规定的时间t（0 < t<1000）；下面有N行，其中第i+2行有两个正整数fi（0 < =fi<=100）和ti（0 < ti<=100），分别表示对项目i的“喜欢度”和它所耗费的时间。输出的时候在第一行输出最大的“喜欢度”之和

【练习】小飞侠的游园方案 --1513

Input

3

5

1 2

5 5

4 3

Output

5

【变形】装箱问题 --1213

Description

有一个箱子容量为v(正整数，o≤v≤20000)，同时有n个物品(o≤n≤30)，每个物品有一个体积 (正整数)。要求从m个物品中，任取若干个装入箱内，使箱子的剩余空间为最小。

Input

第一行为箱子容量为v；第二行为物品数；以下n行分别为每个物品的体积；

Output

箱子剩余空间

Sample Input

24
6
8
3
12
7
9
7

Sample Output

0

【变形】装箱问题 --1213

【题目分析】

本题是01背包的简单变形，由求最大价值变为求箱子中剩余的最小空间。

设一个布尔型数组f[i][j]表示从前i个物品中是否装满容量为j的空间，若值为true，则能装满；否则不能装满。

根据上面的递推关系，可以得到如下的状态转移方程：

f[i][j]=f[i-1][j] or f[i-1][j-v[i]] (1<=i<=n,0<=j<=v)

【变形】装箱问题 --1213

【参考代码】优化到一维时，状态转移方程为：f[j]=f[j] or f[j-v[i]]

```
#include<iostream>
using namespace std;
int v,n,i,j,a[31],f[20005];
int main()
{  cin>>v>>n;                                    //箱子容量和物品数
   for(i=1;i<=n;i++)scanf("%d",&a[i]);    //每个物品体积
   f[0]=1; //初值
   for(i=1;i<=n;i++)        //前i个物品
       for(j=v;j>=a[i];j--)   f[j]=f[j]||f[j-a[i]];
   for(i=v;i>=0;i--)if(f[i]==1){cout<<v-i;break;}
   return 0;
}
```

【例2】金明的预算方案 --1514

Description

金明今天很开心，家里购置的新房就要领钥匙了，新房里有一间金明自己专用的很宽敞的房间。更让他高兴的是，妈妈昨天对他说：“你的房间需要购买哪些物品，怎么布置，你说了算，只要不超过N元钱就行”。今天一早，金明就开始做预算了，他把想买的物品分为两类：主件与附件，附件是从属于某个主件的，下表就是一些主件与附件的例子：

主件	附件
电脑	打印机，扫描仪
书柜	图书
书桌	台灯，文具
工作椅	无

如果要买归类为附件的物品，必须先买该附件所属的主件。每个主件可以有0个、1个或2个附件。附件不再有从属于自己的附件。金明想买的东西很多，肯定会超过妈妈限定的N元。于是，他把每件物品规定了一个重要度，分为5等：用整数1~5表示，第5等最重要。他还从因特网上查到了每件物品的价格（都是10元的整数倍）。他希望在不超过N元（可以等于N元）的前提下，使每件物品的价格与重要度的乘积的总和最大。

设第j件物品的价格为v[j]，重要度为w[j]，共选中了k件物品，编号依次为j1，j2，……，jk，则所求的总和为：v[j1]*w[j1]+v[j2]*w[j2]+ …+v[jk]*w[jk]。（其中*为乘号）请你帮助金明设计一个满足要求的购物单。

【例2】金明的预算方案 --1514

Input

输入文件的第1行，为两个正整数，用一个空格隔开：

N m　其中N（<32000）表示总钱数，m（<60）为希望购买物品的个数。）

从第2行到第m+1行，第j行给出了编号为j-1的物品的基本数据，每行有3个非负整数

v p q （其中v表示该物品的价格（v<10000），p表示该物品的重要度（1~5），q表示该物品是主件还是附件。如果q=0，表示该物品为主件，如果q>0，表示该物品为附件，q是所属主件的编号）

Output

输出文件只有一个不超过总钱数的物品的价格与重要度乘积的总和的最大值（<200000）。

Sample Input

```
1000 5
800 2 0
400 5 1
300 5 1
400 3 0
500 2 0
```

Sample Output

```
2200
```

【例2】金明的预算方案 --1514

【题目分析】

由题意可知，本题如果没有主附件关系，则是一个典型的01背包问题。

即使有主附件关系，通过一定的预处理，亦能转化为01背包问题。

预处理：

针对主件 i 生成搭配方案，由题可知，每个主件最多2个附件。

1.主件

2.主件+附件1

3.主件+附件2

4.主件+附件1+附件2

【例2】金明的预算方案 --1514

【题目分析】

预处理:

假如将4种搭配方案均看作新的物品?

1.主件　　2.主件+附件1　　3.主件+附件2　　4.主件+附件1+附件2

则需求出其对应的价格和重要度。

即定义:

V[i][j] 表示 主件 i 对应第 j 种搭配情况下的 价格

P[i][j] 表示 主件 i 对应第 j 种搭配情况下的 重要度

由此，每种搭配方案均可直接使用了。

【例2】金明的预算方案 --1514

【题目分析】

阶段：前i个物品　（前i个物品中，只处理主件，因为附件已经包括在搭配方案中）

状态：容量

决策：选择该主件的哪种附件搭配方案能够得到最优值

状态转移：

```
for(k=1;k<=d[i];k++)    //决策：选择该主件的哪个搭配方案
    if(j>=v[i][k]&&f[j-v[i][k]]+p[i][k]>f[j])
        f[j]=f[j-v[i][k]]+p[i][k];
```

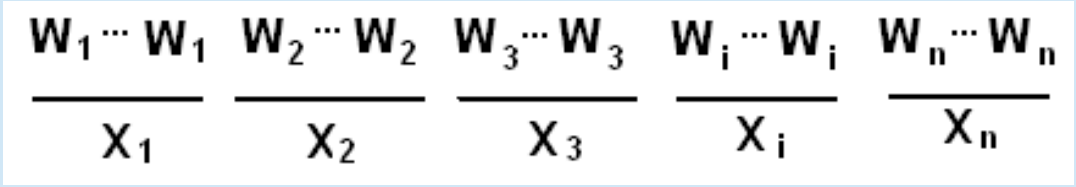
【扩展】多层背包问题

【问题描述】有N件物品;第i件物品Wi公斤;第i件物品价值Ci元;现有一辆载重M公斤的卡车;第i件物品限制最多只能取Xi个;

问选取装载哪些物品，使得卡车运送的总价值最大?

【思路分析】

我们可以类似01背包问题，将第i种物品拆分成x[i]个，这样每个物品只有1件了,如下图:



然后对所有的物品采用动态规划即可。

F(i,j)=max{ f(i-1,j-k*w[i]) + k*c[i] }

0<=k*w[i]<=j

完全背包

【例1】完全背包问题 --1365

Description

设有n种物品，每种物品有一个重量及一个价值。但每种物品的数量是无限的，同时有一个背包，最大载重量为M，今从n种物品中选取若干件(同一种物品可以多次选取)，使其重量的和小于等于M，而价值的和为最大。

Input

第一行：两个整数，M(背包容量，M<=200)和N(物品数量，N<=30)；

第2..N+1行：每行二个整数Wi,Ci，表示每个物品的重量和价值。

Output

仅一行，一个数，表示最大总价值。

Sample Input

10 4
2 1
3 3
4 5
7 9

Sample Output

max=12

【例1】完全背包问题 --1365

【题目分析】

这个问题非常类似于01背包问题，所不同的是每种物品有无限件。也就是从每种物品的角度考虑，与它相关的策略已并非取或不取两种，而是有取0件、取1件、取2件……等很多种。

1、阶段和状态：

f[i][j]：表示在前i种物品用了j重量的资源后所能得到的最大价值；

2、状态转移方程：

f[i][j]=max{f[i-1][j],f[i][j-w[i]]+v[i]](1<=i<=n,w[i]<=j<=m)

answer=f[n][m]

【例1】完全背包问题 --1365

【推导过程】

	j=1	j=2	j=3	j=4	j=5	j=6	j=7	j=8	j=9	j=10
i=1	0	1	1	2	2	3	3	4	4	5
i=2	0	1	3	3	4	6	6	7	9	9
i=3	0	1	3	5	5	6	8	10	10	11
i=4	0	1	3	5	5	6	9	10	10	12

```
int m,n,f[201]={0},i,j,w[201],v[201];
 cin>>m>>n;   //容量和物品数目
 for(i=1;i<=m;i++)f[i]=0;  //初始化
 for(i=1;i<=n;i++)cin>>w[i]>>v[i];  //重量和价格
 for(i=1;i<=n;i++)  //物品数
     for(j=w[i];j<=m;j++)  //容量
         f[j]=max(f[j],f[j-w[i]]+v[i]);
 cout<<f[m]<<endl;
```

【练习】无限硬币问题 --1515

Description

输入硬币的n种不同面值(各种面值的硬币个数不限)和m，输出构成1到m元的最少硬币数。

Input

第一行两个数n(n<=200),m(m<=10000)

第二行n个面值

Output

共m行，每行为组成该值所需最少硬币数(若不能组成则输出-1)

Sample Input

3 4

1 2 3

Sample Output

1 换行 1 换行 1 换行 2