# 泛化物品背包（DP）

考虑这样一种物品，它并没有固定的费用和价值，而是它的价值随着你分配给它的费用而变

化。这就是泛化物品的概念。

更严格的定义之。在背包容量为 V 的背包问题中，泛化物品是一个定义域为 0..V 中的整数的

函数 h，当分配给它的费用为 v 时，能得到的价值就是 h(v)。

这个定义有一点点抽象，另一种理解是一个泛化物品就是一个数组 h[0..V]，给它费用v，可

得到价值 h[V]。

一个费用为 c 价值为 w 的物品，如果它是 01 背包中的物品，那么把它看成泛化物品，它就是除了h(c)=w 其它函数值都为 0 的一个函数。如果它是完全背包中的物品，那么它可以看成这

样一个函数，仅当 v 被 c 整除时有 h(v)=v/c\*w，其它函数值均为 0。如果它是多重背包中重复次数最多为 n 的物品，那么它对应的泛化物品的函数有 h(v)=v/c\*w 仅当 v 被 c 整除且

v/c<=n，其它情况函数值均为 0。

一个物品组可以看作一个泛化物品h。对于一个 0..V 中的 v，若物品组中不存在费用为v的

物品，则 h(v)=0，否则 h(v)为所有费用为 v 的物品的最大价值。 P07 中每个主件及其附件集合等价于一个物品组，自然也可看作一个泛化物品。

## **泛化物品的和**

如果面对两个泛化物品 h 和 l，要用给定的费用从这两个泛化物品中得到最大的价值，怎么

求呢？事实上，对于一个给定的费用 v，只需枚举将这个费用如何分配给两个泛化物品就可 以了。同样的，对于 0..V 的每一个整数 v，可以求得费用 v 分配到 h 和 l 中的最大价值f(v)。也即 f(v)=max{h(k)+l(v-k) |0<=k<=v}。可以看到， f 也是一个由泛化物品 h 和 l 决定的定义域为 0..V 的函数，也就是说， f 是一个由泛化物品 h 和 l 决定的泛化物品。

由此可以定义泛化物品的和： h、 l 都是泛化物品，若泛化物品 f 满足 f(v)=max{h(k)+l(v-k) |0<=k<=v}，则称 f 是 h 与 l 的和，即 f=h+l。这个运算的时间复杂度取决于背包的容量，

是 O(V^2)。

泛化物品的定义表明：在一个背包问题中，若将两个泛化物品代以它们的和，不影响问题的答案。事实上，对于其中的物品都是泛化物品的背包问题，求它的答案的过程也就是求所有这些泛化物品之和的过程。设此和为 s，则答案就是 s[0..V]中的最大值。

背包问题的泛化物品一个背包问题中，可能会给出很多条件，包括每种物品的费用、价值等属性，物品之间的分组、依赖等关系等。但肯定能将问题对应于某个泛化物品。也就是说，给定了所有条件以后，就可以对每个非负整数 v 求得：若背包容量为 v，将物品装入背包可得到的最大价值是多少，这可以认为是定义在非负整数集上的一件泛化物品。这个泛化物品——或者说问题所对应的一个定义域为非负整数的函数——包含了关于问题本身的高度浓缩的信息。一般而言，求得这个泛化物品的一个子域（例如 0..V）的值之后，就可以根据这个函数的取值得到背包问题的最终答案。综上所述， 一般而言，求解背包问题，即求解这个问题所对应的一个函数，即该问题的泛化物品。而求解某个泛化物品的一种方法就是将它表示为若干泛化物品的和然后求之。

# **例题**

[最佳课程选择](https://www.luogu.org/problemnew/show/P1336" \t "https://blog.csdn.net/qq_39670434/article/details/_blank)

****题目描述****

Matrix67要在下个月交给老师n篇论文，论文的内容可以从m个课题中选择。由于课题数有限，Matrix67不得不重复选择一些课题。完成不同课题的论文所花的时间不同。具体地说，对于某个课题i，若Matrix67计划一共写x篇论文，则完成该课题的论文总共需要花费Ai\*x^Bi个单位时间（系数Ai和指数Bi均为正整数）。给定与每一个课题相对应的Ai和Bi的值，请帮助Matrix67计算出如何选择论文的课题使得他可以花费最少的时间完成这n篇论文。

****输入输出格式****

输入格式：

第一行有两个用空格隔开的正整数n和m，分别代表需要完成的论文数和可供选择的课题数。

以下m行每行有两个用空格隔开的正整数。其中，第i行的两个数分别代表与第i个课题相对应的时间系数Ai和指数Bi。

输出格式：

输出完成n篇论文所需要耗费的最少时间。

# **题解**

这道题目就是一个泛化物品的例子。就是一个01背包+有依赖的背包。怎么理解呢。如果我们选一门课要写3篇文章，这三篇文章的耗时是不一样的，所以我们给它们定一个顺序（这里称为1、2、3号），我们必须先选1号，再选2号，才能选3号，所以可以看做三个价值不一样的物品；而且2号依赖于1号，3号依赖于2号。所以我们就建立起了一个有依赖的背包问题模型。

#include<bits/stdc++.h>

using namespace std;

inline int read(){

int num=0;

char c=' ';

bool flag=true;

for(;c>'9'||c<'0';c=getchar())

if(c=='-')

flag=false;

for(;c>='0'&&c<='9';num=num\*10+c-48,c=getchar());

return flag ? num : -num;

}

long long qpow(int i,int b)

{*//i 鐨?b 娆℃柟*

long long ans=1;

long long base=i,times=1;

while(times<=b)

{

if(times&b)

{

ans\*=base;

}

base\*=base;

times<<=1;

}

return ans;

}

const int maxn=500;

int v,n;

const int INF=2e9;

long long dp[maxn],t[maxn][maxn];

void init(){

v=read();

n=read();

memset(dp,10,sizeof dp);

dp[0]=0;

for(int k=1;k<=n;k++)

{

int a=read();

int b=read();

for(int i=1;i<=v;i++)

t[k][i]=a\*qpow(i,b);

}*//预处理所有的情况*

}

void DP(){

for(int i=1;i<=n;i++)

for(int j=v;j>=0;--j)

for(int k=1;k<=j;k++)

dp[j]=min(dp[j],dp[j-k]+t[i][k]);*//01背包*

printf("%d\n",dp[v]);

}

int main(){

init();

DP();

return 0;

}