**最少费用购物问题建模**

**摘要**

现今社会网络越来越普及，网购已经成为一种常用的消费方式，各类电商平台蓬勃发展，同一件商品在不同平台会出现不同的价格，如何让消费者花最少费用购物

**一、问题背景**

现代社会网络普及，网购以方便快捷的好处风靡全国，成为人们的一种消费方式，网上商品越来越多，为了能以最优惠的价格购买到喜欢的商品也成为人们所考虑的问题，针对本案例，采用科学分析的方法并反复进行实践。

**二、问题描述**

淘宝、京东等店铺中每种商品都有标价。例如，一台小米9手机的价格是2600元，一个手机壳的价格是30元。为了吸引顾客，商家提供许多优惠方案，优惠商品把一种或多种商品分成一组，并降价销售。例如，一台小米9手机加一个手机壳的优惠价是2610元。试设计一个算法，计算出某一顾客所购商品应付的最少费用。  
 编程任务：对于给定欲购商品的价格和数量，以及优惠商品价，编程计算所购商品应付的最少费用。

数据输入：由文件input.txt提供欲购商品数据。

文件的第1行中有1 个整数B（0≤B≤5），表示所购商品种类数。

接下来的B 行，每行有3 个数C，K 和P。C 表示商品的编码（每种商品有唯一编码），1≤C≤999。K 表示购买该种商品总数，1≤K≤5。P 是该种商品的正常单价（每件商品的价格），1≤P≤999。请注意，一次最多可购买5\*5＝25件商品。由文件offer.txt提供优惠商品价数据。

文件的第1行中有1 个整数S（0≤S≤99），表示共有S 种优惠商品组合。  
接下来的S 行，每行的第一个数描述优惠商品组合中商品的种类数j。接着是j 个数字对（C，K），其中C 是商品编码，1≤C≤999。K 表示该种商品在此组合中的数量，1≤K≤5。每行最后一个数字P（1≤ P≤9999）表示此商品组合的优惠价。

结果输出:程序运行结束时，将计算出的所购商品应付的最少费用输出到文件output.txt中。

输入文件示例

input.txt

2

7 3 2

8 2 5

offer.txt

2

1 7 3 5

2 7 1 8 2 10

输出文件示例

output.txt

**三、问题分析**

**分析此类问题，我认为跟背包问题差不多。只是背包问题是给定容量，求最大价值的东西。而这道题目是给定所放的东西，求最小的费用（对应背包问题为最小的容量）。恰好是一个求最值的“逆问题”，由于动态规划要满足无后效性和最优化原理，所以我们来分析此题是否满足以上两点。先来状态表示的方法，商品不超过5种，而每种采购的数量又不超过5，那么用一个五元组来表示第I种商品买AI的最小费用。：**

**F（A1，A2，A3，A4，A5） （1）**

**考虑这个状态的由来，当然，我们不用优惠商品也可以买，显然这样不是最优。于是如果我们能够使用第I条商品组合的话，状态就便为了：**

**F（A1-SI1，A2-SI2，A3-SI3，A4-SI4，A5-SI5） （2）**

**这样的话，状态1的费用为状态2的费用加上SI的费用，而状态2的费用必须最低（很显然，用反证法即可），同时，我们也不管状态2是如何来的（因为每一个优惠商品组合的使用是没有限制的），所以本题既满足无后效性，又符合最优化原理，同时还有大量重叠子问题产生，动态规划解决此题是最好不过了。**

**通过对问题的分析，我们知道了状态的表示和转移的基本方法，我们很容易得到一个状态转移方程：**

**F [a, b, c, d, e] = Min {F [a-S1, b-S2, c-S3, d-S4, e-S5] + SaleCost [S]}**

**初始条件为：**

**F [a, b, c, d, e] = Cost [1]\*a+Cost [2]\*b+Cost [3]\*c+Cost [4]\*d+Cost [5]\*e**

**即不用优惠的购买费用。**

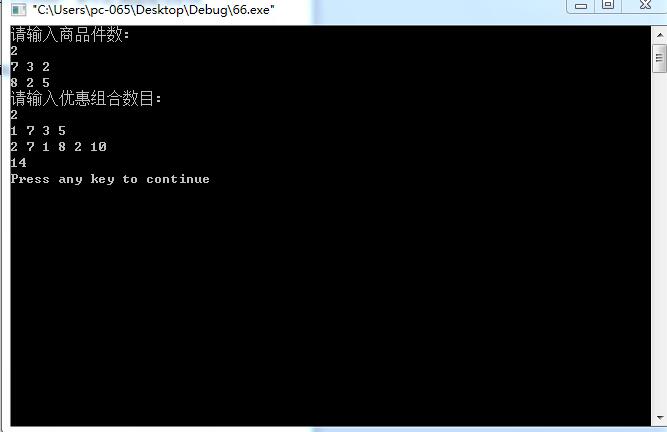
此问题看起来除了进行穷举求最小购物费用之外，似乎没有更好的方法。但是使用动态规划中的备忘录法减少穷举过程中的重复计算。穷举过程的递归式为cost[A][B][C][D][E]=min{cost[A-Ai][B-Bi][C-Ci][D-Di][E-Ei]+Pi},0=

Ai,Bi,Ci,Di,Ei,为第i种优惠方案中，商品种类1-5对应的组合数量，Pi为第i种组合优惠方案的优惠价格

**四、算法设计**

#include<stdio.h>  
#include<string.h>  
#define maxb 6 //所购商品种类数(0<=B<=5)  
#define maxs 100//优惠商品种类数(0<=S<=99)  
int purch[maxb][2];//存放预购商品数据  
int offer[maxs][maxb];//存放优惠商品价数据  
int product[maxb];  
int num[1000];  
int cost[maxb][maxb][maxb][maxb][maxb];  
int b,s;//b件商品,s种组合  
  
void init()  
{  
int i,j,k;  
int code,t,p;//code表编号,t表示优惠商品组合  
memset(offer,0,sizeof(offer));  
    memset(purch,0,sizeof(purch));  
    memset(product,0,sizeof(product));  
printf("请输入商品件数:");  
scanf("%d",&b);  
for(i=1;i<=b;i++)  
{  
scanf("%d %d %d",&code,&purch[i][0],&purch[i][1]);//code表示商品的编码;存放购买该种商品的总数;该商品的正常单价  
num[code]=i;//商品的编号num[2]=1,num[8]=2  
}  
   printf("请输入优惠组合数目:");  
   scanf("%d",&s);  
   for(i=1;i<=s;i++)  
   {  
   scanf("%d",&t);  
   for(j=1;j<=t;j++)  
   {  
   scanf("%d %d",&code,&p);  
   offer[i][num[code]]=p;//第i种优惠组合中编号为code商品的数量  
   }  
    scanf("%d",&offer[i][0]);//利用0下标存第i种优惠组合的总价格  
   }  
}  
  
  
void mincost()  
{  
int i,min;  
int A,B,C,D,E;  
min=0;  
    for(i=1;i<=b;i++)  
{  
min=min+(product[i]\*purch[i][1]);//正常单价购买  
}  
//遍历S种优惠组合  
   for(i=1;i<=s;i++)  
   {    //五种产品减去采用优惠组合的数量  
   A=product[1]-offer[i][1];  
   B=product[2]-offer[i][2];  
       C=product[3]-offer[i][3];  
       D=product[4]-offer[i][4];  
       E=product[5]-offer[i][5];  
 if(A>=0&&B>=0&&C>=0&&D>=0&&E>=0&&min>(cost[A][B][C][D][E]+offer[i][0]))  
 {  
    min=cost[A][B][C][D][E]+offer[i][0];  
 }  
}  
   cost[product[1]][product[2]][product[3]][product[4]][product[5]]=min;  
}  
  
void dp(int i)  
{  
int j;  
if(i>b){  
mincost();  
return;  
}  
for(j=0;j<=purch[i][0];j++)  
{  
  
product[i]=j;  
  
dp(i+1);  
}  
}  
int main(int argc, char \*argv[])  
{  
int A,B,C,D,E;  
init();  
dp(1);  
A=purch[1][0];  
B=purch[2][0];  
C=purch[3][0];  
D=purch[4][0];  
E=purch[5][0];  
printf("%d\n",cost[A][B][C][D][E]);  
return 0;  
}

**五、问题实现结果**

****