# 最少费用购物问题算法设计

目录

[最少费用购物问题算法设计 1](#_Toc8983341)

[一、问题描述 1](#_Toc8983342)

[二、数据输入 1](#_Toc8983343)

[三、代码实现 2](#_Toc8983344)

[四、实现输出 7](#_Toc8983345)

### 一、问题描述

商店中每种商品都有标价。例如，一朵花的价格是 2 元。一个花瓶的价格是 5 元。为了吸引顾客，商店提供了一组优惠商品价。优惠商品是把一种或多种商品分成一组，并降价销 售。例如，3 朵花的价格不是 6 元而是 5 元。2 个花瓶加 1 朵花的优惠价是 10 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。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。

### 二、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。

### 三、代码实现

import java.util.Scanner;

class Commodity{

int piece;//购买数量

int price;//购买价格

}

public class ZuiShaoFeiYongGouWu {

private static int MAXCODE = 999;//商品编码的最大值

private static int SALECOMB = 99;//优惠商品组合数

private static int KIND = 5; //商品种类

private static int QUANTITY = 5; //购买某种商品数量的最大值

private static int b;//购买商品种类数

private static int s;//当前优惠组合数

private static int[] num = new int[MAXCODE+1];//记录商品编码与商品种类的对应关系

private static int[] product = new int[KIND+1];//记录不同种类商品的购买数量

private static int[][] offer = new int[SALECOMB+1][KIND+1];//offer[i][j]: 商品组合的优惠价(j=0)；某种优惠组合中某种商品需要购买的数量(j>0)

private static Commodity[] purch = new Commodity[KIND+1];//记录不同商品的购买数量和购买价格

private static int[][][][][] cost = new int[QUANTITY+1][QUANTITY+1][QUANTITY+1][QUANTITY+1][QUANTITY+1];//记录本次购买的总花费

public static void main(String[] args){

init();

comp(1);

out();

}

private static void minicost(){

int i,j,k,m,n,p,minm;

minm = 0;

for(i=1; i<=b; i++)

minm += product[i]\*purch[i].price;

for(p=1; p<=s; p++){

i = product[1] - offer[p][1];

j = product[2] - offer[p][2];

k = product[3] - offer[p][3];

m = product[4] - offer[p][4];

n = product[5] - offer[p][5];

if(i>=0 && j>=0 && k>=0 && m>=0 && n>=0 && cost[i][j][k][m][n]+offer[p][0] < minm)

minm = cost[i][j][k][m][n] + offer[p][0];

}

cost[product[1]][product[2]][product[3]][product[4]][product[5]] = minm;

}

private static void init(){

Scanner input = new Scanner(System.in);

int i,j,n,p,t,code;

for(i=0; i<100; i++)

for(j=0; j<6; j++)

offer[i][j] = 0;

for(i=0; i<6; i++){

purch[i] = new Commodity();

purch[i].piece = 0;

purch[i].price = 0;

product[i] = 0;

}

b = input.nextInt();

for(i=1; i<=b; i++){

code = input.nextInt();

purch[i].piece = input.nextInt();

purch[i].price = input.nextInt();

num[code] = i;

}

s = input.nextInt();

for(i=1; i<=s; i++){

t = input.nextInt();

for(j=1; j<=t; j++){

n = input.nextInt();

p = input.nextInt();

offer[i][num[n]] = p;

}

offer[i][0] = input.nextInt();

}

}

private static void comp(int i){

if(i > b){

minicost();

return;

}

for(int j=0; j<=purch[i].piece; j++){

product[i] = j;

comp(i+1);

}

}

private static void out(){

System.out.println(cost[product[1]][product[2]][product[3]][product[4]][product[5]]);

}

}

### 四、实现输出

2

7 3 2

8 2 5

2

1 7 3 5

2 7 1 8 2 10

14