



西安交通大学
XI'AN JIAOTONG UNIVERSITY

本科生上机报告
最少费用购物问题

课程： 算法分析与设计

姓名： 杨豪

班级： 软件 2101

时间： 2022 年 10 月

目录

1	问题描述	1
1.1	编程任务	1
1.2	数据输入	1
1.3	结果输出	1
1.4	示例	2
2	问题分析	2
2.1	最优子结构的证明	2
2.2	代码实现	2
3	运行结果展示	7

1 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。

1.1 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。

1.2 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 $B(0 \leq B \leq 5)$, 表示所购商品种类数。在接下来的 B 行中,每行有 3 个数 C, K 和 P 。 C 表示商品的编码(每种商品有惟一编码), $1 \leq C \leq 999$; K 表示购买该种商品总数, $1 \leq K \leq 5$; P 是该种商品的正常单价(每件商品的价格), $1 \leq P \leq 999$ 。请注意,一次最多可购买 $5 \times 5 = 25$ 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 $S(0 \leq S \leq 99)$, 表示共有 S 种优惠商品组合。接下来的 S 行,每行的第 1 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, $1 \leq C \leq 999$, K 表示该种商品在此组合中的数量, $1 \leq K \leq 5$ 。每行最后一个数字 $P(1 \leq P \leq 9999)$ 表示此商品组合的优惠价。

1.3 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。

1.4 示例

输入文件示例		输出文件示例
input.txt	offer.txt	output.txt
2	2	14
7 3 2	1 7 3 5	
8 2 5	2 7 1 8 2 10	

2 问题分析

2.1 最优子结构的证明

设 C 为花费的最少费用, 参数为五个欲购商品分别的数量, 共有 p 种优惠方案, 第 x 种的商品 O 数量为 $O[x]$, 总价格为 C_x

$$C(j, k, l, m, n) = \min_{x=1}^p C(j - J[x], k - K[x], l - L[x], m - M[x], n - N[x]) + C_x$$

可得, 该问题的解满足最优子结构的要求

2.2 代码实现

Listing 1: 3-14.cpp

```

1 #include<stdio.h>
2 #include<stdlib.h>
3
4 #define N 5//每种商品的最大数量
5 #define B 5//最大商品种类数
6 #define S 99//最大的优惠组合数
7 #define C 999//最大的编号
8
9 int cost[N + 1][N + 1][N + 1][N + 1][N + 1] = { 0 };
10 // cost(a,b,c,d,e)表示购买商品 a b c d e 组合的最少花费. 从 1 开始
11 int offer[S][B + 1];

```

```
12 // offer[Si][Bj] Si组合中的Bj(从 1 开始)类商品的数量 offer[1..S][0]存储这
    种组合的花费
13
14 struct Purch
15 {
16     int code;//编号
17     int quatity;//要购买的数量
18     int price;//单价
19 } purch[B];//定义购买物品信息数组
20
21 int product[B] = { 0 };//存储当前已购买的商品数量
22 int num[C] = { -1 }; //code编号的商品对应的在purch数组中的index
23
24 int b;//实际购买的商品种类
25
26 int s;//实际的优惠组合数
27
28 // 初始化
29 void init()
30 {
31     FILE *input;
32     FILE *offerFile;
33     // 打开文件
34     input = fopen("input.txt", "r");//open file
35     if (input == NULL)//open file error
36     {
37         printf("can't not open input.txt!\n");
38         exit(1);
39     }
40     offerFile = fopen("offer.txt", "r");
```

```
41     if (offerFile == NULL)//open file error
42     {
43         printf("can't not open offer.txt!\n");
44         exit(1);
45     }
46     // 读取数据并初始化
47     for (int i = 0; i < N; ++i)
48     {
49         purch[i].code = -1;
50         purch[i].price = 0;
51         purch[i].quantity = 0;
52     }
53     for (int i = 0; i < S; ++i) for (int j = 1; j <= B; ++j) offer[i][j]
        = 0;
54     //读入文件数据
55     fscanf(input, "%d", &b);//实际商品种类
56     for (int i = 0; i < b; ++i)
57     {
58         fscanf(input, "%d %d %d", &purch[i].code, &purch[i].quantity
            , &purch[i].price);//商品编码 数量 单价
59         num[purch[i].code] = i;
60     }
61     fscanf(offerFile, "%d", &s);//实际组合数
62     for (int i = 0; i < s; ++i)
63     {
64         int pairs;//组合中的商品数
65         fscanf(offerFile, "%d", &pairs);
66         for (int j = 0; j < pairs; ++j)
67         {
68             int c;//商品编号
```

```
69         int n;//组合i中c编号商品的数量
70         fscanf(offerFile, "%d %d", &c, &n);
71         offer[i][num[c] + 1] = n;
72     }
73
74     fscanf(offerFile, "%d", &offer[i][0]); //组合的花费
75 }
76 fclose(input);
77 fclose(offerFile);
78 }
79
80 void printResult()
81 {
82     FILE *out = fopen("output.txt", "w");
83     if (out == NULL){
84         printf("can't not open output.txt to write!\n");
85         exit(1);
86     }
87     int mincost = cost[purch[0].quantity][purch[1].quantity][purch[2].
        quantity]
88     [purch[3].quantity][purch[4].quantity];
89     printf("MinCost=%d\n", mincost);
90     fprintf(out, "%d", mincost);
91 }
92
93 void minCost(){
94     int mincost = 0;
95     int quatity1;//已经购买的第一种商品的量
96     int quatity2;
97     int quatity3;
```

```

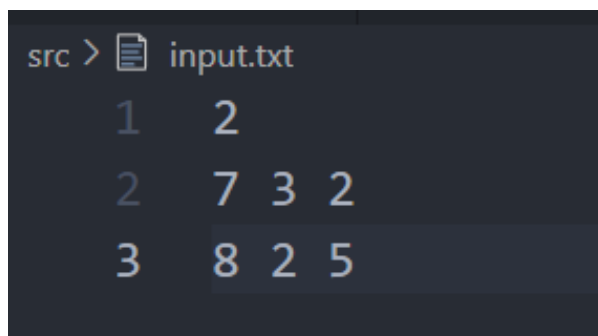
98     int quatity4;
99     int quatity5;
100
101     for (int i = 0; i<b; ++i) mincost += product[i] * purch[i].price;
102     //将最小花费初始为没有优惠策略的花费
103     //对s种优惠政策依次讨论
104     for (int j = 0; j<s; ++j){
105         quatity1 = product[0] - offer[j][1]; //第一种商品扣除当前优
106             惠组合下的购买量的其它购买量
107         quatity2 = product[1] - offer[j][2];
108         quatity3 = product[2] - offer[j][3];
109         quatity4 = product[3] - offer[j][4];
110         quatity5 = product[4] - offer[j][5];
111         if (quatity1 >= 0 && quatity2 >= 0 && quatity3 >= 0 &&
112             quatity4 >= 0 && quatity5 >= 0
113             && cost[quatity1][quatity2][quatity3][quatity4][
114                 quatity5] + offer[j][0]<mincost)
115             //在当前优惠组合下，购买的商品的总花费比没有优惠政策的少
116             mincost = cost[quatity1][quatity2][quatity3][
117                 quatity4][quatity5] + offer[j][0];
118             //购买组合前的总花费，加上这种组合的花费
119     }
120     cost[product[0]][product[1]][product[2]][product[3]][product[4]] =
121         mincost;
122     //子问题最优组合花费
123 }
124
125 void combination(int i){
126     if (i >= b){

```


3 运行结果展示

```
123     minCost();
124     return;
125 }
126     for (int j = 0; j <= purch[i].quantity; ++j){
127         product[i] = j;
128         combination(i + 1);
129     }
130 }
131
132 int main()
133 {
134     init();
135     combination(0); //从第0种商品开始
136     printResult();
137     return 0;
138 }
```

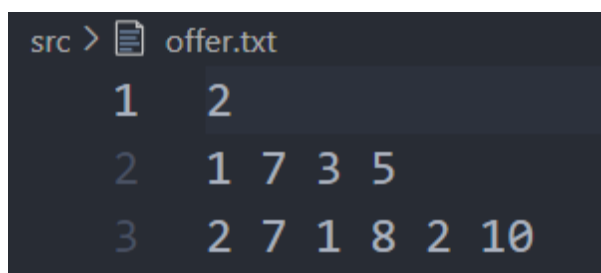
3 运行结果展示



```
src > input.txt
1 2
2 7 3 2
3 8 2 5
```

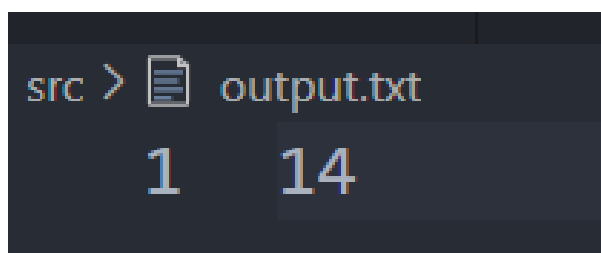
图 1: input.txt

3 运行结果展示

A screenshot of a text editor showing the content of a file named offer.txt. The file contains three lines of numbers. The first line has '1' and '2'. The second line has '2', '1', '7', '3', and '5'. The third line has '3', '2', '7', '1', '8', '2', and '10'.

```
src > offer.txt
1 2
2 1 7 3 5
3 2 7 1 8 2 10
```

图 2: offer.txt

A screenshot of a text editor showing the content of a file named output.txt. The file contains one line with the number '1' followed by '14'.

```
src > output.txt
1 14
```

图 3: output.txt

结果与示例相符