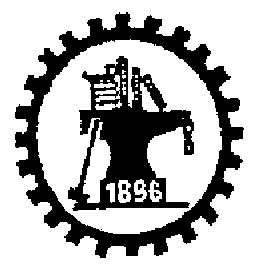
****

**算法设计与分析上机实验报告**



**专业/班级**   **软件41**

**学 号**   **2141601019**

**学生姓名**  **祝超**

**提交日期**   **2016.12.22**

目录

[一、最少费用购物问题 3](#_Toc470214609)

[问题分析 4](#_Toc470214610)

[代码片段 5](#_Toc470214611)

[运行截图 6](#_Toc470214612)

[二、收集样本问题 6](#_Toc470214613)

[问题分析 7](#_Toc470214614)

[代码片段 8](#_Toc470214615)

[运行截图 9](#_Toc470214616)

[三、罗密欧与朱丽叶的迷宫问题 9](#_Toc470214617)

[问题分析 10](#_Toc470214618)

[代码片段 11](#_Toc470214619)

[运行截图 12](#_Toc470214620)

[四、汽车加油问题 12](#_Toc470214621)

[问题分析 13](#_Toc470214622)

[代码片段 13](#_Toc470214623)

[运行截图 13](#_Toc470214624)

[五、运动员最佳配对问题 13](#_Toc470214625)

[问题分析 14](#_Toc470214626)

[代码片段 15](#_Toc470214627)

[运行截图 15](#_Toc470214628)

[六、皇后控制问题 16](#_Toc470214629)

[问题分析 16](#_Toc470214630)

[代码片段 17](#_Toc470214631)

[运行截图 19](#_Toc470214632)

## 一、最少费用购物问题

**★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。

**★算法设计**：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

14

### 问题分析

这道题目和背包问题有些类似。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”。背包问题是经典的动态规划问题，那么这道题呢？

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）

Ai为商品种类1-5对应的组合数量

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）

Sij为第i种组合优惠方案的商品j的数量

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：

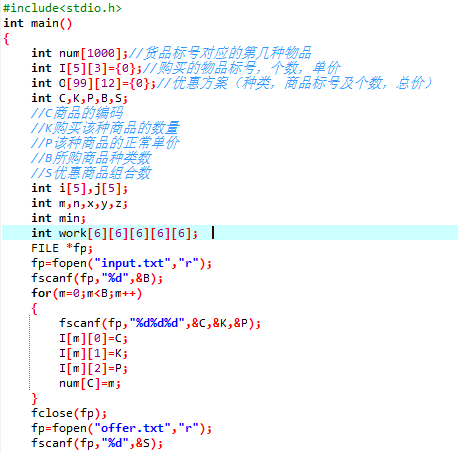
F [a, b, c, d, e] = Min {F [a-S1, b-S2, c-S3, d-S4, e-S5] + SaleCost [S]}

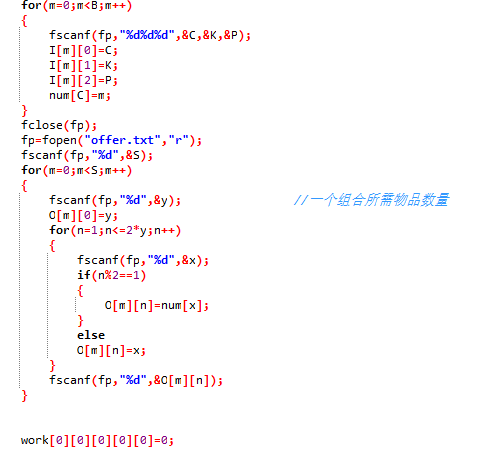
初始条件为：

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。

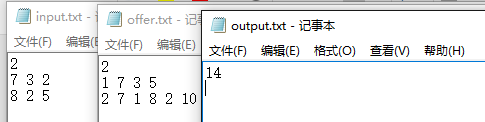
### 代码片段







### 运行截图



## 二、收集样本问题

**★问题描述**： 机器人Rob在一个有n\*n个方格的方形区域F 中收集样本。（i,j）方格中样本的价值为v(i,j)，如图所示。Rob从方形区域F的左上角A点出发，向下或向右行走，直到右下角的B点，在走过的路上，收集方格中的样本。Rob从A点到B 点共走2次，试找出Rob的2条行走路径，使其取得的样本总价值最大。

**★算法设计**： 给定方形区域F中的样本分布，编程计算Rob的2条行走路径，使其取得的样本总价值最大。

**★数据输入**： 由文件input.txt给出输入数据。输入第1行有1个正整数n，表示方形区域F有n\*n个方格。接下来每行有3个整数，前2个表示方格位置，第3个数为该位置样本价值。最后一行是3个0。

**★结果输出**: 将计算的最大样本总价值输出到文件output.txt。

输入文件示例

input.txt

8

2 3 13

2 6 6

3 5 7

4 4 14

5 2 21

5 6 4

6 3 15

7 2 14

0 0 0

输出文件示例

output.txt

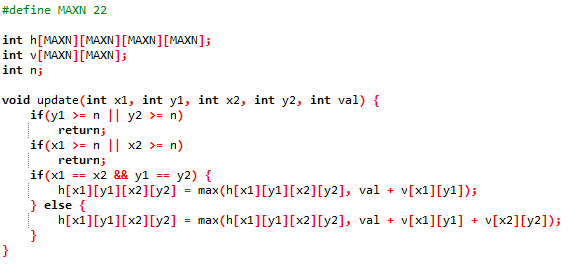
67

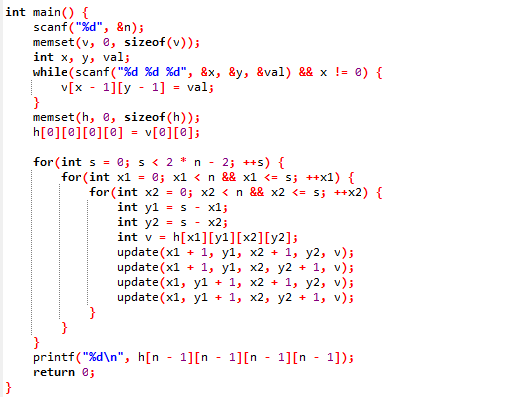
### 问题分析

由于机器人只能往右走或向下走，所以如果每个位置走过后，它左边或上边的点就不需要考虑了。每个机器人到达终点时都经过2\*n-1步。可以设h[x1][y1][x2][y2] 表示第一个机器人到达(x1,y1)第二个机器人走到(x2,y2)时的最优值。如果现在为第S步，如果某个机器的X坐标被确定，那么它的Y坐标也可以推出来（有X+Y = S – 2)。于是我们可以有在第由第S步的最大值去更新S+1步的最大值即可。

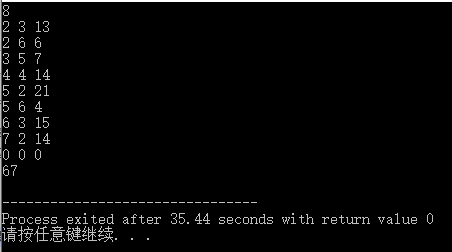
而在S步时，可以根据所在的两个位置选择一个方向进行推导（共四个，每个机器人往下或往右）。更新时需要注意如果两个机器人走到同一个格子时，它的值只更新一次（每个样本只能收集一次）。我们使用的是第S步推导第S+1步，我们知道在第S步时，它左边，上边的方格我们己经不会再使用了，而S+1步所更新的值的位置在它的右边与下边，这样我们就保证了更新的时候没有哪一次把某个方格的值取了两次（如果两个机器人走向同一个方格，需要特殊判断）。这样就保证了算法的正确性。

### 代码片段





### 运行截图



## 三、罗密欧与朱丽叶的迷宫问题

**★问题描述**： 罗密欧与朱丽叶的迷宫。罗密欧与朱丽叶身处一个m×n的迷宫中，如图所示。每一个方格表示迷宫中的一个房间。这m×n个房间中有一些房间是封闭的，不允许任何人进入。在迷宫中任何位置均可沿8 个方向进入未封闭的房间。罗密欧位于迷宫的(p，q)方格中，他必须找出一条通向朱丽叶所在的(r，s)方格的路。在抵达朱丽叶之前，他必须走遍所有未封闭的房间各一次，而且要使到达朱丽叶的转弯次数为最少。每改变一次前进方向算作转弯一次。请设计一个算法帮助罗密欧找出这样一条道路。

**★算法设计**： 对于给定的罗密欧与朱丽叶的迷宫，编程计算罗密欧通向朱丽叶的所有最少转弯道路。

**★数据输入**：由文件input.txt给出输入数据。第一行有3个正整数n，m，k，分别表示迷宫的行数，列数和封闭的房间数。接下来的k行中，每行2 个正整数，表示被封闭的房间所在的行号和列号。最后的2 行，每行也有2 个正整数，分别表示罗密欧所处的方格(p，q)和朱丽叶所处的方格(r，s)。

**★结果输出**: 将计算出的罗密欧通向朱丽叶的最少转弯次数和有多少条不同的最少转弯道路输出到文件output.txt。文件的第一行是最少转弯次数。文件的第2 行是不同的最少转弯道路数。接下来的n行每行m个数，表示迷宫的一条最少转弯道路。A[i][j]=k表示第k步到达方格(i,j)；A[i][j]=-1 表示方格(i,j)是封闭的。

如果罗密欧无法通向朱丽叶则输出“No Solution!”。

输入文件示例

input.txt

3 4 2

1 2

3 4

1 1

2 2

输出文件示例

output.txt

6

7

1 -1 9 8

2 10 6 7

3 4 5 -1

### 问题分析

（1）函数1：bool comp (int x,int y) 判断是（x,y）否越界。

（2）函数2：void save() 保存一条转弯最少的路径

（3）函数3：void search(int dep,int x,int y,int di) 在当前位置（x,y）

按照八个方向搜索，dep用于标记已经走过的房间数,di表示八个方向。

（4）函数4：void main() 主函数初始化迷宫数组，并调用search函数输出一条迷宫路线。

首先,初始化数组,也就是将已经设定好的数组迷宫图装入二维数组,载入方式：读一个字节数字,写入一个数组中,循环写入,直到迷宫图装载完成 。

然后,设定迷宫入口点和出口点.判断迷宫路径搜索的依据 。

最后就是寻路,依次搜索当前点的周围点是否为可走点,

p[N-1][M] p[N+1][M] p[N][M-1] p[N][M+1] ,

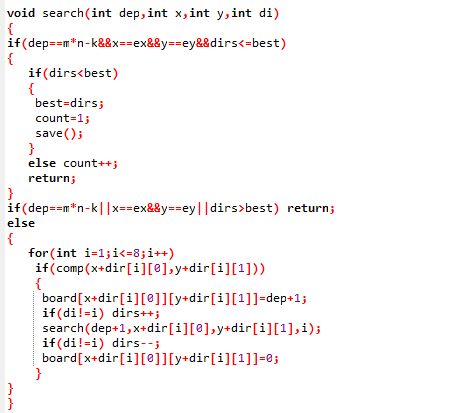
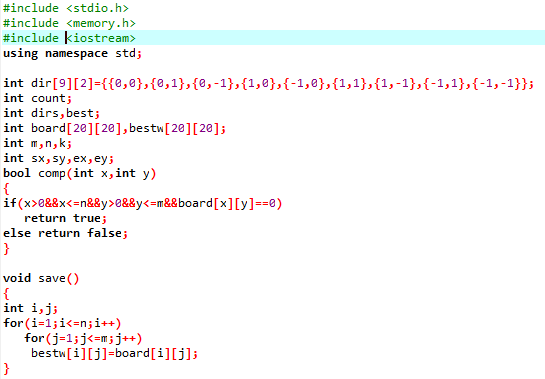
1、判断周围点是否为出口点标记。

2、判断是否为路口点，当有2个以上可用点，应保留这个路口点。

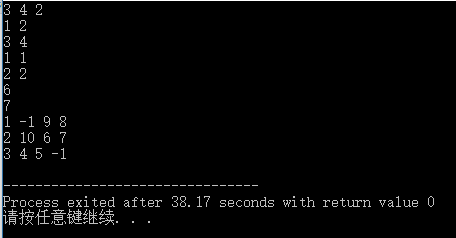
3、路口点的分支记录，保留路口点的同时，还要记录当前走的那条路做为记录。

4、记录已走路点，当走不通的时候，要对比路口点维护表，并比较已走过的分支，如果当前路口点分支都走完了，应删除死路的路口点，并在记录表中找到最后一个路口点，开始走新的分支。

### 代码片段



### 运行截图



## 四、汽车加油问题

**★问题描述**： 一辆汽车加满油后可行驶n公里。旅途中有若干个加油站。设计一个有效算法，指出应在哪些加油站停靠加油，使沿途加油次数最少。对于给定的n(n <= 5000)和k(k <= 1000)个加油站位置，编程计算最少加油次数。并证明算法能产生一个最优解。

**★算法设计**：对于给定的n和k个加油站位置，计算最少加油次数。

**★数据输入**： 由文件input.txt 给出输入数据。第1行有2个正整数n和k，表示汽车加满油后可行驶n公里，且旅途中有k个加油站。接下来的1 行中，有k+1 个整数，表示第k个加油站与第k-1 个加油站之间的距离。第0 个加油站表示出发地，汽车已加满油。第k+1 个加油站表示目的地。

**★结果输出**: 将计算的最少加油次数输出到文件output.txt 。如果无法到达目的地，则输出”No Solution”。

输入文件示例

input.txt

7 7

1 2 3 4 5 1 6

输出文件示例

output.txt

4

### 问题分析

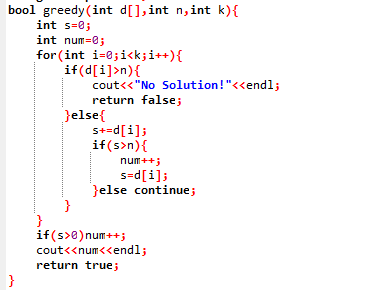
思路：

汽车行驶过程中，应走到自己能走到并且离自己最远的那个加油站，在那个加油站加油后再按照同样的方法贪心。

具体算法：

先检测各加油站之间的距离，若发现其中有一个距离大于汽车加满油能跑的距离，则输出no solution，否则，对加油站间的距离进行逐个扫描，尽量选择往远处走，不能走了就让num++，最终统计出来的num便是最少的加油站数

### 代码片段



### 运行截图



## 五、运动员最佳配对问题

**★问题描述**： 羽毛球队有男女运动员各n 人。给定2 个n×n 矩阵P 和Q。P[i][j] 是男运动员i 和女运动员j 配对组成混合双打的男运动员竞赛优势；Q[i][j] 是女运动员i 和男运动员j 配合的女运动员竞赛优势。由于技术配合和心理状态等各种因素影响，P[i][j] 不一定等于Q[j][i] 。男运动员i 和女运动员j 配对组成混合双打的男女双方竞赛优势为P[i][j]\*Q[j][i] 。设计一个算法，计算男女运动员最佳配对法，使各组男女双方竞赛优势的总和达到最大。

**★算法设计**： 设计一个算法，对于给定的男女运动员竞赛优势，计算男女运动员最佳配对法，使各组男女双方竞赛优势的总和达到最大。

**★数据输入**： 由文件input.txt 给出输入数据。第一行有1 个正整数n (1≤n≤20)。接下来的2n 行，每行n 个数。前n 行是p，后n行是q。

**★结果输出**: 将计算出的男女双方竞赛优势的总和的最大值输出到文件output.txt 。

输入文件示例

input.txt

3

10 2 3

2 3 4

3 4 5

2 2 2

3 5 3

4 5 1

输出文件示例

output.txt

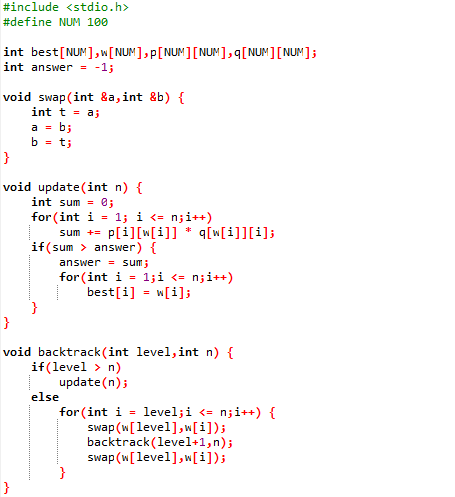
52

### 问题分析

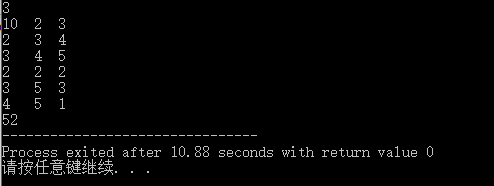
让男队员按自己编号顺序站定，女运动员和他们搭配的各种组合就是女运动员的各种排列。

（如果让女运动员按编号顺序站定，男运动员各种排列和她们搭配也可以）因此，搜索的解空间树是“排列树”。

### 代码片段



### 运行截图



## 六、皇后控制问题

**★问题描述**： 在一个nxn个方格组成的棋盘上的任一方格中放置一个皇后，该皇后可以控制他所在 的行，列以及对角线上的所有方格。对于给定的自然数n，在nxn个方格组成的棋盘上最少要放置多少个皇后才能控制棋盘上的所有方格，且放置的皇后互不攻击？

**★算法设计**： 设计一个拉斯维加斯算法，对于给定的自然数n (1<=n<=100)计算在nxn个方格组成的棋盘上最少要放置多少个皇后才能控制棋盘上的所有方格，且放置的皇后互不攻击。

**★数据输入**： 由文件input.txt给出输入数据。第1行有1个正整数n。

**★结果输出**: 将计算的最少皇后数及最佳放置方案输出到文件output.txt。文件的第一行是最少皇后数；接下来的1行是皇后的最佳放置方案。

输入文件示例

input.txt

8

输出文件示例

output.txt

5  
0 3 6 0 0 2 5 8

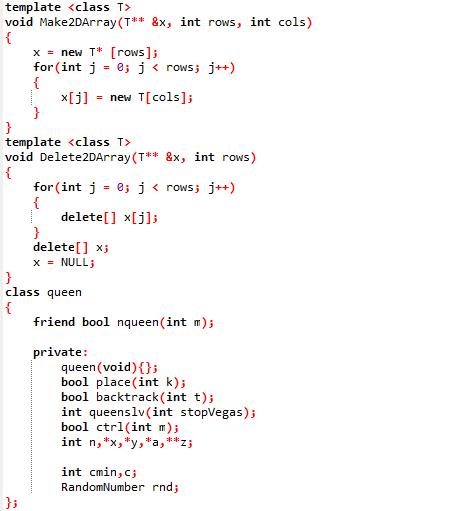
### 问题分析

将拉斯维加斯与回溯法相结合求解n皇后问题，先在棋盘的若干行中随机地放置皇后，然后在后继行中用回溯法继续放置，直到找到一个解或宣告失败。随机放置的皇后越多，后继回溯搜索所需的时间就越少，但失败的概率也就越大。

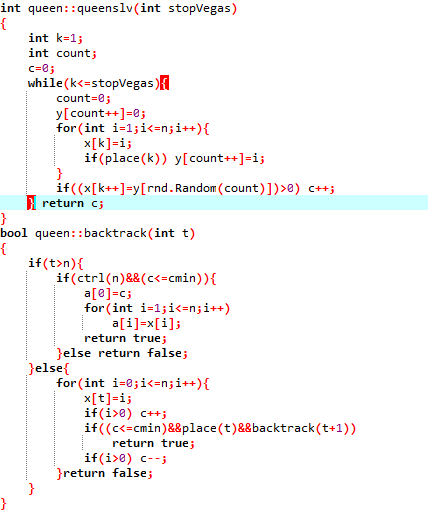
下表给出用拉斯维加斯回溯算法解八皇后问题时，对于不同的stopVegas值，算法成功的概率p，一次成功搜索访问的结点数平均值s，一次不成功搜索访问的节点数平均值e，以及反复调用算法使得最终找到一个解所访问的节点数的平均值t=s+(1-p)e/p。

其中stopVegas=0相应于完全使用回溯法的情形，由表可以看出来，当n=8时，取stopVegas=2时，效率很高。故本算法取stopVegas=2。

### 代码片段









### 运行截图

