POJ 2935【基础】

题目大意：
在一个 6×6 的迷宫中有三堵长度为 1 的墙，墙是在两个网格中间的。给出出发的位置和目的地位置，求一条最短的路径（一定可以走得到）。

输入：
有若干组测试数据。每一组测试数据第一行有两个整数，即起始位置的列数和行数。第二行有两个整数，即终点坐标的列数和行数。下面有三行，每一行有两个整数，分别是一堵墙两端的列数和行数。

输出：
每一组测试数据输出一行，即行走的路径。向上下左右走一步分别输出 NSWE。两组测试数据之间有一个空行。

题解：
由于地图不大，可以直接把地图大小变成 12×12 的，然后墙占一行或者一列，走的时候一步两格就可以了。

POJ 1606【基础】

题目大意：
给出两个水桶 a 和 b 的容量，有六种操作，即对 a（b）装满、清空和倒向 b（a），其中倒向另一个水桶的话，如果当前水桶内的水能全部倒过去就全部倒，否则就倒到另一个桶满了为止。给出最后 b 桶要装有多少水 n，求最快的倒水方式，输出每一步。

输入：
有若干组测试数据，每一组测试数据有一行，包含三个整数 a b n（1<=a b n<=1000）。

输出：
每一组测试数据输出若干行，每一行包括一条指令，有：
fill A
fill B
empty A

empty B
pour A B
pour B A
最后输出一行 success。
两组测试数据的输出不需要换行。

题解：
经典的广搜，每执行一次操作可以得到两个桶的水量，记录在二维数组 vis 中表示已经走到了这个状态，然后加入队列中，并记录前一状态。
和这题几乎完全一样的还有一题 POJ 3414，附上代码，不另写题解。

POJ 2049【中等】

题目大意：
有一个迷宫，在迷宫中有墙和门。
有 m 堵墙，每一道墙表示为(x,y,d,t)。x,y 表示墙的左下角坐标，d 为 0 即向右 t 个单位都是墙，d 为 1 即向上 t 个单位都是墙。
有 n 扇门，每一道门表示为(x,y,d)。x,y 表示门的起始坐标，d 为 0 即向右一个单位表示门，d 为 1 即向上一个单位表示门。
给出起点位置(sx,sy)，并保证这个点的位置不会再墙或者门中，求起点到(0,0)最少要穿过多少扇门。

输入：
有若干组测试数据。每一组测试数据第一行有两个整数 m 和 n。m=n=-1 时测试数据结束。
接下来有 m 行，每一行有四个整数，表示一堵墙，格式如上述。
接下来有 n 行，每一行有三个整数，表示一扇门，格式如上述。
接下来有一行，有两个小数，表示起始位置。
迷宫的 x y 坐标范围都是[1,199]

输出：
每一组测试数据输出一行，包含一个整数：如果能走到终点，即输出最少需要穿过多少扇门，否则输出一行-1。

题解：
也是比较经典的一题，如何处理坐标、网格和边的关系。每一个格子以左下角为基点，则坐标(0,0)对应网格（1，1），坐标(1,2)对应网格（2，3）……定义 UpEdge[x][y]为网格(x,y)上边的值，RightEdge 为网格（x,y）右边的值：

0=没有东西，1=门，INF=墙。搜索的时候，根据原来的网格和移动方向，可以得到跨一格的距离。
从数据规模上估计按 POJ 2935 的方式处理应该也可以，不过我没有试过。
有一点我查了一下 Discuss 才知道为什么 WA：如果人物起始坐标不在迷宫范围内，需要输出 0。这个坑爹啊！

POJ 1482【中等】

题目大意：
有 N 个（1<=N<=20）特性，和 M 个程序包（1<=M<=100），每一个程序包会引入某一些特性或者破坏某一些特性，它们安装的时候也需要某一些特性存在。它们需要依赖和能改变的特性都不超过 N 个。每一个包的安装需要一定的时间。现在需要让 N 个特性都存在，问最少需要多少时间。

输入：
有若干组测试数据。每一组测试数据第一行有两个整数 N 和 M，当 N=M=0 时测试数据结束。接下来有 M 行，每一行给出了一个程序包的信息。首先是一个整数，表示程序包安装需要的时间。然后有两个字符串，由“0”“+”“-”组成，第一个字符串是安装时需要的特性集合，第二个字符串是安装完以后改变的特性集合。如果一个包安装需要的特性集合为 0--+，那么只能在第二、三个特性已经存在，第四个特性不存在，第一个特性存在不存在都可以的情况下安装。如果一个包安装后改变的特性集合是 0-+-，那么安装后第一个特性的状态不改变，第二、四个特性将不存在，第三个特性将存在。

输出：
每一组测试数据输出两行。第一行格式为“Product %测试数据组编号%”，编号从 1 开始。
第二行，如果可以让 N 个特性都存在，输出“Fastest sequence takes %最少使用秒数% seconds.”，否则输出“Bugs cannot be fixed.”。
两组测试数据之间需要有一个空行。

题解：
使用状态压缩，二进制每一位表示一种特性是否存在，一个 int 表示一种状态。广搜的话使用优先队列，优先将用时最小的状态节点弹出来进行扩展。另外判重的时候，如果第二次到某一个状态，但是用时比记录的少，那么就需要再次扩展。
这题用 SPFA 也 ke

POJ 3322【中等偏上】

题目大意：

Bloxorz 是一种在格子地图上滚动一个长方体的游戏。要滚动的长方体由两个正方体组成。地图是一个 row*col（3<=row,col<=500）的矩阵。地图上的格子有三种，一种格子上面无法放置东西，一种格子上面只能承受半个长方体（一个正方体）的重量，一种格子上面能承受整个长方体的重量（即可以在这个格子上竖立整个长方体）。这个长方体在地图上“滚动”来进行移动。给出地图、长方体的起始位置（可能占一个格子或两个格子）和终点格子（只有一个格子），求出需要移动的最小次数。

输入：

有若干组测试数据。每一组测试数据第一行有两个整数 row 和 col，下面有一个 row 行 col 列的字符矩阵。其中，“#”表示不能放东西，“.”表示空的可以承受整个长方体重量的格子，“E”表示空的只能承受一个正方体重量的格子，“X”表示长方体的初始位置，“O”表示长方体的最后位置。

输出：

每一组测试数据输出一行，如果可以滚到目的地就输出最少的步数，否则输出“Impossible”。

题解：

这题很关键一点是要先找到一个方法来描述搜索的状态，要能给每一种状态一个唯一的描述，并且是简洁明了的。vis 数组也要与之进行配合。我一开始没有想到合适的方法，参考了一下网上的题解，下面简单说一下：

首先定义状态，包含四个信息：坐标、已走步长和摆放状态。信息记录右下角坐标，摆放状态 0=竖立，1=横平放，2=竖平放。

由此，可以得到一个状态转移数组 dirs，存储的前两维内容为 dirs[当前状态（012）][转移方向（上下左右）]，第三位是一个三元组，保存的是 x y 坐标的变化量和当前状态按这个方向转移得到的新状态。

接下来就是根据这个来进行广搜了。具体的操作见代码。

POJ 1475【中等偏上】

题目大意：

给出一个经典的推箱子小游戏，其中有一个人，一个箱子，需要把箱子推到目的地。问最少的步数是多少步。

输入：
有若干组测试数据。每一组测试数据第一行有两个整数 r 和 c，表示行数和列数（1<=r,c<=20）。接下来有一个 r 行 c 列的字符矩阵，其中“S”表示人的开始位置，“B”表示箱子的开始位置，“T”表示目的地，“#”表示不能走的墙，“.”表示空地。r=c=0 时测试数据结束。

输出：
每一组测试数据输出两行。第一行格式为“Maze #%测试数据组编号%”，其中测试数据组编号从 1 开始算。第二行一系列移动的字符串。其中上下左右分为别 NSEW，箱子的用大写，人的用小写。如果无法达到，输出一行
“Impossible.”。
两组测试数据的输出之间需要有一个空行。

题解：
很经典的一道双重广搜。第一层（外层）对箱子进行广搜，每次确定一个推的方向，然后根据当前位置确定人需要在哪个位置来进行推。箱子的广搜队列节点保存的状态需要同时包含人的位置和箱子的位置，然后再广搜人能否走到那个位置，如果可以的话将人的行走方案和箱子的移动方向加入到当前节点的行走方案后面。
一开始我写箱子的广搜，简单地不允许箱子进入一个格子两次，后来遇到了这么一组数据：

```
8 9
#########
#......T#
#.S.....#
##B######
#.......#
#.......#
#.......#
#########
```

我的程序就错了。看网上的说法，又想了一下，箱子最多可以踏入一个格子四次，即从不同的方向进入。再多就没有意义了。所以简单地改了一下原来的代码（虽然原来的也可以直接 AC），就可以了。

POJ 1872【中等偏上】

题目大意：给出一个最大不超过 10×10 个网格的棋盘，棋盘上每一格有一些数字，从-1～6。给出一个骰子上面和正面的数字，以及这个骰子的初始位置，问

骰子能否找到最短的路线滚回初始位置。滚动可以向上下左右四个方向，如果满足骰子上面的数字和要滚去的那一个格子的数字一样，或者要滚去的那一个格子的数字是-1。在最短路线长度相同的情况下，选滚动操作字典序最小的，滚动操作的字典序从大到小为上下左右。

输入：
有若干组测试数据。每一组测试数据第一行有一个不超过 20 个字符且不含空格的字符串，作为地图的名字。接下来有 6 个整数，分别是地图的行列数、起始位置的行列和起始状态的上面、正面数字。下面有一个对应大小的数字矩阵，给出了地图上每一格的数字。当地图的名字为 END 时，测试数据结束。

输出：
每一组测试数据先输出一行，即地图的名字。接下来，输出经过的格子的坐标。每一行输出 9 个坐标，格式为“(%行数%,%列数%),……(%行数%,%列数%)，”最后一个坐标不需要后跟“,”。每一行开头空两格。如果没有可行路线，输出一行“ No Solution Possible”。

题解：
首先，知道骰子的上面和正面就一定能知道另外几个面的情况，因为骰子对应两面的数字之和是 7。先手动算出一个转移数组，保存对于每一种上面数字和正面数字的组合情况右面的数字是多少。然后进行广搜，状态判重用四个参数：坐标、上面数字、正面数字。我直接在每一个节点里用一个 vector 保存了经过的坐标。因为只是要求回到原来的位置，所以只要搜索到一个状态的坐标符合要求就可以直接输出了。

POJ 1184【难】

题目大意、输入输出格式见原题中文。

题解：
据说是相当经典的一道状态压缩的题目，我最后还是没有看懂。给出几个我看过的参考资料：
http://wenku.baidu.com/view/0c2742fb770bf78a65295406.html
http://hi.baidu.com/kmzchchycfdeovr/item/6d7358b32be4edf762388eb6
http://blog.163.com/xiangzaihui@126/blog/static/166955749201178625317 54