实验四

# 1. Rescue

**Problem Description**

Angel was caught by the MOLIGPY! He was put in prison by Moligpy. The prison is described as a N \* M (N, M <= 200) matrix. There are WALLs, ROADs, and GUARDs in the prison.  
  
Angel's friends want to save Angel. Their task is: approach Angel. We assume that "approach Angel" is to get to the position where Angel stays. When there's a guard in the grid, we must kill him (or her?) to move into the grid. We assume that we moving up, down, right, left takes us 1 unit time, and killing a guard takes 1 unit time, too. And we are strong enough to kill all the guards.  
  
You have to calculate the minimal time to approach Angel. (We can move only UP, DOWN, LEFT and RIGHT, to the neighbor grid within bound, of course.)

**Input**

First line contains two integers stand for N and M.  
  
Then N lines follows, every line has M characters. "." stands for road, "a" stands for Angel, and "r" stands for each of Angel's friend.  
  
Process to the end of the file.

**Output**

For each test case, your program should output a single integer, standing for the minimal time needed. If such a number does no exist, you should output a line containing "Poor ANGEL has to stay in the prison all his life."

**Sample Input**

7 8

#.#####.

#.a#..r.

#..#x...

..#..#.#

#...##..

.#......

........

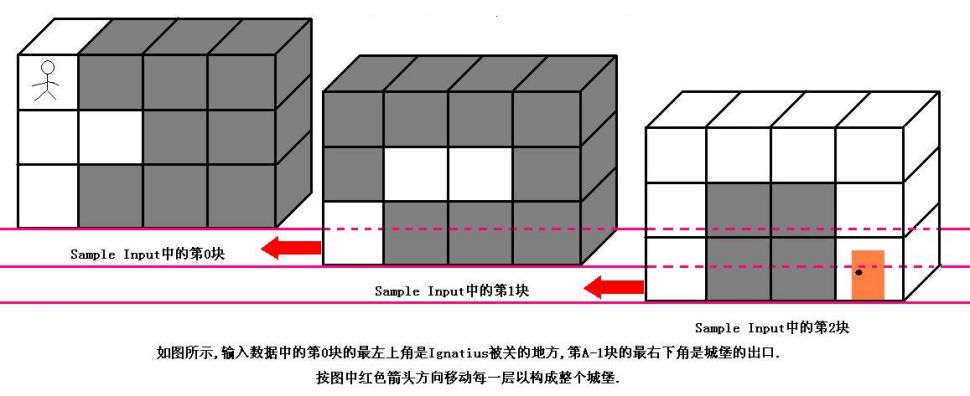
**Sample Output**

13

**2. 胜利大逃亡**

**Problem Description**

Ignatius被魔王抓走了,有一天魔王出差去了,这可是Ignatius逃亡的好机会.  
  
魔王住在一个城堡里,城堡是一个A\*B\*C的立方体,可以被表示成A个B\*C的矩阵,刚开始Ignatius被关在(0,0,0)的位置,离开城堡的门在(A-1,B-1,C-1)的位置,现在知道魔王将在T分钟后回到城堡,Ignatius每分钟能从一个坐标走到相邻的六个坐标中的其中一个.现在给你城堡的地图,请你计算出Ignatius能否在魔王回来前离开城堡(只要走到出口就算离开城堡,如果走到出口的时候魔王刚好回来也算逃亡成功),如果可以请输出需要多少分钟才能离开,如果不能则输出-1.



**Input**

输入数据的第一行是一个正整数K,表明测试数据的数量.每组测试数据的第一行是四个正整数A,B,C和T(1<=A,B,C<=50,1<=T<=1000),它们分别代表城堡的大小和魔王回来的时间.然后是A块输入数据(先是第0块,然后是第1块,第2块......),每块输入数据有B行,每行有C个正整数,代表迷宫的布局,其中0代表路,1代表墙.(如果对输入描述不清楚,可以参考Sample Input中的迷宫描述,它表示的就是上图中的迷宫)  
  
特别注意:本题的测试数据非常大,请使用scanf输入,我不能保证使用cin能不超时.在本OJ上请使用Visual C++提交.

**Output**

对于每组测试数据,如果Ignatius能够在魔王回来前离开城堡,那么请输出他最少需要多少分钟,否则输出-1.

**Sample Input**

1

3 3 4 20

0 1 1 1

0 0 1 1

0 1 1 1

1 1 1 1

1 0 0 1

0 1 1 1

0 0 0 0

0 1 1 0

0 1 1 0

**Sample Output**

11

# 3. Oil Deposits

**Problem Description**

The GeoSurvComp geologic survey company is responsible for detecting underground oil deposits. GeoSurvComp works with one large rectangular region of land at a time, and creates a grid that divides the land into numerous square plots. It then analyzes each plot separately, using sensing equipment to determine whether or not the plot contains oil. A plot containing oil is called a pocket. If two pockets are adjacent, then they are part of the same oil deposit. Oil deposits can be quite large and may contain numerous pockets. Your job is to determine how many different oil deposits are contained in a grid.

**Input**

The input file contains one or more grids. Each grid begins with a line containing m and n, the number of rows and columns in the grid, separated by a single space. If m = 0 it signals the end of the input; otherwise 1 <= m <= 100 and 1 <= n <= 100. Following this are m lines of n characters each (not counting the end-of-line characters). Each character corresponds to one plot, and is either `\*', representing the absence of oil, or `@', representing an oil pocket.

**Output**

For each grid, output the number of distinct oil deposits. Two different pockets are part of the same oil deposit if they are adjacent horizontally, vertically, or diagonally. An oil deposit will not contain more than 100 pockets.

**Sample Input**

1 1

\*

3 5

\*@\*@\*

\*\*@\*\*

\*@\*@\*

1 8

@@\*\*\*\*@\*

5 5

\*\*\*\*@

\*@@\*@

\*@\*\*@

@@@\*@

@@\*\*@

0 0

**Sample Output**

0

1

2

2

**4. I NEED A OFFER!**

**Problem Description**

Speakless很早就想出国，现在他已经考完了所有需要的考试，准备了所有要准备的材料，于是，便需要去申请学校了。要申请国外的任何大学，你都要交纳一定的申请费用，这可是很惊人的。Speakless没有多少钱，总共只攒了n万美元。他将在m个学校中选择若干的（当然要在他的经济承受范围内）。每个学校都有不同的申请费用a（万美元），并且Speakless估计了他得到这个学校offer的可能性b。不同学校之间是否得到offer不会互相影响。“I NEED A OFFER”，他大叫一声。帮帮这个可怜的人吧，帮助他计算一下，他可以收到至少一份offer的最大概率。（如果Speakless选择了多个学校，得到任意一个学校的offer都可以）。

**Input**

输入有若干组数据，每组数据的第一行有两个正整数n,m(0<=n<=10000,0<=m<=10000)  
后面的m行，每行都有两个数据ai(整型),bi(实型)分别表示第i个学校的申请费用和可能拿到offer的概率。  
输入的最后有两个0。

**Output**

每组数据都对应一个输出，表示Speakless可能得到至少一份offer的最大概率。用百分数表示，精确到小数点后一位。

**Sample Input**

10 3

4 0.1

4 0.2

5 0.3

0 0

**Sample Output**

44.0%

**Hint**

You should use printf("%%") to print a '%'.

# 5. 非常可乐

**Problem Description**

大家一定觉的运动以后喝可乐是一件很惬意的事情，但是seeyou却不这么认为。因为每次当seeyou买了可乐以后，阿牛就要求和seeyou一起分享这一瓶可乐，而且一定要喝的和seeyou一样多。但seeyou的手中只有两个杯子，它们的容量分别是N 毫升和M 毫升 可乐的体积为S （S<101）毫升　(正好装满一瓶) ，它们三个之间可以相互倒可乐 (都是没有刻度的，且 S==N+M，101＞S＞0，N＞0，M＞0) 。聪明的ACMER你们说他们能平分吗？如果能请输出倒可乐的最少的次数，如果不能输出"NO"。

**Input**

三个整数 : S 可乐的体积 , N 和 M是两个杯子的容量，以"0 0 0"结束。

**Output**

如果能平分的话请输出最少要倒的次数，否则输出"NO"。

**Sample Input**

7 4 3

4 1 3

0 0 0

**Sample Output**

NO

3

**6. Pots**

**Description**

You are given two pots, having the volume of A and B liters respectively. The following operations can be performed:

FILL(i) fill the pot i (1 ≤ i ≤ 2) from the tap;

DROP(i) empty the pot i to the drain;

POUR(i,j) pour from pot i to pot j; after this operation either the pot j is full (and there may be some water left in the pot i), or the pot i is empty (and all its contents have been moved to the pot j).

Write a program to find the shortest possible sequence of these operations that will yield exactly C liters of water in one of the pots.

**Input**

On the first and only line are the numbers A, B, and C. These are all integers in the range from 1 to 100 and C≤max(A,B).

**Output**

The first line of the output must contain the length of the sequence of operations K. The following K lines must each describe one operation. If there are several sequences of minimal length, output any one of them. If the desired result can’t be achieved, the first and only line of the file must contain the word ‘impossible’.

**Sample Input**

3 5 4

**Sample Output**

6

FILL(2)

POUR(2,1)

DROP(1)

POUR(2,1)

FILL(2)

POUR(2,1)

**7. A计划**

**Problem Description**

可怜的公主在一次次被魔王掳走一次次被骑士们救回来之后，而今，不幸的她再一次面临生命的考验。魔王已经发出消息说将在T时刻吃掉公主，因为他听信谣言说吃公主的肉也能长生不老。年迈的国王正是心急如焚，告招天下勇士来拯救公主。不过公主早已习以为常，她深信智勇的骑士LJ肯定能将她救出。  
现据密探所报，公主被关在一个两层的迷宫里，迷宫的入口是S（0，0，0），公主的位置用P表示，时空传输机用#表示，墙用\*表示，平地用.表示。骑士们一进入时空传输机就会被转到另一层的相对位置，但如果被转到的位置是墙的话，那骑士们就会被撞死。骑士们在一层中只能前后左右移动，每移动一格花1时刻。层间的移动只能通过时空传输机，且不需要任何时间。

**Input**

输入的第一行C表示共有C个测试数据，每个测试数据的前一行有三个整数N，M，T。 N，M迷宫的大小N\*M（1 <= N,M <=10)。T如上所意。接下去的前N\*M表示迷宫的第一层的布置情况，后N\*M表示迷宫第二层的布置情况。

**Output**

如果骑士们能够在T时刻能找到公主就输出“YES”，否则输出“NO”。

**Sample Input**

1

5 5 14

S\*#\*.

.#...

.....

\*\*\*\*.

...#.

..\*.P

#.\*..

\*\*\*..

...\*.

\*.#..

**Sample Output**

YES

**8. Fire**

**Problem Description**

Joe works in a maze. Unfortunately, portions of the maze have caught on fire, and the owner of the maze neglected to create a fire escape plan. Help Joe escape the maze. Given Joe’s location in the maze and which squares of the maze are on fire, you must determine whether Joe can exit the maze before the fire reaches him, and how fast he can do it. Joe and the fire each move one square per minute, vertically or horizontally (not diagonally). The fire spreads all four directions from each square that is on fire. Joe may exit the maze from any square that borders the edge of the maze. Neither Joe nor the fire may enter a square that is occupied by a wall

**Input**

The first line of input contains a single integer, the number of test cases to follow. The first line of each test case contains the two integers R and C, separated by spaces, with 1 ≤ R, C ≤ 1000. The following R lines of the test case each contain one row of the maze. Each of these lines contains exactly C characters, and each of these characters is one of:

• #, a wall

• ., a passable square

• J, Joe’s initial position in the maze, which is a passable square

• F, a square that is on fire

There will be exactly one J in each test case.

**Output**

For each test case, output a single line containing ‘IMPOSSIBLE’ if Joe cannot exit the maze before the fire reaches him, or an integer giving the earliest time Joe can safely exit the maze, in minutes.

**Sample Input**

2

4 4

####

#JF#

#..#

#..#

3 3

###

#J.

#.F

**Sample Output**

3

IMPOSSIBLE

# 9. 哈密顿绕行世界问题

**Problem Description**

一个规则的实心十二面体，它的 20个顶点标出世界著名的20个城市，你从一个城市出发经过每个城市刚好一次后回到出发的城市。

**Input**

前20行的第i行有3个数,表示与第i个城市相邻的3个城市.第20行以后每行有1个数m,m<=20,m>=1.m=0退出.

**Output**

输出从第m个城市出发经过每个城市1次又回到m的所有路线,如有多条路线,按字典序输出,每行1条路线.每行首先输出是第几条路线.然后个一个: 后列出经过的城市.参看Sample output

**Sample Input**

2 5 20

1 3 12

2 4 10

3 5 8

1 4 6

5 7 19

6 8 17

4 7 9

8 10 16

3 9 11

10 12 15

2 11 13

12 14 20

13 15 18

11 14 16

9 15 17

7 16 18

14 17 19

6 18 20

1 13 19

5

0

**Sample Output**

1: 5 1 2 3 4 8 7 17 18 14 15 16 9 10 11 12 13 20 19 6 5

2: 5 1 2 3 4 8 9 10 11 12 13 20 19 18 14 15 16 17 7 6 5

3: 5 1 2 3 10 9 16 17 18 14 15 11 12 13 20 19 6 7 8 4 5

4: 5 1 2 3 10 11 12 13 20 19 6 7 17 18 14 15 16 9 8 4 5

5: 5 1 2 12 11 10 3 4 8 9 16 15 14 13 20 19 18 17 7 6 5

6: 5 1 2 12 11 15 14 13 20 19 18 17 16 9 10 3 4 8 7 6 5

7: 5 1 2 12 11 15 16 9 10 3 4 8 7 17 18 14 13 20 19 6 5

8: 5 1 2 12 11 15 16 17 18 14 13 20 19 6 7 8 9 10 3 4 5

9: 5 1 2 12 13 20 19 6 7 8 9 16 17 18 14 15 11 10 3 4 5

10: 5 1 2 12 13 20 19 18 14 15 11 10 3 4 8 9 16 17 7 6 5

11: 5 1 20 13 12 2 3 4 8 7 17 16 9 10 11 15 14 18 19 6 5

12: 5 1 20 13 12 2 3 10 11 15 14 18 19 6 7 17 16 9 8 4 5

13: 5 1 20 13 14 15 11 12 2 3 10 9 16 17 18 19 6 7 8 4 5

14: 5 1 20 13 14 15 16 9 10 11 12 2 3 4 8 7 17 18 19 6 5

15: 5 1 20 13 14 15 16 17 18 19 6 7 8 9 10 11 12 2 3 4 5

16: 5 1 20 13 14 18 19 6 7 17 16 15 11 12 2 3 10 9 8 4 5

17: 5 1 20 19 6 7 8 9 10 11 15 16 17 18 14 13 12 2 3 4 5

18: 5 1 20 19 6 7 17 18 14 13 12 2 3 10 11 15 16 9 8 4 5

19: 5 1 20 19 18 14 13 12 2 3 4 8 9 10 11 15 16 17 7 6 5

20: 5 1 20 19 18 17 16 9 10 11 15 14 13 12 2 3 4 8 7 6 5

21: 5 4 3 2 1 20 13 12 11 10 9 8 7 17 16 15 14 18 19 6 5

22: 5 4 3 2 1 20 19 18 17 16 15 14 13 12 11 10 9 8 7 6 5

23: 5 4 3 2 12 11 10 9 8 7 6 19 18 17 16 15 14 13 20 1 5

24: 5 4 3 2 12 13 14 18 17 16 15 11 10 9 8 7 6 19 20 1 5

25: 5 4 3 10 9 8 7 6 19 20 13 14 18 17 16 15 11 12 2 1 5

26: 5 4 3 10 9 8 7 17 16 15 11 12 2 1 20 13 14 18 19 6 5

27: 5 4 3 10 11 12 2 1 20 13 14 15 16 9 8 7 17 18 19 6 5

28: 5 4 3 10 11 15 14 13 12 2 1 20 19 18 17 16 9 8 7 6 5

29: 5 4 3 10 11 15 14 18 17 16 9 8 7 6 19 20 13 12 2 1 5

30: 5 4 3 10 11 15 16 9 8 7 17 18 14 13 12 2 1 20 19 6 5

31: 5 4 8 7 6 19 18 17 16 9 10 3 2 12 11 15 14 13 20 1 5

32: 5 4 8 7 6 19 20 13 12 11 15 14 18 17 16 9 10 3 2 1 5

33: 5 4 8 7 17 16 9 10 3 2 1 20 13 12 11 15 14 18 19 6 5

34: 5 4 8 7 17 18 14 13 12 11 15 16 9 10 3 2 1 20 19 6 5

35: 5 4 8 9 10 3 2 1 20 19 18 14 13 12 11 15 16 17 7 6 5

36: 5 4 8 9 10 3 2 12 11 15 16 17 7 6 19 18 14 13 20 1 5

37: 5 4 8 9 16 15 11 10 3 2 12 13 14 18 17 7 6 19 20 1 5

38: 5 4 8 9 16 15 14 13 12 11 10 3 2 1 20 19 18 17 7 6 5

39: 5 4 8 9 16 15 14 18 17 7 6 19 20 13 12 11 10 3 2 1 5

40: 5 4 8 9 16 17 7 6 19 18 14 15 11 10 3 2 12 13 20 1 5

41: 5 6 7 8 4 3 2 12 13 14 15 11 10 9 16 17 18 19 20 1 5

42: 5 6 7 8 4 3 10 9 16 17 18 19 20 13 14 15 11 12 2 1 5

43: 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 1 2 3 4 5

44: 5 6 7 8 9 16 17 18 19 20 1 2 12 13 14 15 11 10 3 4 5

45: 5 6 7 17 16 9 8 4 3 10 11 15 14 18 19 20 13 12 2 1 5

46: 5 6 7 17 16 15 11 10 9 8 4 3 2 12 13 14 18 19 20 1 5

47: 5 6 7 17 16 15 11 12 13 14 18 19 20 1 2 3 10 9 8 4 5

48: 5 6 7 17 16 15 14 18 19 20 13 12 11 10 9 8 4 3 2 1 5

49: 5 6 7 17 18 19 20 1 2 3 10 11 12 13 14 15 16 9 8 4 5

50: 5 6 7 17 18 19 20 13 14 15 16 9 8 4 3 10 11 12 2 1 5

51: 5 6 19 18 14 13 20 1 2 12 11 15 16 17 7 8 9 10 3 4 5

52: 5 6 19 18 14 15 11 10 9 16 17 7 8 4 3 2 12 13 20 1 5

53: 5 6 19 18 14 15 11 12 13 20 1 2 3 10 9 16 17 7 8 4 5

54: 5 6 19 18 14 15 16 17 7 8 9 10 11 12 13 20 1 2 3 4 5

55: 5 6 19 18 17 7 8 4 3 2 12 11 10 9 16 15 14 13 20 1 5

56: 5 6 19 18 17 7 8 9 16 15 14 13 20 1 2 12 11 10 3 4 5

57: 5 6 19 20 1 2 3 10 9 16 15 11 12 13 14 18 17 7 8 4 5

58: 5 6 19 20 1 2 12 13 14 18 17 7 8 9 16 15 11 10 3 4 5

59: 5 6 19 20 13 12 11 10 9 16 15 14 18 17 7 8 4 3 2 1 5

60: 5 6 19 20 13 14 18 17 7 8 4 3 10 9 16 15 11 12 2 1 5

**10.Eight**

**Description**

The 15-puzzle has been around for over 100 years; even if you don't know it by that name, you've seen it. It is constructed with 15 sliding tiles, each with a number from 1 to 15 on it, and all packed into a 4 by 4 frame with one tile missing. Let's call the missing tile 'x'; the object of the puzzle is to arrange the tiles so that they are ordered as:

1 2 3 4

5 6 7 8

9 10 11 12

13 14 15 x

where the only legal operation is to exchange 'x' with one of the tiles with which it shares an edge. As an example, the following sequence of moves solves a slightly scrambled puzzle:

1 2 3 4 1 2 3 4 1 2 3 4 1 2 3 4

5 6 7 8 5 6 7 8 5 6 7 8 5 6 7 8

9 x 10 12 9 10 x 12 9 10 11 12 9 10 11 12

13 14 11 15 13 14 11 15 13 14 x 15 13 14 15 x

r-> d-> r->

The letters in the previous row indicate which neighbor of the 'x' tile is swapped with the 'x' tile at each step; legal values are 'r','l','u' and 'd', for right, left, up, and down, respectively.

Not all puzzles can be solved; in 1870, a man named Sam Loyd was famous for distributing an unsolvable version of the puzzle, and

frustrating many people. In fact, all you have to do to make a regular puzzle into an unsolvable one is to swap two tiles (not counting the missing 'x' tile, of course).

In this problem, you will write a program for solving the less well-known 8-puzzle, composed of tiles on a three by three

arrangement.

**Input**

You will receive a description of a configuration of the 8 puzzle. The description is just a list of the tiles in their initial positions, with the rows listed from top to bottom, and the tiles listed from left to right within a row, where the tiles are represented by numbers 1 to 8, plus 'x'. For example, this puzzle

1 2 3

x 4 6

7 5 8

is described by this list:

1 2 3 x 4 6 7 5 8

**Output**

You will print to standard output either the word ``unsolvable'', if the puzzle has no solution, or a string consisting entirely of the letters 'r', 'l', 'u' and 'd' that describes a series of moves that produce a solution. The string should include no spaces and start at the beginning of the line.

**Sample Input**

2 3 4 1 5 x 7 6 8

**Sample Output**

ullddrurdllurdruldr