**[**报告**]** **F - Find the Winning Move**

**[Source]**

<http://acm.zju.edu.cn/onlinejudge/showProblem.do?problemCode=1703>

**[Description]**

在4×4棋盘上的tic-tac-toe。定义forced win：在x的回合，存在一步x的下法，在这之后无论o怎么走都有x必胜。现给出一种局面，此时是x的回合，按照从上到下、从左到右的顺序找出第一个能够形成forced win的x的下法。

**[Solution]**

我的思路就是先在题目给出的局面上枚举x下一步的位置。再以此为基础，dfs回溯枚举接下来o和x可能的下法。要想有x必胜，须(1)在o的回合，对于所有o的下法都有x必胜；(2)在x的回合，存在x的下法使x必胜。

用一个字符数组board保存当前局面。定义int r\_x[4], r\_o[4], c\_x[4], c\_o[4], d\_x[2], d\_o[2];记录各行、列、对角线上x、o出现的次数，用来辅助判断游戏胜负情况。具体程序见代码和注释吧（略渣见谅）。

另外在网上找到两组感觉挺管用的数据：

**Input**

?

.ooo

.x..

xx..

x...

?

.o..

.xx.

xo.o

.ox.

$

**Output**

(0,0)

(0,0)

**[Code]**

#include<cstdio>

#include<cstring>

#include<algorithm>

#include<queue>

**using namespace** std**;**

**char** T**[**5**],** board**[**4**][**10**];**

**int** r\_x**[**4**],** r\_o**[**4**],** c\_x**[**4**],** c\_o**[**4**],** d\_x**[**2**],** d\_o**[**2**];**

**void** add\_x**(int** i**, int** j**)** //在(i,j)下x

**{**

board**[**i**][**j**] =** 'x'**;**

**if (**i **==** j**)**

d\_x**[**0**]++;**

**if (**i**+**j **==** 3**)**

d\_x**[**1**]++;**

r\_x**[**i**]++;**

c\_x**[**j**]++;**

**}**

**void** del\_x**(int** i**, int** j**)** //撤销在(i,j)下x

**{**

board**[**i**][**j**] =** '.'**;**

**if (**i **==** j**)**

d\_x**[**0**]--;**

**if (**i**+**j **==** 3**)**

d\_x**[**1**]--;**

r\_x**[**i**]--;**

c\_x**[**j**]--;**

**}**

**void** add\_o**(int** i**, int** j**)** //在(i,j)下o

**{**

board**[**i**][**j**] =** 'o'**;**

**if (**i **==** j**)**

d\_o**[**0**]++;**

**if (**i**+**j **==** 3**)**

d\_o**[**1**]++;**

r\_o**[**i**]++;**

c\_o**[**j**]++;**

**}**

**void** del\_o**(int** i**, int** j**)** //撤销在(i,j)下o

**{**

board**[**i**][**j**] =** '.'**;**

**if (**i **==** j**)**

d\_o**[**0**]--;**

**if (**i**+**j **==** 3**)**

d\_o**[**1**]--;**

r\_o**[**i**]--;**

c\_o**[**j**]--;**

**}**

**bool** Search**(int** x**)** //Search(1)表示当前局面x走下一步，x是否必胜；Search(0)表示当前局面o走下一步，x是否必胜

**{**

**if (**x**)**

**{**

**int** cnt **=** 0**;** //此时为x的回合，若存在某行x冲3，则x必胜

**for (int** k **=** 0**;** k **<** 4**;** k**++)**

**if (**r\_x**[**k**] ==** 3 **&&** r\_o**[**k**] ==** 0**)**

cnt**++;**

**for (int** k **=** 0**;** k **<** 4**;** k**++)**

**if (**c\_x**[**k**] ==** 3 **&&** c\_o**[**k**] ==** 0**)**

cnt**++;**

**for (int** k **=** 0**;** k **<** 2**;** k**++)**

**if (**d\_x**[**k**] ==** 3 **&&** d\_o**[**k**] ==** 0**)**

cnt**++;**

**if (**cnt**) return** 1**;**

**}**

**bool** flag\_o **=** 1**;** //在o的回合，存在一步o使x不能必胜时就为false，对于所有o的下法都有x必胜时，才true

**bool** flag\_x **=** 0**;** //在x的回合，只要存在x的下法使x必胜，就true

**int** cnt\_ **=** 0**;** //枚举过的合法下法的次数

**for (int** i **=** 0**;** i **<** 4**;** i**++)**

**for (int** j **=** 0**;** j **<** 4**;** j**++)**

**if (**board**[**i**][**j**] ==** '.' **&&** flag\_o **&& !**flag\_x**)**

**{**

cnt\_**++;**

**if (**x**)**

**{**

add\_x**(**i**,** j**);**

**if (**Search**(**0**))** flag\_x **=** 1**;**

del\_x**(**i**,** j**);**

**}**

**else**

**{**

add\_o**(**i**,** j**);**

**if (!**Search**(**1**))** flag\_o **=** 0**;**

del\_o**(**i**,** j**);**

**}**

**}**

**if (!**cnt\_**) return** 0**;** //cnt\_为零则棋盘满了游戏结束，此时必为o的回合

**return** x **?** flag\_x **:** flag\_o**;**

**}**

**int main()**

**{**

**while (**fgets**(**T**,** 5**,** stdin**))**

**{**

**if (**T**[**0**] ==** '$'**) break;**

**for (int** i **=** 0**;** i **<** 4**;** i**++)**

fgets**(**board**[**i**],** 10**,** stdin**);**

memset**(**r\_x**,** 0**, sizeof(**r\_x**));**

memset**(**r\_o**,** 0**, sizeof(**r\_o**));**

memset**(**c\_x**,** 0**, sizeof(**c\_x**));**

memset**(**c\_o**,** 0**, sizeof(**c\_o**));**

memset**(**d\_x**,** 0**, sizeof(**d\_x**));**

memset**(**d\_o**,** 0**, sizeof(**d\_o**));**

**for (int** i **=** 0**;** i **<** 4**;** i**++)**

**for (int** j **=** 0**;** j **<** 4**;** j**++)**

**{**

**if (**board**[**i**][**j**] ==** 'x'**)**

add\_x**(**i**,** j**);**

**if (**board**[**i**][**j**] ==** 'o'**)**

add\_o**(**i**,** j**);**

**}**

**bool** flag **=** 0**;** //为true则存在forced win下法

**for (int** i **=** 0**;** i **<** 4**;** i**++)**

**for (int** j **=** 0**;** j **<** 4**;** j**++)**

**if (**board**[**i**][**j**] ==** '.' **&& !**flag**)**

**{**

add\_x**(**i**,** j**);**

**int** cnt **=** 0**;** //记录4个x连成一条线的个数

**for (int** k **=** 0**;** k **<** 4**;** k**++)**

**if (**r\_x**[**k**] ==** 4**)** cnt**++;**

**for (int** k **=** 0**;** k **<** 4**;** k**++)**

**if (**c\_x**[**k**] ==** 4**)** cnt**++;**

**for (int** k **=** 0**;** k **<** 2**;** k**++)**

**if (**d\_x**[**k**] ==** 4**)** cnt**++;**

**if (**cnt **||** Search**(**0**))** //cnt>0说明下完这一步后x就已经赢了，不需要再dfs

flag **=** 1**,** printf**(**"(%d,%d)\n"**,** i**,** j**);**

del\_x**(**i**,** j**);**

**}**

**if (!**flag**)** printf**(**"#####\n"**);**

**}**

**return** 0**;**

**}**