给定一个m × n (m行, n列)的迷宫，迷宫中有两个位置，gloria想从迷宫的一个位置走到另外一个位置，当然迷宫中有些地方是空地，gloria可以穿越，有些地方是障碍，她必须绕行，从迷宫的一个位置，只能走到与它相邻的4个位置中,当然在行走过程中，gloria不能走到迷宫外面去。令人头痛的是，gloria是个没什么方向感的人，因此，她在行走过程中，不能转太多弯了，否则她会晕倒的。我们假定给定的两个位置都是空地，初始时，gloria所面向的方向未定，她可以选择4个方向的任何一个出发，而不算成一次转弯。gloria能从一个位置走到另外一个位置吗？

**Input**

　　第1行为一个整数t (1 ≤ t ≤ 100),表示测试数据的个数，接下来为t组测试数据，每组测试数据中，   
　　第1行为两个整数m, n (1 ≤ m, n ≤ 100),分别表示迷宫的行数和列数，接下来m行，每行包括n个字符，其中字符'.'表示该位置为空地，字符'\*'表示该位置为障碍，输入数据中只有这两种字符，每组测试数据的最后一行为5个整数k, x 1, y 1, x 2, y 2 (1 ≤ k ≤ 10, 1 ≤ x 1, x 2 ≤ n, 1 ≤ y 1, y 2 ≤ m),其中k表示gloria最多能转的弯数，(x 1, y 1), (x 2, y 2)表示两个位置，其中x 1，x 2对应列，y 1, y 2对应行。

**Output**

　　每组测试数据对应为一行，若gloria能从一个位置走到另外一个位置，输出“yes”，否则输出“no”。

**Sample Input**

2

5 5

...\*\*

\*.\*\*.

.....

.....

\*....

1 1 1 1 3

5 5

...\*\*

\*.\*\*.

.....

.....

\*....

2 1 1 1 3

**Sample Output**

no

yes

#include <iostream>

#include<stdio.h>

using namespace std;

#define inf 0x3fffffff

#define M 105

//1、wan用于转弯数剪枝；2、step用于步数剪枝，就不用vis标记访问了

int r, c, ex, ey, k, wan[M][M];

char map[M][M];

int dx[4] = {-1, 0, 1, 0};

int dy[4] = {0, 1, 0, -1};

bool flag;

void dfs (int x, int y, int dir) //dir为当前方向

{

if(flag)return;

if (x==ex&&y==ey)

{

if (wan[x][y]<=k)

flag=true;

return;

}

if (wan[x][y] > k)

return ;

if (wan[x][y] == k && x !=ex && y != ey)

return ;

for (int i = 0; i < 4; i++)//枚举四个方向

{

int tx = x + dx[i];

int ty = y + dy[i];

if (tx<0||tx>=r||ty<0||ty>=c)

continue;

//转弯数相等不可剪掉，所以是wan[tx][ty] < wan[x][y]而不是wan[tx][ty] <= wan[x][y]

if (map[tx][ty] == '\*' || wan[tx][ty] < wan[x][y]) //转弯数剪枝

continue;//如果已经比它小了，则它就一定不是最优的了

if (dir != -1 && i != dir && wan[tx][ty] < wan[x][y] + 1) //转弯数剪枝

continue;//如果已经比它小了，则它就一定不是最优的了

if (dir != -1 && i != dir)

wan[tx][ty]=wan[x][y]+1;else wan[tx][ty]=wan[x][y]; //如果方向变了转弯+1

dfs (tx,ty,i);

}

}

int main()

{

int t, i, j, sx, sy; //sx， sy是起点

cin>>t;

while (t--)

{

cin>>r>>c;

for (i = 0; i < r; i++)

gets(map[i]);

cin>>k>>sy>>sx>>ey>>ex;

sx--, sy--, ex--, ey--; //我从0开始编号，而题目是从1开始

for (i = 0; i < r; i++)

for (j = 0; j < c; j++)

wan[i][j] = inf; //初始化转弯数和步数为无穷大

wan[sx][sy] = 0; //到达起点的转弯数

flag = false;

dfs (sx, sy, -1); //一开始可以走任意方向，所以设方向为-1

if (flag)

puts ("yes");

else puts ("no");

}

return 0;

}