29th ACM International Collegiate
Programming Contest, 2004-2005
Asia Region, Tehran Site
Sharif University of Technology
1–3 Dec. 2004

Sponsored by
IBM®

Problem H (Program filename: H.cpp, H.dpr, or H.java)

Prince of Persia

A new job for the well known Prince of Persia! Not surprisingly, Jafar, the treacherous vizier of the king has put the daughter of the king in a cellar while he is away to visit a neighboring country. The prince fights bravely with the guards on his way down to the cellar. Just one step to the cellar, the prince enters a dark room containing the information to unlock the door of the cellar. The information is carved on several stone plates installed in the walls of the room. Luckily, there is a ray of light emanating through a tiny hole in one of the walls. There are a number of mirrors in the room that the prince may use to direct the ray of light to illuminate a stone plate. Each mirror may be placed in certain locations in the room without any change in its directions (i.e., angles with the walls). The question is, if the prince can illuminate every plate on the wall using the mirrors.

To simplify the problem, we consider the room when viewed from above as a grid with n rows and m columns. The bounding cells are walls of the room and the mirrors can be put in the interior cells. No two mirrors can be placed in one cell. We have a number of mirrors available, each allowed to be put in certain positions inside the room. The direction of each mirror is known in advance and cannot be changed. To illuminate one plate, the prince must place a subset of mirrors in their allowed positions such that the ray of light reaches the plate after reflections made by the mirrors. The mirrors are opaque, meaning that the back-side of the mirrors absorbs the light. The problem is to find whether he can illuminate all and each plate on the wall. Obviously, the plates must be illuminated in turn since there is no way to fork the ray of light into multiple rays. When illuminating a plate, he may use a different subset of mirrors in possibly different positions.

Input (filename: H.in)

The first line of the input contains a single integer t ($1 \leq t \leq 20$) which is the number of test cases in the input. Each test case starts with zero or more lines each describing one mirror. A mirror is described by a single digit which is its identifier, followed by exactly one blank character, and one of the four character pairs `-/`, $-\backslash$, `/-`, and $\backslash-$ indicating the mirror's direction. The hyphen character (`-`) determines the back-side of the mirror. For example, `-/` indicates a mirror which reflects the ray from the right to bottom and vice versa, but blocks the rays from the left or top.

After the mirror description lines, there are n lines each of length m characters which describe the grid representing the room when viewed from the above (($2 \leq m, n \leq 40$)). The walls of the room (which are always in the boundary of the grid) are denoted by the hash signs (`#`). Those cells of the wall which contain plates are represented by `-` and `|` for horizontal and vertical walls respectively. The only cell in the boundary containing a blank character represents the hole from which the light is emanated into the room. The light ray enters the room perpendicular to the wall holding the hole. You may assume that the four cells in the corner always contain hash signs. The interior cells of the grid are all filled by the dot character (`.`) except those that a mirror can be placed into which are specified by digits. A mirror can be placed in a cell only if the digit of the cell is the same as the mirror identifier.

Output (filename: H.out)

For each test case, there is one line in the output containing the word `YES` or `NO` indicating if all plates in the input can be illuminated according to the problem conditions or not.

Sample Input

```
2
0 -/
1 /-
3 -/
5 \-
######-#####-#########
#....................#
|....................#
#.0..............3...#
#..........0.........#
#....................#
#....................#
#.....1....1.........#
#..........1..5......#
|....................#
|....................#
#....................#
#........1......1....#
#....................#
#....0......0........#
#..............3.....#
#...3................#
#...........3........#
## #####--############
0 -/
1 /-
3 -\
## ####-##
#........#
#.3..0.1.#
|.1......#
##########
```

Sample Output

```
NO
YES
```