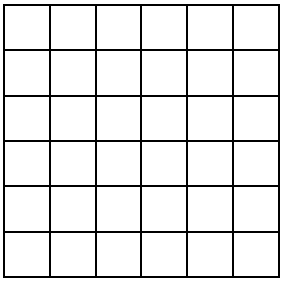
4. Blocks Box

Program Name: Blocks.java Input File: blocks.dat



Rocco has a box for his blocks. The inside measurements of the box are 18" x 18". All of his blocks are 3" wide and have lengths of 3", 6", 9", or 12". If Rocco wanted to completely cover the bottom of the box, he could place thirty-six 3" blocks on the bottom or he could place twelve 9" blocks in the bottom. A diagram of thirty-six 3" spots for blocks is to the right. For the purpose of this program, each 3" spot will be indicated by an asterisk or a period as defined below.

Rocco's mother wants Rocco to learn to problem solve so she places some blocks in place for him. He is to place the other blocks into the box using the following rules:

- Blocks must be placed either vertically or horizontally on the bottom of the box.
- Blocks must lie flat on the bottom of the box.
- He may not move any block that his mother pre-placed.
- No block may overlap another.
- When finished, the bottom of the box must be completely covered with blocks.

You are to write a program that will determine if it is possible for Rocco to cover the bottom of the box given the placement of the blocks by his mother and given the lengths of the blocks he has to use. He does not have to use all of the blocks and he may select the blocks in any order.

Input

- The first line of input will contain a single integer `n` that indicates the number of data sets to follow.
- Each data set will consist of 7 lines.
- Each of the first six lines will contain six consecutive characters consisting only of:
  - An asterisk (`*`) that indicates a 3" spot Rocco's mother covered by a block or
  - A period (`.`) that indicates an empty 3" spot.
- The seventh line will be the lengths, in inches, of the blocks that Rocco can use to cover the bottom of the box, each separated by a space.

Output

For each data set and on a separate line, print `YES` if Rocco can cover the bottom of the box with the blocks given or `NO` if it is impossible.

Example Input File

```
2
......
..*...
..*...
...**.
.**...
...*..
12 6 6 3 9 3 12 9 12 3 6 6
......
..*...
*..*..
.*..**
...*..
..***.
9 12 9 9 3 6 6 6 12 6 12 9
```

Example Output to Screen

```
YES
NO
```