<u>COT 4210 Program #4: Turing Machine Simulator</u>

<u>The Problem (tm.java)</u>

Given a description of a Turing Machine and a string, simulate the string running on the Turing Machine for a given number of steps. At the end of the simulation, you are to determine if the machine accepted the string, rejected the string, or did not halt in the required number of steps.

<u>Input Format (standard input)</u>

The first line of the input file will contain a single positive integer, *n (n < 100)*, representing the number of Turing Machines that are going to be described in the file.

For each Turing Machine, the first line will have a positive integer, *q (2 < q < 100)*, representing the number of states and a positive integer, *r (r < 1000)*, representing the number of rules for the machine, separated by a space. By default, the states will be labeled 0 through q-1. ***0 will be the designated start state, 1 will be the designated accept state and 2 will be the designated reject state. The input alphabet will always be {a,b}. The tape alphabet will always be {a,b,B,#}, where B stands for a blank and # is a special tape symbol.***

You will be guaranteed that for all situations that arise, a rule will be present to apply to that situation. Thus, for example, in the first machine in the sample, it's impossible to get a situation where you ever read # because no rule ever writes # to the tape.

The next r lines will contain one rule each. Each of these lines will contain five items each separated by spaces:

1) An integer representing the input state
2) A character representing the character to read
3) An integer representing the output state
4) A character representing the character to write
5) Either L or R, representing which way the tape head should move

This will be followed by a line with positive integers, *n (n < 20)*, representing the number of input strings to test this particular Turing Machine on, and *s (s < 10000)*, the maximum number of steps to run each simulation. The following *n* lines will contain the strings to test, one string per line. These strings will be comprised solely of the characters 'a' and 'b'.

<u>Output Format (standard out)</u>

For each test case, output a header as follows: `Machine #k:` where k represents the grammar number, starting at 1.

For each of the following *s* lines, write the string in question, followed by a colon, followed by a space, followed by "YES", "NO", or "DOES NOT HALT IN S STEPS", where S is the maximum number of steps for the simulation.

Output these lines in the order in which the strings were given in the input file.Separate the output for each case with a blank line.

<u>Implementation Restrictions</u>

1) Write your program in Java, with standard input, standard output.

2) Submit **<u>tm.java</u>** via WebCourses.

<u>Sample Input</u>

```
2
5 9
0 B 2 B R
0 a 3 a R
0 b 4 b R
3 a 1 a R
3 b 1 b R
3 B 1 B R
4 a 2 a R
4 b 2 b R
4 B 2 B R
3 10
aaab
ba
b
7 20
0 B 2 B R
0 # 2 # R
0 a 3 B R
0 b 2 b R
3 # 3 # R
3 B 1 B R
3 a 4 # R
3 b 2 b R
4 # 4 # R
4 a 5 a R
4 B 6 B L
4 b 2 b R
5 # 5 # R
5 a 4 # R
5 B 2 B R
5 b 2 b R
6 a 6 a L
6 # 6 # L
6 B 3 B R
6 b 2 b R
3 10
a
aa
aaaaaaaa
```

Sample Output

```
Machine #1:
aaab: YES
ba: NO
b: NO

Machine #2:
a: YES
aa: YES
aaaaaaaa: DOES NOT HALT IN 10 STEPS
```