Problem J

The "Win-stay and Lose-shift" Strategy

Source file name: `wsls.c`, `wsls.cpp` *or* `wsls.java`

In recent weeks, geek tabloids have hit the newsstands around the world with a truly remarkable breakthrough in science: a group of researchers from Chinese universities have written a paper about the role of psychology in winning (or losing) at rock-paper-scissors (RPS). After studying how players change or keep their strategies during multiple-round sessions, the scientists figured out a basic rule that people tend to play by that could potentially be exploited. This rule is called the *win-stay and lose-shift* strategy.

An RPS *session* is a finite sequence of rounds played between two opponents. In each *round*, players simultaneously form one of three shapes with an outstretched hand: *rock* (`R`), *paper* (`P`), and *scissors* (`S`). Rock beats scissors, scissors beat paper, and paper beats rock; if both players throw the same shape, the round is tied. The *outcome of a round* for a player is 1 point if he/she wins, -1 point if he/she loses, and 0 points if it is a tie. The *outcome of a session* for a player is the sum over the outcome points of his/her rounds. For example, assume that a and b are playing a session of three rounds. In the first round a plays scissors and b plays paper; in the second round a plays paper and b plays paper; and, in the last round a plays rock and b plays rock. Then, the outcome of the first round for a is 1 point (for b is -1 point), and the outcomes of the second and third rounds for a is 0 points (for b is also 0 points). Consequently, the outcome of this session for a is 1 point and for b is -1 point.

During an RPS session, the *win-stay and lose-shift* strategy for a player p is as follows:

- If it is the first round or if it was a tie in the previous round, for the current round p makes a guess.
- If p lost in the previous round, for the current round p switches to the thing that beats p's opponent previous choice.
- If p won in the previous round, for the current round p switches to the thing that beats p's previous choice.

For example, assume that a and b are playing a session of three rounds, and a is playing under the win-stay and lose-shift strategy and that b plays as above. Initially a guesses `R` and loses (`P` beats `R`). In the second round a switches to `S` because it beats b's previous winning choice (i.e., `P`) and wins (`S` beats `P`). In the third round a switches to `R` because it beats a's previous choice (i.e., `S`) and ties (b also plays `R`). In this session the outcome for a is 0 points. However, this is not the only possible outcome for a under the win-stay and lose-shift strategy.

Given a session of n rounds for players a and b, and the probabilities of a guessing `R`, `P`, and `S` during the session, you are asked to write a program that decides if a's *expected* session outcome when playing under the win-stay and lose-shift strategy against b is better than a's actual session outcome.

Input

The first line of the input contains a non-negative integer number N ($N \geq 0$) indicating the number of test cases. Then N test cases follow, each consisting of three lines of input. The first and second lines of a test case contain, respectively, strings a and b only containing characters `R`, `P`, and `S` ($1 \leq |a| \leq 10^4$, $1 \leq |b| \leq 10^4$, with $|a| = |b|$) defining an RPS session of $|a|$ rounds played between players a and b. The third line of a test case contains three blank-separated integer numbers p_R, p_P, and p_S ($0 \leq p_R \leq 100$, $0 \leq p_P \leq 100$, $0 \leq p_S \leq 100$, with $p_R + p_P + p_S = 100$) indicating, respectively, the probability (amplified by 100) of a guessing rock, scissors, and paper.

The input must be read from standard input.

Output

For each test case output a single line containing three blank-separated quantities of the form

```
x y z
```

where

- x is an integer indicating a's actual session outcome against b,
- y is a floating point number indicating a's expected session outcome when playing against b with probabilities p_R, p_P, and p_S under the win-stay and lose-shift strategy (rounded up to exactly 4 decimal places, with no leading zeroes but at least one digit before the decimal point), and
- z is the character `Y` if y is strictly greater than x, and `N`, otherwise.

The output must be written to standard output.

Sample input	**Output for the sample input**

<pre>4
SPR
PPR
5 80 15
RRR
PPR
5 80 15
S
S
33 34 33
S
S
34 33 33</pre>
	<pre> 1 0.3060 N
	-2 0.3060 Y
	 0 -0.0100 N
	 0 0.0100 Y</pre>