Problem D

Paper Folding

Input: Standard Input
Output: Standard Output
Time Limit: 1 Second

Often when inserting a paper into a letter, you have to fold the paper one or more times to make it fit. Depending on how you fold it, different parts of the original view will be visible after the folding. You are to write a program which 1) calculates which part of the paper is at the front given a sequence of foldings, 2) calculates how to fold a paper given the wanted visible part.

There are **8** possible different ways to fold a paper if you want to halve the size of the paper. Those are:

- Folding the left half of the paper over the right half **('l')**
- Folding the right half of the paper over the left half **('r')**
- Folding the top half of the paper over the bottom half **('t')**
- Folding the bottom half of the paper over the top half **('b')**
- Folding the left half of the paper below the right half **('L')**
- Folding the right half of the paper below the left half **('R')**
- Folding the top half of the paper below the bottom half **('T')**
- Folding the bottom half of the paper below the top half **('B')**

In this problem, the size of the paper is **1048576*1048576** units (by pure coincidence, this means that the length of the sides of the paper after up to **20** foldings will always be an integer). The coordinates of the top left corner of the paper is **(0,0)** and the bottom left **(1048576,1048576)**. This is the case for both the front of the paper as well as the back. Thus, if you would (for some reason) turn the paper by flipping it horizontally, you would have the coordinate **0,0** to your upper right.

For instance, if your first fold is a '**t**', the top half of the paper is folded so the top half of the backside of the paper appears on the front. The visible part (the front) will then be **(0,0)-(1048576,524288)** from the back side of the original view.

You may assume that the foldings are razor sharp. In reality it would be next to impossible to fold a paper in half **20** times!

Input

The first line in the input file contains an integer **n** which is the number of cases to follow.

Next follows **n** lines, each describing either 1) a folding sequence or 2) a wanted visible part. A folding sequence is described as a string containing between **1** and **20** characters from the set of valid folding characters (**'l'**, **'r'**, **'t'**, **'b'**, **'L'**, **'R'**, **'T'**, **'B'**). A visible part is given in the format

(x1,y1)-(x2,y2) S where **x1,y1** and **x2,y2** are the coordinates of the corners of the part **(x1<x2, y1<y2)** and **S** is either **F** (if the front side of the original view is desired) or **B** (back side).

Output

For each input line you should output one line in the exact same format as the input (but for the opposite case!). See sample output for details.

In case there are more than one folding sequence (you may assume there are at least one) yielding the desired part, you should output the lexicographically smallest of them. That is, the preference order of the foldings should be '**B**', '**L**', '**R**', '**T**', '**b**', '**l**', '**r**', '**t**'.

Sample Input	Output for Sample Input
6 l (0,0)-(524288,1048576) B rLbb (786432,262144)-(1048576,524288) B BBtttTLbLRrLb (98304,319488)-(131072,323584) F	(0,0)-(524288,1048576) B l (786432,262144)-(1048576,524288) B bbLL (98304,319488)-(131072,323584) F BBbBBbbbLlllL

Problemsetter: Jimmy Mårdell, Member of Elite Problemsetters' Panel