|  |  |
| --- | --- |
| **Problem Statement** | |
|  | A robot is placed on an infinitely long line. Initially the position of the robot is 0. Cat Snuke sends commands to move this robot.  You are given a String **commands**. For each i, the i-th character of **commands** (0-based index) represents the i-th command Snuke sends. If the i-th character of **commands** is 'R', the robot moves one unit to the right (i.e., from position x to position x+1). If this character is 'L', the robot moves one unit to the left (i.e., from position x to position x-1). The robot has a built-in safety mechanism that prevents it from going too far and losing the signal. The safety mechanism makes sure that the robot always stays between the positions -**A** and **B**, inclusive. If the robot receives the command 'R' when the robot is at **B**, or the command 'L' when the robot is at -**A**, the command will be ignored.  You are given the String **commands** and the ints **A** and **B**. Return the final position of the robot. |
| **Definition** | |
|  | |  |  | | --- | --- | | Class: | OneDimensionalRobotEasy | | Method: | finalPosition | | Parameters: | String, int, int | | Returns: | int | | Method signature: | int finalPosition(String commands, int A, int B) | | (be sure your method is public) | | |
| **Limits** | |
|  | |  |  | | --- | --- | | Time limit (s): | 2.000 | | Memory limit (MB): | 256 | |
| **Constraints** | |
| - | **commands** will contain between 1 and 50 characters, inclusive. |
| - | Each character in **commands** will be either 'R' or 'L'. |
| - | **A** and **B** will be between 1 and 50, inclusive. |
| **Examples** | |
| 0) |  |
|  | |  |  |  |  | | --- | --- | --- | --- | | |  | | --- | | "RRLRRLLR" | | 10 | | 10 | | | Returns: 2 | | |  | | --- | | The robot will move as follows: 0 -> 1 -> 2 -> 1 -> 2 -> 3 -> 2 -> 1 -> 2. | | |
| 1) |  |
|  | |  |  |  |  | | --- | --- | --- | --- | | |  | | --- | | "RRRRR" | | 3 | | 4 | | | Returns: 4 | | |  | | --- | | The robot will move as follows: 0 -> 1 -> 2 -> 3 -> 4 -> 4. | | |
| 2) |  |
|  | |  |  |  |  | | --- | --- | --- | --- | | |  | | --- | | "LLLLLLLLLLR" | | 2 | | 6 | | | Returns: -1 | | |  | | --- | | The robot will move as follows: 0 -> -1 -> -2 -> -2 -> -2 -> -2 -> -2 -> -2 -> -2 -> -2 -> -2 -> -1. | | |
| 3) |  |
|  | |  |  |  |  | | --- | --- | --- | --- | | |  | | --- | | "RRRRRRRLRRLRRRRRRRRRRRRLRLRRRRRRRRLRRRRRLRRRRRRRRR" | | 5 | | 20 | | | Returns: 20 | | |  | | --- | |  | | |
| 4) |  |
|  | |  |  |  |  | | --- | --- | --- | --- | | |  | | --- | | "RLRLLLLLLLRLLLRLLLLLLLLRLLLLLRLLLRRLLLLLRLLLLLRLLL" | | 34 | | 15 | | | Returns: -30 | | |  | | --- | |  | | |

This problem statement is the exclusive and proprietary property of TopCoder, Inc. Any unauthorized use or reproduction of this information without the prior written consent of TopCoder, Inc. is strictly prohibited. (c)2003, TopCoder, Inc. All rights reserved.