|  |  |
| --- | --- |
| **Problem Statement** | |
|  | *NOTE: There are images in the examples section of this problem statement that help describe the problem. Please view the problem statement in the HTML window to view them.*  Given a picture composed entirely of horizontal and vertical line segments, calculate the minimum number of times you must lift your pen to trace every line segment in the picture exactly **n** times.  Each line segment will be of the form "<x1> <y1> <x2> <y2>" (quotes for clarity), representing a segment from (x1,y1) to (x2,y2). Segments may cross each other. Segments may also overlap, in which case you should count the overlapping region as appearing in the drawing only once. For example, say the drawing were composed of two lines: one from (6,4) to (9,4), and one from (8,4) to (14,4). Even though they overlap from (8,4) to (9,4), you should treat the drawing as if it were a single line from (6,4) to (14,4). You would not need to lift your pen at all to trace this drawing. |
| **Definition** | |
|  | |  |  | | --- | --- | | Class: | PenLift | | Method: | numTimes | | Parameters: | vector <string>, int | | Returns: | int | | Method signature: | int numTimes(vector <string> segments, int n) | | (be sure your method is public) | | |
|  |  |
|  |  |
| **Notes** | |
| - | The pen starts on the paper at a location of your choice. This initial placement does not count toward the number of times that the pen needs to be lifted. |
| **Constraints** | |
| - | **segments** will contain between 1 and 50 elements, inclusive. |
| - | Each element of **segments** will contain between 7 and 50 characters, inclusive. |
| - | Each element of **segments** will be in the format "<X1>\_<Y1>\_<X2>\_<Y2>" (quotes for clarity). The underscore character represents exactly one space. The string will have no leading or trailing spaces. |
| - | <X1>, <Y1>, <X2>, and <Y2> will each be between -1000000 and 1000000, inclusive, with no unnecessary leading zeroes. |
| - | Each element of **segments** will represent a horizontal or vertical line segment. No line segment will reduce to a point. |
| - | **n** will be between 1 and 1000000, inclusive. |
| **Examples** | |
| 0) |  |
|  | |  |  |  | | --- | --- | --- | | |  | | --- | | {"-10 0 10 0","0 -10 0 10"} | | 1 | | | Returns: 1 | | |  | | --- | | This picture looks like a plus sign centered at the origin. One way to trace this image is to start your pen at (-10,0), move right to (10,0), lift your pen and place it at (0,-10), and then move up to (0,10). There is no way to trace the picture without lifting your pen at all, so the method returns 1. | | |
| 1) |  |
|  | |  |  |  | | --- | --- | --- | | |  | | --- | | {"-10 0 0 0","0 0 10 0","0 -10 0 0","0 0 0 10"} | | 1 | | | Returns: 1 | | |  | | --- | | The picture is the same as the previous example, except that it has been described with four line segments instead of two. Therefore, the method still returns 1. | | |
| 2) |  |
|  | |  |  |  | | --- | --- | --- | | |  | | --- | | {"-10 0 0 0","0 0 10 0","0 -10 0 0","0 0 0 10"} | | 4 | | | Returns: 0 | | |  | | --- | | You are now required to trace each segment exactly 4 times. You can do so without lifting your pen at all. Start at (0,0). Move your pen left to (-10,0), then back right to (0,0), then left again to (-10,0), then right again to (0,0). You have now traced the first line segment 4 times. Repeat this process for the other three segments as well. Since no pen lifts were required, the method returns 0. | | |
| 3) |  |
|  | |  |  |  | | --- | --- | --- | | |  | | --- | | {"0 0 1 0", "2 0 4 0", "5 0 8 0", "9 0 13 0",  "0 1 1 1", "2 1 4 1", "5 1 8 1", "9 1 13 1",  "0 0 0 1", "1 0 1 1", "2 0 2 1", "3 0 3 1",  "4 0 4 1", "5 0 5 1", "6 0 6 1", "7 0 7 1",  "8 0 8 1", "9 0 9 1", "10 0 10 1", "11 0 11 1",  "12 0 12 1", "13 0 13 1"} | | 1 | | | Returns: 6 | | |  | | --- | | The picture looks like this:  http://www.topcoder.com/contest/problem/PenLift/penlift1.gif  To trace the picture using the minimum number of pen lifts, refer to the following diagram:  http://www.topcoder.com/contest/problem/PenLift/penlift2.gif  Start by placing your pen at the yellow dot. Trace the yellow square. Now lift your pen and place it on the red dot. Move downward, tracing the vertical line segment, and then around the perimeter of the red rectangle. Lift your pen again and place it on the green dot. Trace the green lines using the same method as you did for the red lines. Lift your pen a third time, placing it on the magenta dot. Trace the magenta lines in a similar fashion. You will need to lift your pen three more times to trace each of the leftover white segments, for a grand total of 6 pen lifts. | | |
| 4) |  |
|  | |  |  |  | | --- | --- | --- | | |  | | --- | | {"-2 6 -2 1", "2 6 2 1", "6 -2 1 -2", "6 2 1 2",  "-2 5 -2 -1", "2 5 2 -1", "5 -2 -1 -2", "5 2 -1 2",  "-2 1 -2 -5", "2 1 2 -5", "1 -2 -5 -2", "1 2 -5 2",  "-2 -1 -2 -6","2 -1 2 -6","-1 -2 -6 -2","-1 2 -6 2"} | | 5 | | | Returns: 3 | | |  | | --- | | This is an example of overlap. Once all the segments are drawn, the picture looks like this:  http://www.topcoder.com/contest/problem/PenLift/penlift3.gif  You would need to lift your pen 3 times to trace every segment in this drawing exactly 5 times. | | |
| 5) |  |
|  | |  |  |  | | --- | --- | --- | | |  | | --- | | {"-252927 -1000000 -252927 549481","628981 580961 -971965 580961",  "159038 -171934 159038 -420875","159038 923907 159038 418077",  "1000000 1000000 -909294 1000000","1000000 -420875 1000000 66849",  "1000000 -171934 628981 -171934","411096 66849 411096 -420875",  "-1000000 -420875 -396104 -420875","1000000 1000000 159038 1000000",  "411096 66849 411096 521448","-971965 580961 -909294 580961",  "159038 66849 159038 -1000000","-971965 1000000 725240 1000000",  "-396104 -420875 -396104 -171934","-909294 521448 628981 521448",  "-909294 1000000 -909294 -1000000","628981 1000000 -909294 1000000",  "628981 418077 -396104 418077","-971965 -420875 159038 -420875",  "1000000 -1000000 -396104 -1000000","-971965 66849 159038 66849",  "-909294 418077 1000000 418077","-909294 418077 411096 418077",  "725240 521448 725240 418077","-252927 -1000000 -1000000 -1000000",  "411096 549481 -1000000 549481","628981 -171934 628981 923907",  "-1000000 66849 -1000000 521448","-396104 66849 -396104 1000000",  "628981 -1000000 628981 521448","-971965 521448 -396104 521448",  "-1000000 418077 1000000 418077","-1000000 521448 -252927 521448",  "725240 -420875 725240 -1000000","-1000000 549481 -1000000 -420875",  "159038 521448 -396104 521448","-1000000 521448 -252927 521448",  "628981 580961 628981 549481","628981 -1000000 628981 521448",  "1000000 66849 1000000 -171934","-396104 66849 159038 66849",  "1000000 66849 -396104 66849","628981 1000000 628981 521448",  "-252927 923907 -252927 580961","1000000 549481 -971965 549481",  "-909294 66849 628981 66849","-252927 418077 628981 418077",  "159038 -171934 -909294 -171934","-252927 549481 159038 549481"} | | 824759 | | | Returns: 19 | |  | |

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.