

```

1  /*Author: Bochen (mddboc@foxmail.com)
2  Last Modified: Tue Apr 10 22:28:44 CST 2018*/
3
4  /*Rotate an array of n elements to the right by k steps.
5
6  .....For example, with n = 7 and k = 3, the array [1,2,3,4,5,6,7] is rotated to
7  .....[5,6,7,1,2,3,4].
8
9  .....Note:
10 .....Try to come up as many solutions as you can, there are at least 3 different
11 .....ways to solve this problem.
12 */
13
14 import java.util.*;
15 import java.lang.Math;
16 import java.lang.System;
17 import java.lang.Integer;
18
19
20 public class Main {
21
22     ....public static void main(String[] args) throws ArithmeticException {
23
24         ....TreeNode root = new TreeNode(1);
25         ....root.left = new TreeNode(2);
26         ....root.right = new TreeNode(2);
27         ....root.left.left = new TreeNode(3);
28         ....root.left.right = new TreeNode(4);
29         ....root.right.left = new TreeNode(4);
30         ....root.right.right = new TreeNode(3);
31
32         ....boolean result = new Solution().isSymmetric(root);
33
34         ....System.out.println(result);
35     ....}
36
37 }
38
39
40 class ListNode {
41     ....int val;
42     ....ListNode next;
43
44     ....ListNode(int x) {
45         ....val = x;
46     ....}
47 }
48
49
50 class TreeNode {
51     ....int val;
52     ....TreeNode left;
53     ....TreeNode right;
54
55     ....TreeNode(int x) {
56         ....val = x;
57     ....}
58 }
59
60
61 class Solution {
62     ....public void rotate(int[] nums, int k) {
63
64         ———> ....// 方法一：需要开辟新空间，更快
65         ———> ..../* int numsLength = nums.length;
66         ....k = k % numsLength;
67
68         ....int[] helper = new int[k];
69
70         ....System.arraycopy(nums, numsLength - k, helper, 0, k);
71         ....System.arraycopy(nums, 0, nums, k, numsLength - k);

```

```
72     .... System.arraycopy(helper, 0, nums, 0, k); */
73     ——>
74     ——> .... // 方法二：不需要开辟新空间，相对慢点
75     .... int numsLength = nums.length;
76     .... k = k % numsLength;
77
78     .... reverse(nums, 0, numsLength - 1);
79     .... reverse(nums, 0, k - 1);
80     .... reverse(nums, k, numsLength - 1);
81     .... }
82
83     .... private void reverse(int[] nums, int startIndex, int endIndex) {
84
85     .... while (startIndex < endIndex) {
86     ....     .... int temp = nums[startIndex];
87     ....     .... nums[startIndex] = nums[endIndex];
88     ....     .... nums[endIndex] = temp;
89
90     ....     .... startIndex++;
91     ....     .... endIndex--;
92     .... }
93     .... }
94 }
```