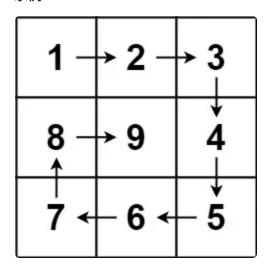
59.螺旋矩阵Ⅱ

题目描述:

给你一个正整数 n ,生成一个包含 1 到 n^2 所有元素,且元素按顺时针顺序螺旋排列的 n x n 正方形矩阵 matrix 。

示例 1:



1 输入: n = 3

2 输出: [[1,2,3],[8,9,4],[7,6,5]]

示例 2:

1 输入: n = 1 2 输出: [[1]]

示例3:

1	2	3	4	5
16	17	18	19	6
15	24		20	7
14	23	22	21	8
13	12	11	10	9

解题思路:

本题看似简单,实则容易出错! 得考虑清楚变量之间的关系,使用模拟的思路,要维护的变量众多:

- 1. 维护 loop 的次数,为n/2次;
- 2. 维护 startx,starty 初始为0,每一圈增加1;
- 3. 维护 offset(边界余量),为了平衡每次在各个方向上走的路程是有一样的,应该预留位置给下一条边;然后通过loop去模拟为四条边赋值的情况,最后别忘了如果是奇数的情况要为 nums[mid][mid] 赋上最大值;

代码:

▼ 59.螺旋矩阵II Java

```
1 * class Solution {
         public int[][] generateMatrix(int n) {
             int[][] nums = new int[n][n];
3
4
             int startx = 0, starty = 0, i = 0, j = 0;
5
6
             int count = 1, offset = 1;
             int loop = 0;
7
             int mid=n/2;
8
9 =
             while (loop++ < n / 2) {
                 for (j = starty; j < n - offset; j++) {
10 -
                      nums[startx][j] = count++;
11
12
                 for (i = startx; i < n - offset; i++) {</pre>
13 -
14
                      nums[i][j] = count++;
15
16 -
                 for (; j > starty; j--) {
                      nums[i][j] = count++;
17
18
19 -
                 for (; i > startx; i--) {
20
                      nums[i][j] = count++;
21
22
                 offset++;
23
                 startx++;
24
                 starty++;
25
             }
26 -
             if (n % 2 == 1) {
27
                 nums[mid] [mid] = count;
28
             }
29
             return nums;
         }
30
    }
31
```