```
/*
给定一个正整数 n，生成一个包含 1 到 n2 所有元素，且元素按顺时针顺序螺旋排列的正方形矩阵。
示例:
输入: 3
输出:
[
 [ 1, 2, 3 ],
 [ 8, 9, 4 ],
 [ 7, 6, 5 ]
]
来源：力扣（LeetCode）
链接：https://leetcode-cn.com/problems/spiral-matrix-ii
著作权归领扣网络所有。商业转载请联系官方授权，非商业转载请注明出处。
*/
```

分析:

- 根据题意，我们必须一圈一圈写，直到最后中心位置剩下1个空格未填或4个空格未填.
  - 方法一:尾递归法
  - 方法二:迭代法

---

方法一:C++_尾递归法

```
class Solution
{
    void __fillN4Edge(vector<vector<int>>& vvi,int n,int rows_start,int idx_start)
    {
        int cols_start = rows_start;
        if(n==0)
        {
            return;
        }
        if(n==1)
        {
            vvi[rows_start][cols_start] = idx_start;
            return ;
        }

        if(n==2)
        {
            vvi[rows_start][cols_start]         = idx_start++;
            vvi[rows_start][cols_start+1]       = idx_start++;
            vvi[rows_start+1][cols_start+1]     = idx_start++;
            vvi[rows_start+1][cols_start]       = idx_start;

            return ;
        }

        /* up edge */
        for(int i = 0 ; i < n ; i++)
        {
            vvi[rows_start][cols_start+i] = idx_start++;
```

```
        }

        /* right edge */
        for(int i = 1 ; i < n ; i++)
        {
            vvi[rows_start+i][cols_start+n-1] = idx_start++;
        }

        /* bottom edge*/
        for(int i = cols_start+n-2 ; i >=cols_start ; i--)
        {
            vvi[rows_start+n-1][i] = idx_start++;
        }

        /* left edge*/
        for(int i = rows_start+n-2; i > rows_start;i--)
        {
            vvi[i][cols_start] = idx_start++;
        }

        /* tail recursion*/
        __fillN4Edge(vvi,n-2,rows_start+1,idx_start);
    }

    public:
        vector<vector<int>> generateMatrix(int n)
        {
            vector<vector<int>> ret_val;

            if(n>0)
            {
                vector<int> temp;
                for(int i = 0; i < n ; i++)
                {
                    temp.push_back(0);
                }
                for(int i = 0; i < n ; i++)
                {
                    ret_val.push_back(temp);
                }

                __fillN4Edge(ret_val,n,0,1);
            }
            return ret_val;
        }
};


/*
执行结果:
通过
显示详情
执行用时 :8 ms, 在所有 C++ 提交中击败了68.56% 的用户
内存消耗 :9.1 MB, 在所有 C++ 提交中击败了11.81%的用户
*/
```

方法一:C++_迭代法

```
class Solution
{
    public:
        vector<vector<int>> generateMatrix(int n)
        {
            vector<vector<int>> ret_val;
            int i = 0;

            if(n>0)
            {
                vector<int> temp;
                for(i = 0; i < n ; i++)
                {
                    temp.push_back(0);
                }
                for(i = 0; i < n ; i++)
                {
                    ret_val.push_back(temp);
                }

                //__fillN4Edge(ret_val,n,0,1);
                int rows_start = 0;
                int idx_start  = 1;
                int cols_start = 0;
                while(n>0)
                {
                    if(n==1)
                    {
                        ret_val[rows_start][cols_start] = idx_start;
                        break;
                    }
                    if(n==2)
                    {
                        ret_val[rows_start][cols_start]         =
idx_start++;
                        ret_val[rows_start][cols_start+1]       =
idx_start++;
                        ret_val[rows_start+1][cols_start+1]     =
idx_start++;
                        ret_val[rows_start+1][cols_start]       =
idx_start;
                        break;
                    }

                    /* up edge */
                    for(i = 0 ; i < n ; i++)
                    {
                        ret_val[rows_start][cols_start+i] = idx_start++;
                    }

                    /* right edge */
                    for(i = 1 ; i < n ; i++)
                    {
                        ret_val[rows_start+i][cols_start+n-1] =
idx_start++;
```

```
                    }

                    /* bottom edge*/
                    for(i = cols_start+n-2 ; i >=cols_start ; i--)
                    {
                        ret_val[rows_start+n-1][i] = idx_start++;
                    }

                    /* left edge*/
                    for(i = rows_start+n-2; i > rows_start;i--)
                    {
                        ret_val[i][cols_start] = idx_start++;
                    }

                    n = n - 2;
                    rows_start++;
                    cols_start++;
                }


            }

            return ret_val;
        }
};


/*
执行结果:
通过
显示详情
执行用时 :4 ms, 在所有 C++ 提交中击败了94.85% 的用户
内存消耗 :8.9 MB, 在所有 C++ 提交中击败了51.11%的用户
*/

```

AlimyBreak
2019.10.01