```
/*
给出 R 行 C 列的矩阵，其中的单元格的整数坐标为 (r, c)，满足 0 <= r < R 且 0 <= c < C。
另外，我们在该矩阵中给出了一个坐标为 (r0, c0) 的单元格。
返回矩阵中的所有单元格的坐标，并按到 (r0, c0) 的距离从最小到最大的顺序排，其中，两单元格(r1, c1) 和 (r2, c2) 之间的距离是曼哈顿距离，|r1 - r2| + |c1 - c2|。（你可以按任何满足此条件的顺序返回答案。）
示例 1：
输入：R = 1, C = 2, r0 = 0, c0 = 0
输出：[[0,0],[0,1]]
解释：从 (r0, c0) 到其他单元格的距离为：[0,1]
示例 2：
输入：R = 2, C = 2, r0 = 0, c0 = 1
输出：[[0,1],[0,0],[1,1],[1,0]]
解释：从 (r0, c0) 到其他单元格的距离为：[0,1,1,2]
[[0,1],[1,1],[0,0],[1,0]] 也会被视作正确答案。
示例 3：
输入：R = 2, C = 3, r0 = 1, c0 = 2
输出：[[1,2],[0,2],[1,1],[0,1],[1,0],[0,0]]
解释：从 (r0, c0) 到其他单元格的距离为：[0,1,1,2,2,3]
其他满足题目要求的答案也会被视为正确，例如 [[1,2],[1,1],[0,2],[1,0],[0,1],[0,0]]。
提示：
    1 <= R <= 100
    1 <= C <= 100
    0 <= r0 < R
    0 <= c0 < C
来源：力扣（LeetCode）
链接：https://leetcode-cn.com/problems/matrix-cells-in-distance-order
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:

- 通过画图可知道,与$(r0,c0)$点距离为d的点是4条直线围成的正方形,我们只要依次枚举这个4条直线上的每个点,判断每个点的坐标是否在允许范围内即可.

方法一:C++_直线方程遍历法

```
class Solution
{
    bool check_point_valid(int R, int C,int x, int y)
    {
        return ( (x>=0) && (x<R) && (y>=0) && ( y < C) );

    }


    public:
        vector<vector<int>> allCellsDistOrder(int R, int C, int r0, int c0)
        {
            vector<vector<int>> vvi_ret                             ;
            vector<int>         vi                                  ;
```

```
            int                 edge_distance[4]        =   {0,}    ;   //l_t_to_point,r_t_to_point,l_b_to_point,r_b_to_point
            int                 max_distance            =   0       ;
            int                 i                       =   0       ;
            int                 j                       =   0       ;
            int                 x                       =   0       ;
            int                 y                       =   0       ;

            /*计算最长距离*/
            edge_distance[0]    =   abs(0-r0)   +   abs(0-c0);
            edge_distance[1]    =   abs(0-r0)   +   abs(C-1-c0);
            edge_distance[2]    =   abs(R-1-r0) +   abs(0-c0);
            edge_distance[3]    =   abs(R-1-r0) +   abs(C-1-c0);

            max_distance = edge_distance[0];
            for(i = 1;i < 4;i++)
            {
                if(max_distance < edge_distance[i])
                {
                    max_distance = edge_distance[i];
                }
            }


            /*按距离从小到大遍历(菱形)*/
            vi.push_back(r0);
            vi.push_back(c0);
            vvi_ret.push_back(vi);
            vi.clear();
            for(i=1;i<=max_distance;i++)
            {
                /*
                Left-->Bottom:(r0,c0-i) ---> (r0+i,c0),不到buttom点
                */
                for(j=0;j<i;j++)
                {
                    x = r0 + j;     //这里的x表示行下标 , y表示列下表
                    y = c0 - i + j;
                    if(check_point_valid(R,C,x,y))
                    {
                        vi.push_back(x);
                        vi.push_back(y);
                        vvi_ret.push_back(vi);
                        vi.clear();
                    }
                }

                /*
                Bottom-->Right:(r0+i,c0) --->(r0,c0+i),不到right点
                */
                for(j=0;j<i;j++)
                {
                    x = r0 + i - j;
                    y = c0 + j;
                    if(check_point_valid(R,C,x,y))
                    {
                        vi.push_back(x);
                        vi.push_back(y);
```

```
                        vvi_ret.push_back(vi);
                        vi.clear();
                    }
                }

                /*
                Right-->Top:(r0,c0+i) --->(r0-i,c0),不到top点
                */
                for(j=0;j<i;j++)
                {
                    x = r0 - j;
                    y = c0 + i - j ;
                    if(check_point_valid(R,C,x,y))
                    {
                        vi.push_back(x);
                        vi.push_back(y);
                        vvi_ret.push_back(vi);
                        vi.clear();
                    }
                }

                /*
                Top-->Left:(r0-i,c0) --->(r0,c0-i) ,不到top点
                */
                for(j=0;j<i;j++)
                {
                    x = r0 - i + j;
                    y = c0 - j ;
                    if(check_point_valid(R,C,x,y))
                    {
                        vi.push_back(x);
                        vi.push_back(y);
                        vvi_ret.push_back(vi);
                        vi.clear();
                    }
                }
            }
            return vvi_ret;
        }
};

/*
执行结果:
通过
显示详情
执行用时 :124 ms, 在所有 cpp 提交中击败了98.05% 的用户
内存消耗 :17.3 MB, 在所有 cpp 提交中击败了100.00%的用户
*/
```

AlimyBreak
2019.11.08