**难度: 简单**

给定一个非负整数 numRows，生成杨辉三角的前 numRows 行。



在杨辉三角中，每个数是它左上方和右上方的数的和。

示例:

输入: 5

输出:

[

[1],

[1,1],

[1,2,1],

[1,3,3,1],

[1,4,6,4,1]

]

来源：力扣（LeetCode）

链接：https://leetcode-cn.com/problems/pascals-triangle

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。

**思路:**

1). 从图可以看出每次第一个和最后一个都是1

2). 从第三行开始就每次用第一个加上第二个给到第四行的第二个

3). 从第四行开始就每次用第一个加上第二个给到第五行的第二个,然后第二个加上第三个给到第三个

代码有详细注释

|  |
| --- |
| public List<List<Integer>> generate(int numRows) {  //创建一个Lists来存储list  List<List<Integer>> lists = new ArrayList<>();  for(int i = 0; i < numRows; i++) {  //每次创建一个list来保存  List<Integer> list = new ArrayList<>();  for (int j = 0; j <= i; j++) {  //判断是不是第一个和最后一个  if (j == 0 || j== i) {  //如果是直接添加1  list.add(1);  }else {  //获取上一行的list,然后获取它的下表  //这里j是不会等于0的,因为上面已经判断了,因此如果这个成立  //比如是第三行i为2,j为1的时候  //i-1也就是2-1,就获取了第二行的,然后获取它的值j-1也就是1-1  //也就获取了第二行第一个,得到1,然后加同样的第二行的第二个  //i-1也就是2-1,并且j为1也就是第二行的第二个  //不需要当心越界的情况,因为如果是最后一个,同样不会执行这个语句  list.add(lists.get(i-1).get(j-1)+lists.get(i-1).get(j));  }  }  //把list的内容添加到lists中  lists.add(list);  }  return lists;  } |