Problem #2373: Largest Local Values in a Matrix

<https://leetcode.com/problems/largest-local-values-in-a-matrix/description/>

My Solution: Runtime beats 68.32% and memory beats 99.51%

**Intuition**

Consider submatrix starting at row 0 and column 0.  
Then slide by one column to the right at a time until that row is done.  
Do this for each row.  
For each submatrix find the max element and save this to the result matrix.

**Approach**

1. Consider submatrices of size 3 x 3 in the grid and starting with row 0 + next 2 rows and starting with column 0 + next 2 columns. Then in row 0 + next 2 rows, move to column 1 + next 2 columns. Keep sliding on the same rows to the end of the columns, i.e. row 0 + next 2 rows and column (n-3) + next 2 columns.
2. Find the maximum element in the submatix and append it to the temp\_list
3. When done with starting row 0, then append the temp\_list.
4. Go back to step 1, except this time we will have the submatrix starting with row 1 + next 2 rows and col 0 + next 2 columns. Repeat steps 2 and 3 for the submatrix.
5. Repeat step 4 until submatrices starting in row n-3 are done.
6. Return the result matrix.

**Complexity**

* Time complexity:

O(n^2)

* Space complexity:

O(n)

class Solution:

def findMax(self, matrix: List[List[int]]) -> int:

num\_rows = len(matrix)

num\_cols = len(matrix[0])

max\_num = matrix[0][0]

for r in range(num\_rows):

for c in range(num\_cols):

if matrix[r][c] > max\_num:

max\_num = matrix[r][c]

return max\_num

def largestLocal(self, grid: List[List[int]]) -> List[List[int]]:

n = len(grid)

result = []

for i in range(n - 2):

temp\_list = []

for j in range(n - 2):

matrix = [ l[j: j+3] for l in grid[i : i + 3]]

max\_ele = self.findMax(matrix)

temp\_list.append(max\_ele)

result.append(temp\_list)

return result