Problem # 74: Search a 2D Matrix (Medium)

<https://leetcode.com/problems/search-a-2d-matrix/>

My Solution:

<https://leetcode.com/problems/search-a-2d-matrix/discuss/897785/Simple-Python-3-solution-Runtime-beats-99.86>

Simple Python 3 solution – Runtime beats 99.86%

1. If there are no rows or no columns return False.
2. If the target is equal to the first element of the matrix or the target is equal to the last element of the matrix, then return True.
3. If the target is less than the first element of the matrix or the target is more than the last element of the matrix, then return False
4. Create an array arr with only the candidate row. For this check the last element of every row whether it is equal to the target. If so, return True. If last element of the row is less than the target, then it is the candidate row. Traverse through the candidate row and append the values in the candidate row to arr.
5. Now arr has number of elements equal to the number of columns of the matrix. Perform a binary search on it to find the element which is equal to the target.

class Solution:

def searchMatrix(self, matrix: List[List[int]], target: int) -> bool:

rows = len(matrix)

if not rows:

return(False)

cols = len(matrix[0])

if not cols:

return(False)

if target == matrix[0][0] or target == matrix[-1][-1]:

return(True)

if target < matrix[0][0] or target > matrix[-1][-1]:

return(False)

# Create an array with just the candidate row

arr = []

i = 0

while i < rows:

if target == matrix[i][-1]:

return(True)

elif target < matrix[i][-1]:

for j in range(cols):

arr.append(matrix[i][j])

break

i += 1

# Perform Binary search on arr which is just a single row

low = 0

high = cols - 1

while low < high:

middle = (low + high) // 2

if target in [arr[low], arr[middle], arr[high]]:

return(True)

elif target < arr[middle]:

high = middle - 1

else:

low = middle + 1

return(False)