EXERCISE-57: Find the Kth Smallest Sum of a Matrix With Sorted Rows You are given an m x n matrix mat that has its rows sorted in non-decreasing order and an integer k. You are allowed to choose exactly one element from each row to form an array. Return the kth smallest array sum among all possible arrays. Example 1: Input: mat = [[1,3,11],[2,4,6]], k = 5 Output: 7 Explanation: Choosing one element from each row, the first k smallest sum are: [1,2], [1,4], [3,2], [3,4], [1,6]. Where the 5th sum is 7. Example 2: Input: mat = [[1,3,11],[2,4,6]], k = 9 Output: 17

PROGRAM:

```
import heapq

def kth_smallest_sum(mat, k):
    m, n = len(mat), len(mat[0])
    heap = [(sum(row[0] for row in mat), [0] * m)]
    for _ in range(k):
        curr_sum, indices = heapq.heappop(heap)
        for i in range(m):
            if indices[i] + 1 < n:
                new_indices = indices[:]
                new_indices[i] += 1
                new_sum = curr_sum - mat[i][indices[i]] + mat[i][new_indices[i]]
                heapq.heappush(heap, (new_sum, new_indices))
    return curr_sum
print(kth_smallest_sum([[1,3,11],[2,4,6]], 5))
print(kth_smallest_sum([[1,3,11],[2,4,6]], 9))
```

output;

```
7
9
```

Time complexity;

O(k * log(m)).