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

AIM: To Find the Kth Smallest Sum of a Matrix With Sorted Rows

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)]
    while k > 1:
        _, indices = heapq.heappop(heap)
        for i in range(m):
            if indices[i] < n - 1:
                new_indices = indices[:]
                new_indices[i] += 1
                new_sum = sum(mat[row][new_indices[row]] for row in range(m))
                heapq.heappush(heap, (new_sum, new_indices))
        k -= 1

    return heapq.heappop(heap)[0]
print(kth_smallest_sum([[1,3,11],[2,4,6]], 5))
```

OUTPUT: 

TIME COMPLEXITY: O(k*log n)