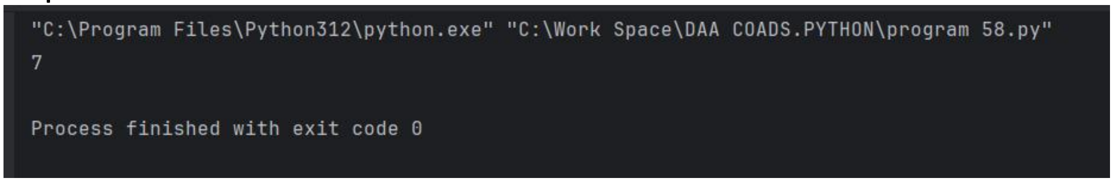
**Program 58.**ind the Kth Smallest Sum of a Matrix With Sorted Rows
You are given an m x n matrix mat that has its rows sorted in non-decreasing order and an integer k.
You are allowed to choose exactly one element from each row to form an array.
Return the kth smallest array sum among all possible 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.

**Program:**

```
import heapq
def kthSmallest(mat, k):
    # Start with the smallest possible sum using the first element of each row
    m, n = len(mat), len(mat[0])
    min_heap = [(sum(row[0] for row in mat), [0] * m)]  # (sum, indices)
    visited = set(tuple([0] * m))  # To keep track of visited indices

    while k > 0:
        current_sum, indices = heapq.heappop(min_heap)
        k -= 1
        if k == 0:
            return current_sum

        # Try to increment each index in the indices array
        for i in range(m):
            if indices[i] + 1 < n:
                new_indices = list(indices)
                new_indices[i] += 1
                new_sum = current_sum - mat[i][indices[i]] + mat[i][new_indices[i]]
                new_tuple = tuple(new_indices)

                if new_tuple not in visited:
                    visited.add(new_tuple)
                    heapq.heappush(min_heap, (new_sum, new_indices))

# Example usage
mat = [[1, 3, 11], [2, 4, 6]]
k = 5
print(kthSmallest(mat, k))  # Output: 7
```

**Output:**



```
"C:\Program Files\Python312\python.exe" "C:\Work Space\DAA COADS.PYTHON\program 58.py"
7

Process finished with exit code 0
```

**Time complexity:**
**O(m*(k log k)**