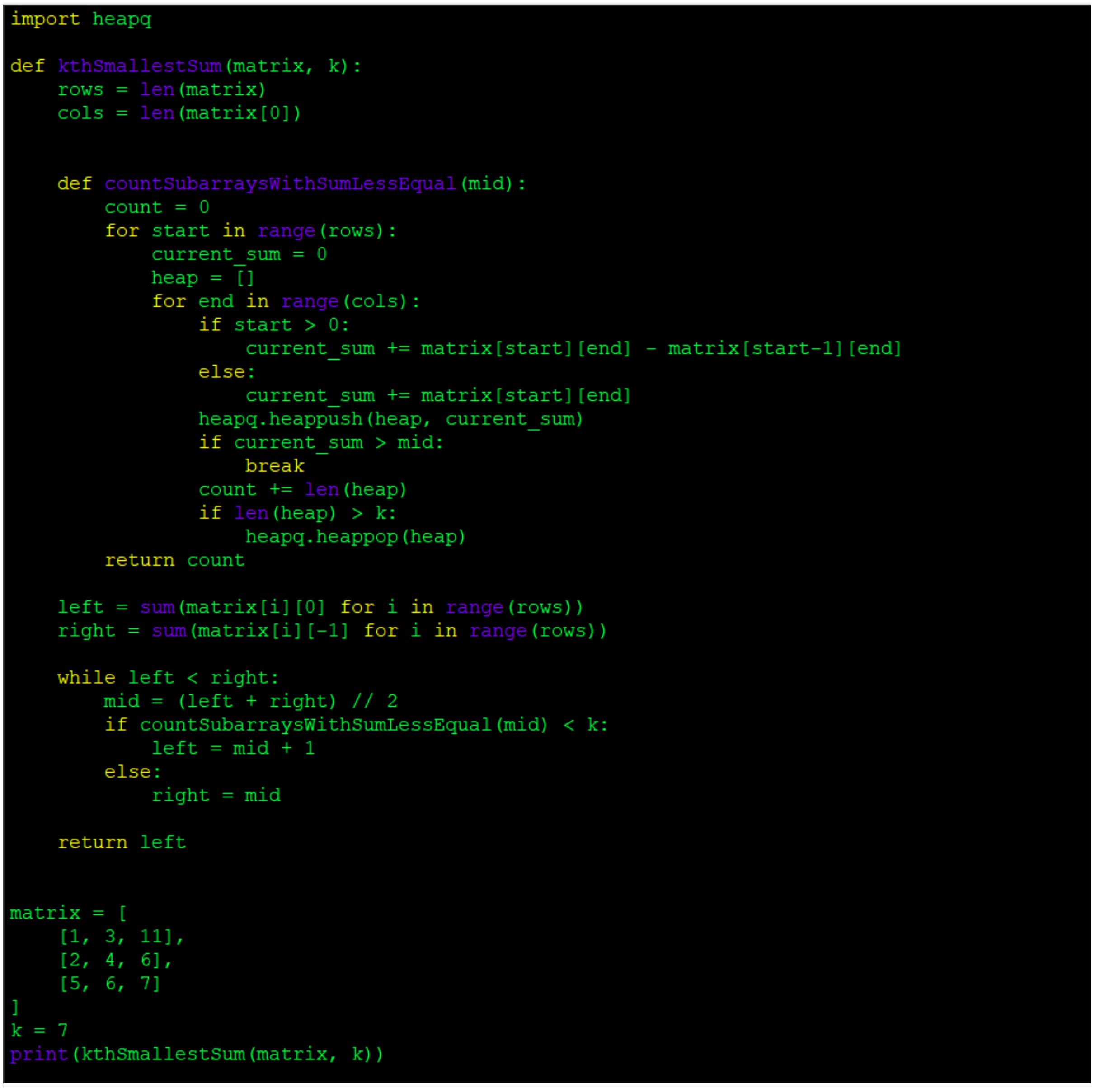
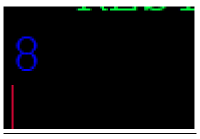
57. Find the kth smallest sum of matrix with sorted rows

Code:



```python
import heapq

def kthSmallestSum(matrix, k):
    rows = len(matrix)
    cols = len(matrix[0])


    def countSubarraysWithSumLessEqual(mid):
        count = 0
        for start in range(rows):
            current_sum = 0
            heap = []
            for end in range(cols):
                if start > 0:
                    current_sum += matrix[start][end] - matrix[start-1][end]
                else:
                    current_sum += matrix[start][end]
                heapq.heappush(heap, current_sum)
                if current_sum > mid:
                    break
                count += len(heap)
                if len(heap) > k:
                    heapq.heappop(heap)
        return count

    left = sum(matrix[i][0] for i in range(rows))
    right = sum(matrix[i][-1] for i in range(rows))

    while left < right:
        mid = (left + right) // 2
        if countSubarraysWithSumLessEqual(mid) < k:
            left = mid + 1
        else:
            right = mid

    return left


matrix = [
    [1, 3, 11],
    [2, 4, 6],
    [5, 6, 7]
]
k = 7
print(kthSmallestSum(matrix, k))
```

Output:



```
8
```

Time Complexity:

- T(n)= O(klogk)