Question solution:

In this question, I found that in some cases, for a given matrix and summed number, there are a variety of sets of operations that can get the number. And my code outputs one set of these operations for a given matrix and number.

First, I find that in a given m * n matrix, the sum of the values in down operations path is always fixed:

downSum =
$$1 + 2 + 3 + 4 + ... + m$$

And, the time of right operations is also fixed:

$$rightTimes = n - 1$$

Thus, the question is transformed to find n-1 numbers from 1 to m (Repeatable), and the sum of the numbers is equal to totalSum – downSum.

My method is to assume that all numbers are 1, and try to replace the numbers with the largest number as many as possible. When the sum is out of bounds, replace the numbers with a smaller number. I define a list (called rightPosition) to record the times of different numbers replaced. The values in the list represent the times of right operations in the specified position.

Results:

a. For m=9, n=9 matrix, find the operations for the following summed numbers: 65, 72, 90, 110.

65 RRRRRRDDDDRDDDDR

72 RRRRRDDDRDDDDRR

90 RRRDDDDDDDDDRRRR

110 DRDDDDDDDRRRRRRR

b. For m=90,000, n=100,000 matrix (90,000 rows, 100,000 columns), find the operations for the following summed numbers: 87127231192 and 5994891682.

87127231192 Cannot find the operations

5994891682