DESIGN AND ANALYSIS OF ALGORITHM

**DAY -10 PROGRAMS**

**NAME: G. GANESH**

**REGISTER NO:192373008**

**COURSE CODE: CSA0689**

**1.** Discuss the importance of visualizing the solutions of the N-Queens Problem to understand

the placement of queens better. Use a graphical representation to show how queens are

placed on the board for different values of N. Explain how visual tools can help in debugging

the algorithm and gaining insights into the problem's complexity. Provide examples of visual

representations for N = 4, N = 5, and N = 8, showing different valid solutions.

a. Visualization for 4-Queens:

Input: N = 4

Output:

Explanation: Each 'Q' represents a queen, and '.' represents an empty space.

b. Visualization for 5-Queens:

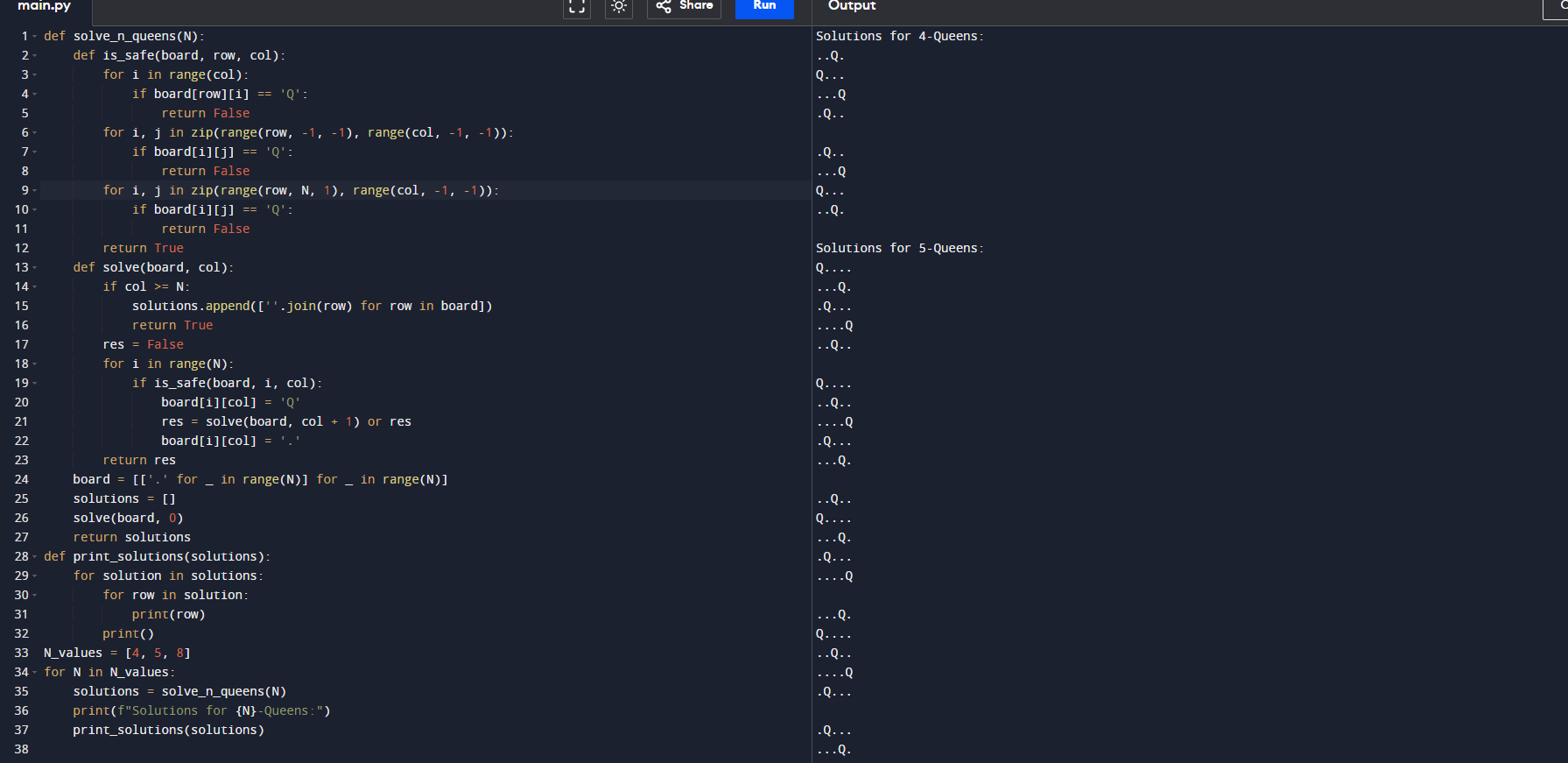
Input: N = 5

Output:

c. Visualization for 8-Queens:

Input: N = 8

Output:



1. Discuss the generalization of the N-Queens Problem to other board sizes and shapes, such as

rectangular boards or boards with obstacles. Explain how the algorithm can be adapted to

handle these variations and the additional constraints they introduce. Provide examples of

solving generalized N-Queens Problems for different board configurations, such as an 8×10

board, a 5×5 board with obstacles, and a 6×6 board with restricted positions.

a. 8×10 Board:

8 rows and 10 columns

Output: Possible solution [1, 3, 5, 7, 9, 2, 4, 6]

Explanation: Adapt the algorithm to place 8 queens on an 8×10 board, ensuring no two

queens threaten each other.

b. 5×5 Board with Obstacles:

Input: N = 5, Obstacles at positions [(2, 2), (4, 4)]

Output: Possible solution [1, 3, 5, 2, 4]

Explanation: Modify the algorithm to avoid placing queens on obstacle positions,

ensuring a valid solution that respects the constraints.

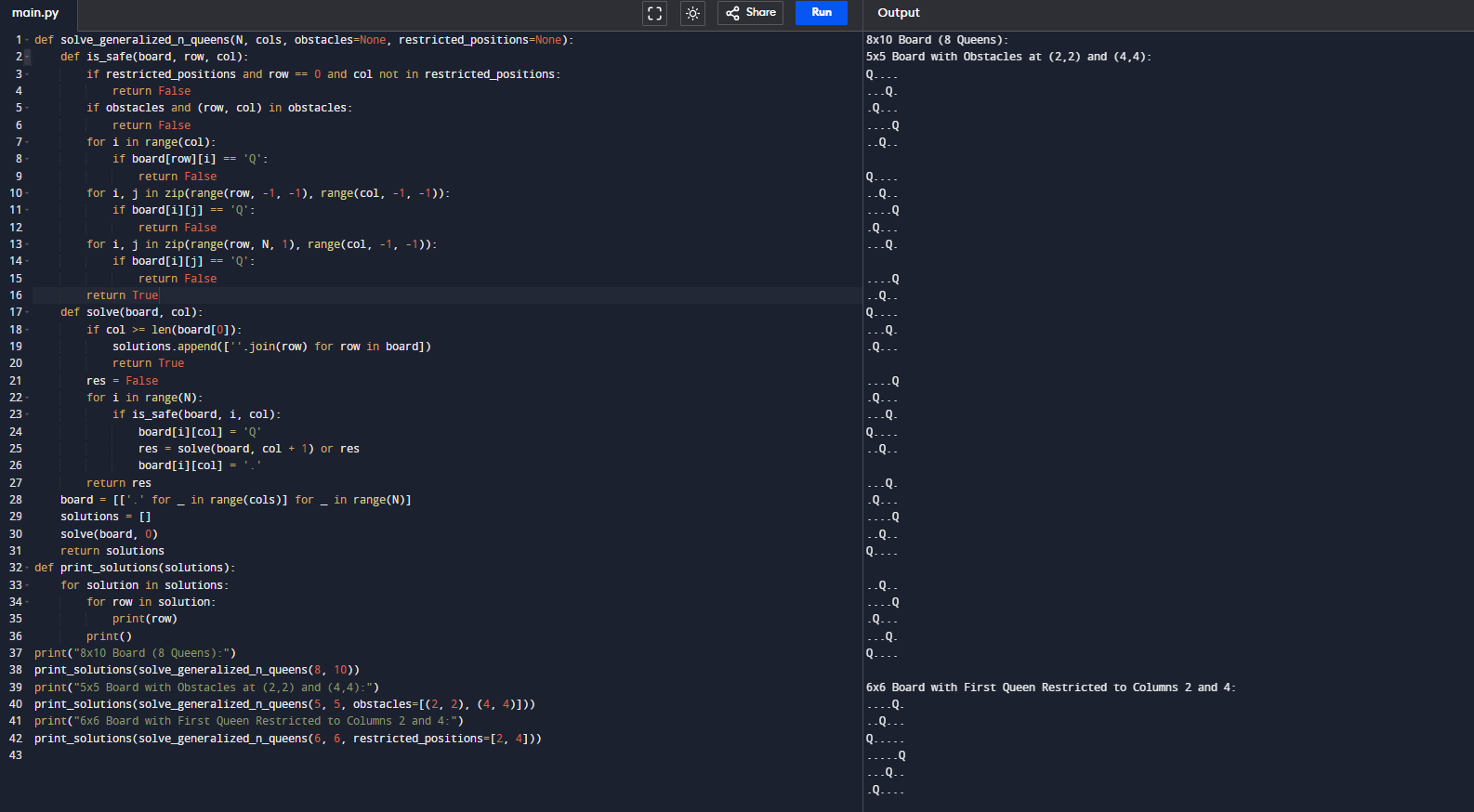
c. 6×6 Board with Restricted Positions:

Input: N = 6, Restricted positions at columns 2 and 4 for the first queen

Output: Possible solution [1, 3, 5, 2, 4, 6]

Explanation: Adjust the algorithm to handle restricted positions, ensuring the queens are

placed without conflicts and within allowed columns.



**3.** Write a program to solve a Sudoku puzzle by filling the empty cells.A sudoku solution must

satisfy all of the following rules:Each of the digits 1-9 must occur exactly once in each

row.Each of the digits 1-9 must occur exactly once in each column.Each of the digits 1-9

must occur exactly once in each of the 9 3x3 sub-boxes of the grid.The '.' character

indicates empty cells.

Example 1:

Input: board =

[["5","3",".",".","7",".",".",".","."],

["6",".",".","1","9","5",".",".","."],[".","9","8",".",".",".",".","6","."],

["8",".",".",".","6",".",".",".","3"],

["4",".",".","8",".","3",".",".","1"],

["7",".",".",".","2",".",".",".","6"],

[".","6",".",".",".",".","2","8","."],

[".",".",".","4","1","9",".",".","5"],

[".",".",".",".","8",".",".","7","9"]]

Output:

[["5","3","4","6","7","8","9","1","2"],

["6","7","2","1","9","5","3","4","8"],

["1","9","8","3","4","2","5","6","7"],

["8","5","9","7","6","1","4","2","3"],

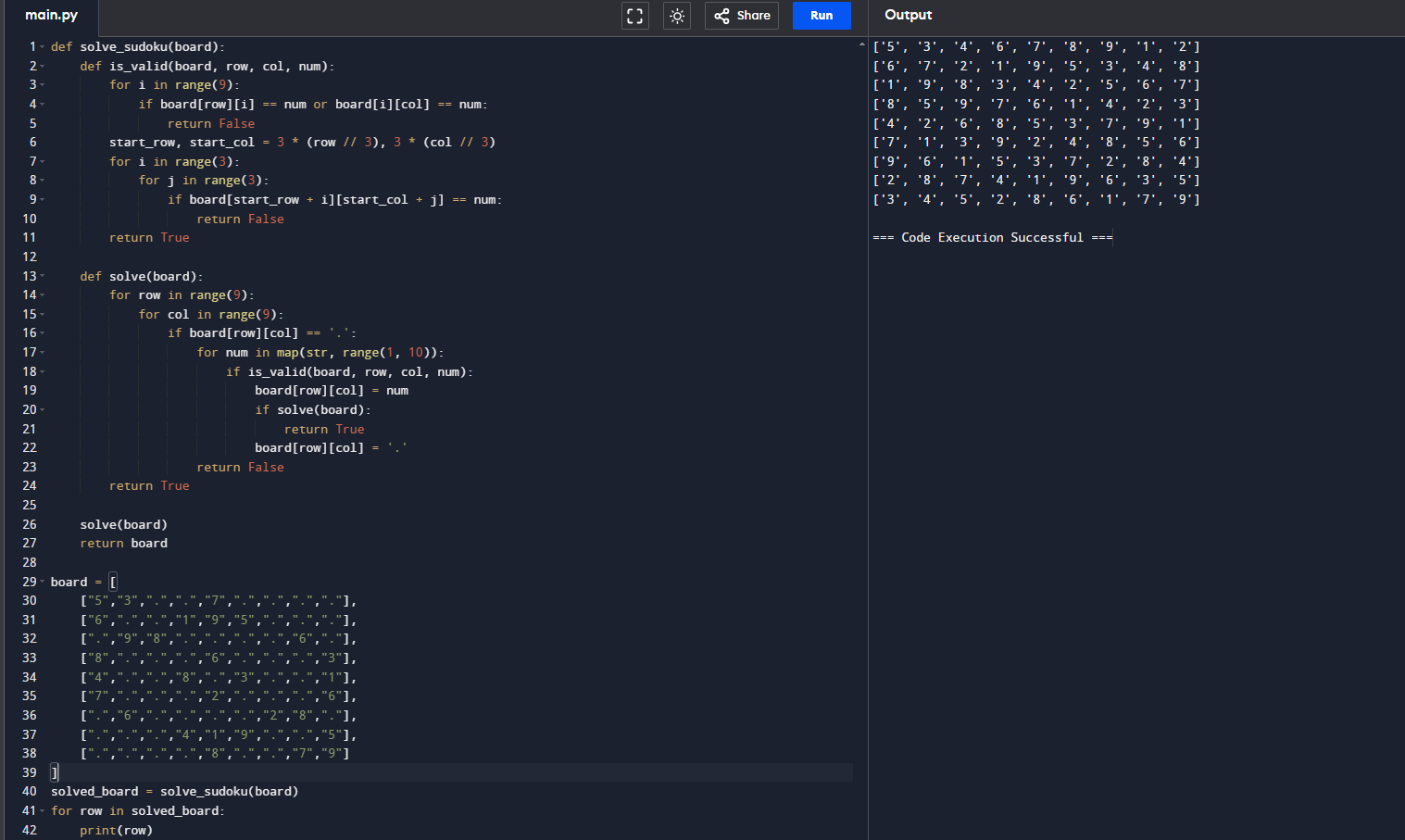
["4","2","6","8","5","3","7","9","1"],

["7","1","3","9","2","4","8","5","6"],

["9","6","1","5","3","7","2","8","4"],

["2","8","7","4","1","9","6","3","5"],

["3","4","5","2","8","6","1","7","9"]]



**4.** Write a program to solve a Sudoku puzzle by filling the empty cells.A sudoku solution must

satisfy all of the following rules:Each of the digits 1-9 must occur exactly once in each

row.Each of the digits 1-9 must occur exactly once in each column.Each of the digits 1-9

must occur exactly once in each of the 9 3x3 sub-boxes of the grid.The '.' character indicates

empty cells.

Example 1:

Input: board =

[["5","3",".",".","7",".",".",".","."],

["6",".",".","1","9","5",".",".","."],

[".","9","8",".",".",".",".","6","."],

["8",".",".",".","6",".",".",".","3"],

["4",".",".","8",".","3",".",".","1"],

["7",".",".",".","2",".",".",".","6"],

[".","6",".",".",".",".","2","8","."],

[".",".",".","4","1","9",".",".","5"],

[".",".",".",".","8",".",".","7","9"]]

Output:

[["5","3","4","6","7","8","9","1","2"],

["6","7","2","1","9","5","3","4","8"],

["1","9","8","3","4","2","5","6","7"],

["8","5","9","7","6","1","4","2","3"],

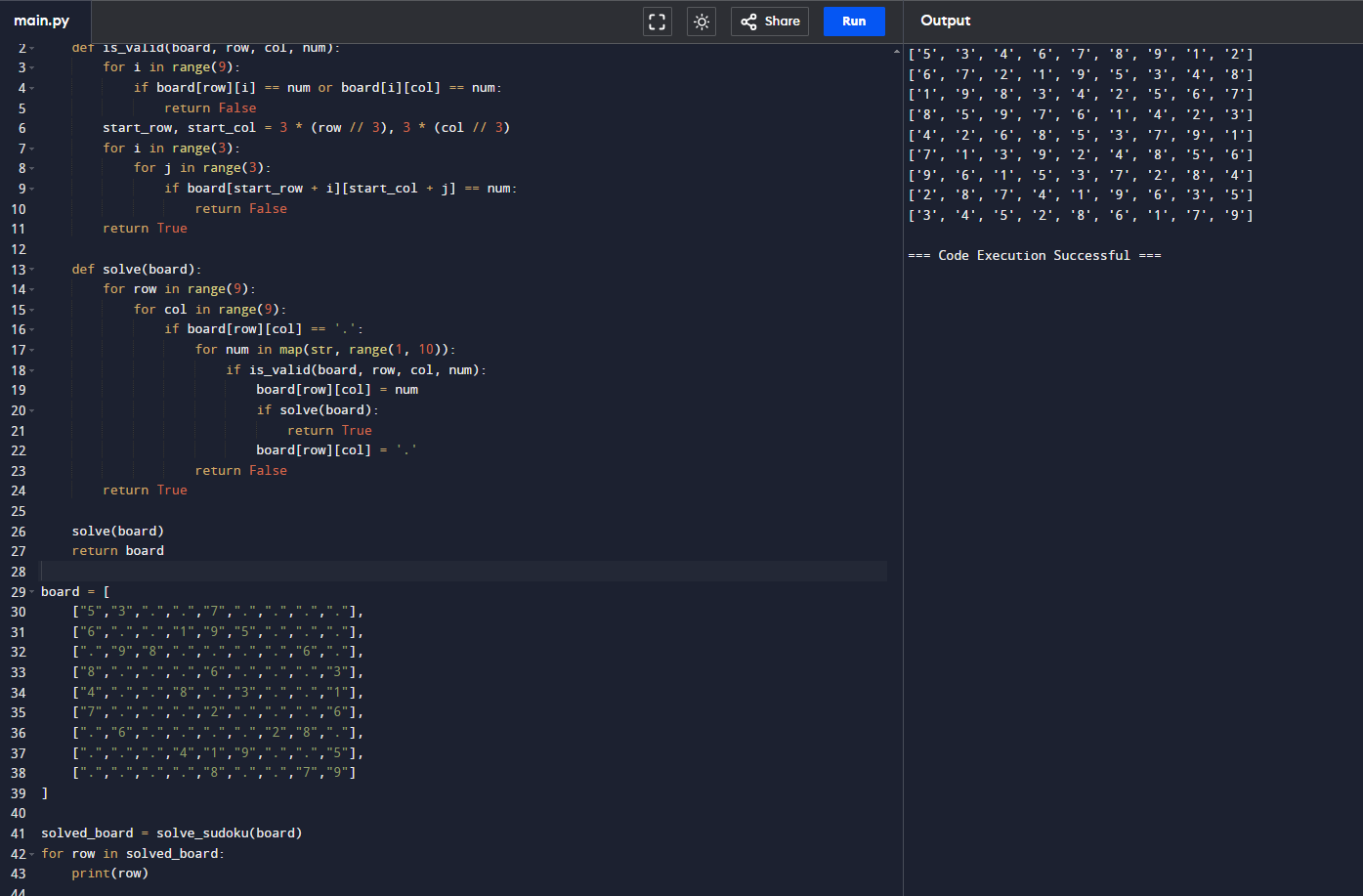
["4","2","6","8","5","3","7","9","1"],

["7","1","3","9","2","4","8","5","6"],

["9","6","1","5","3","7","2","8","4"],

["2","8","7","4","1","9","6","3","5"],

["3","4","5","2","8","6","1","7","9"]]



**5.** You are given an integer array nums and an integer target. You want to build an expression

out of nums by adding one of the symbols '+' and '-' before each integer in nums and then concatenate all the integers.For example, if nums = [2, 1], you can add a '+' before 2 and a '-'

before 1 and concatenate them to build the expression "+2-1" Return the number of different

expressions that you can build, which evaluates to target.

Example 1:

Input: nums = [1,1,1,1,1], target = 3

Output: 5

Explanation: There are 5 ways to assign symbols to make the sum of nums be target 3.

-1 + 1 + 1 + 1 + 1 = 3

+1 - 1 + 1 + 1 + 1 = 3

+1 + 1 - 1 + 1 + 1 = 3

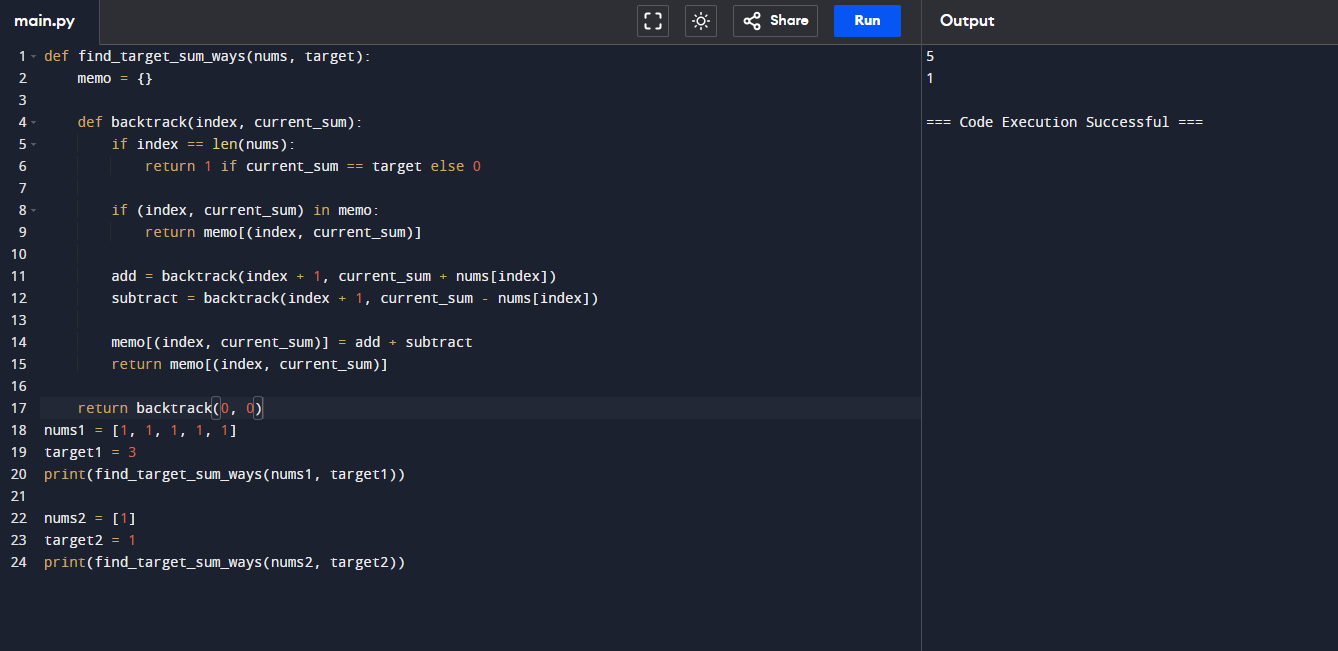
+1 + 1 + 1 - 1 + 1 = 3

+1 + 1 + 1 + 1 - 1 = 3

Example 2:

Input: nums = [1], target = 1

Output: 1



**6**. Given an array of integers arr, find the sum of min(b), where b ranges over every

(contiguous) subarray of arr. Since the answer may be large, return the answer modulo 109 + 7.

Example 1:

Input: arr = [3,1,2,4]

Output: 17

Explanation:

Subarrays are [3], [1], [2], [4], [3,1], [1,2], [2,4], [3,1,2], [1,2,4], [3,1,2,4].

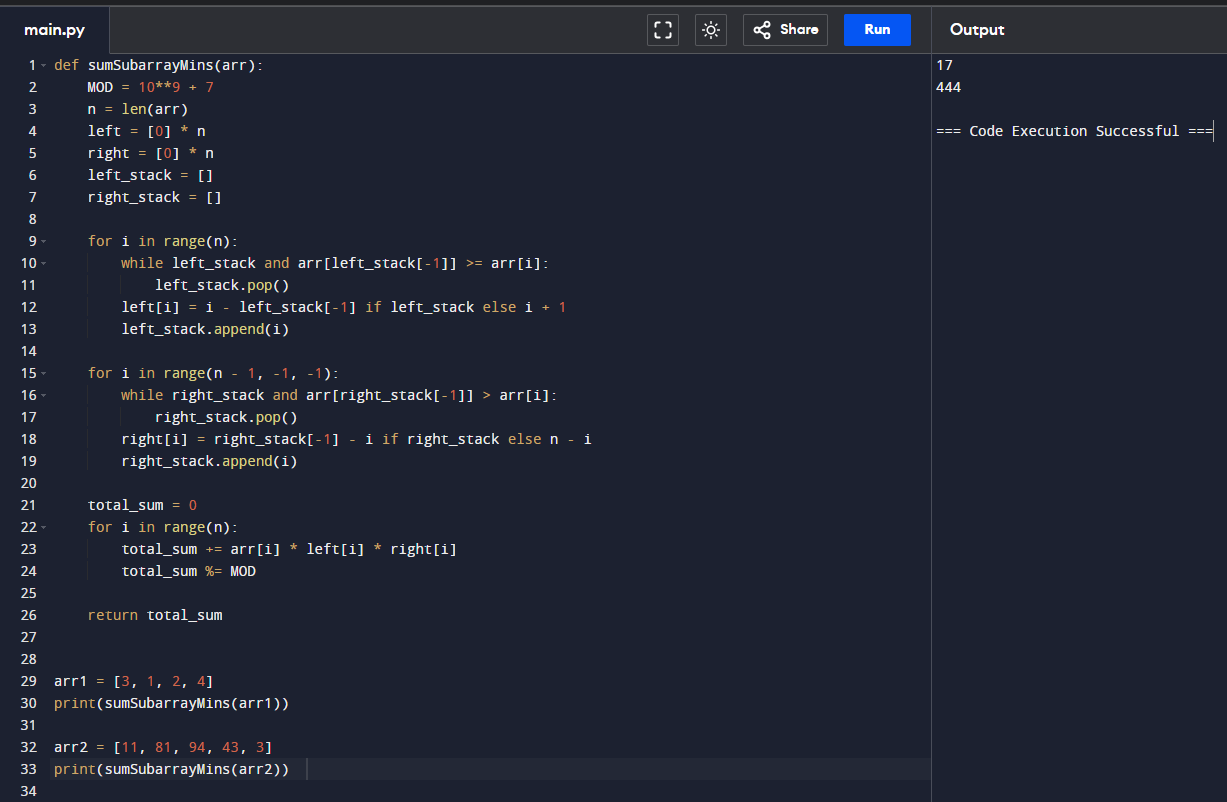
Minimums are 3, 1, 2, 4, 1, 1, 2, 1, 1, 1.

Sum is 17.

Example 2:

Input: arr = [11,81,94,43,3]

Output: 444



**7.** Given an array of distinct integers candidates and a target integer target, return a list of all

unique combinations of candidates where the chosen numbers sum to target. You may return

the combinations in any order.The same number may be chosen from candidates an unlimited

number of times. Two combinations are unique if the frequency of at least one of the chosen

numbers is different.The test cases are generated such that the number of unique

combinations that sum up to target is less than 150 combinations for the given input.

Example 1:

Input: candidates = [2,3,6,7], target = 7

Output: [[2,2,3],[7]]

Explanation:

2 and 3 are candidates, and 2 + 2 + 3 = 7. Note that 2 can be used multiple times.

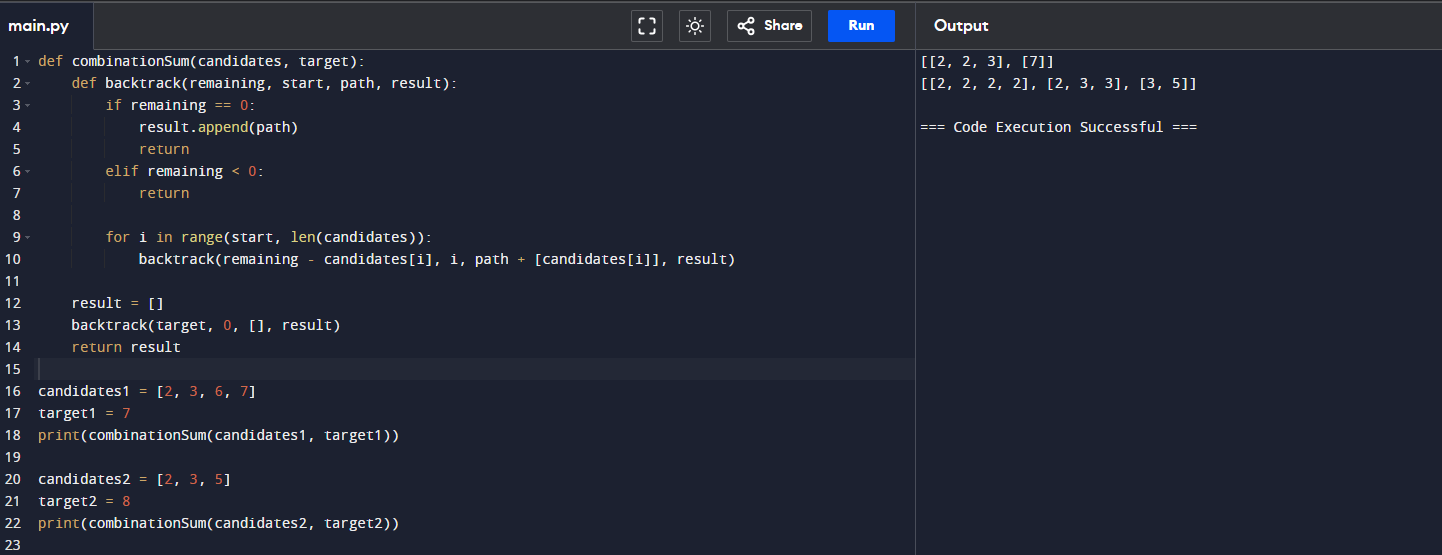
7 is a candidate, and 7 = 7.

These are the only two combinations.

Example 2:

Input: candidates = [2,3,5], target = 8

Output: [[2,2,2,2],[2,3,3],[3,5]]4. COMBINATION SUM 2:



**8**. Given a collection of candidate numbers (candidates) and a target number (target), find all

unique combinations in candidates where the candidate numbers sum to target. Each number

in candidates may only be used once in the combination. The solution set must not contain

duplicate combinations.

Example 1:

Input: candidates = [10,1,2,7,6,1,5], target = 8

Output:

[

[1,1,6],

[1,2,5],

[1,7],

[2,6]

]

Example 2:

Input: candidates = [2,5,2,1,2], target = 5

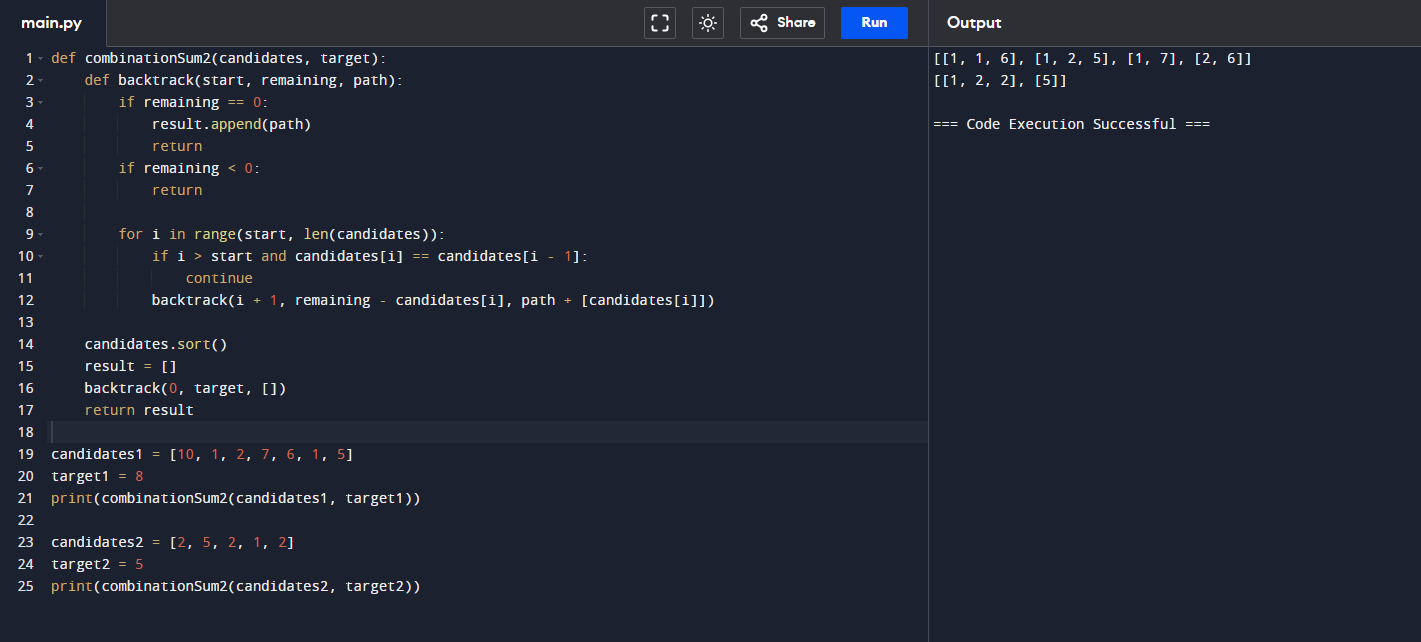
Output:

[

[1,2,2],

[5]

]



**9.** Given an array nums of distinct integers, return all the possible permutations. You can return

the answer in any order.

Example 1:

Input: nums = [1,2,3]

Output: [[1,2,3],[1,3,2],[2,1,3],[2,3,1],[3,1,2],[3,2,1]]

Example 2:

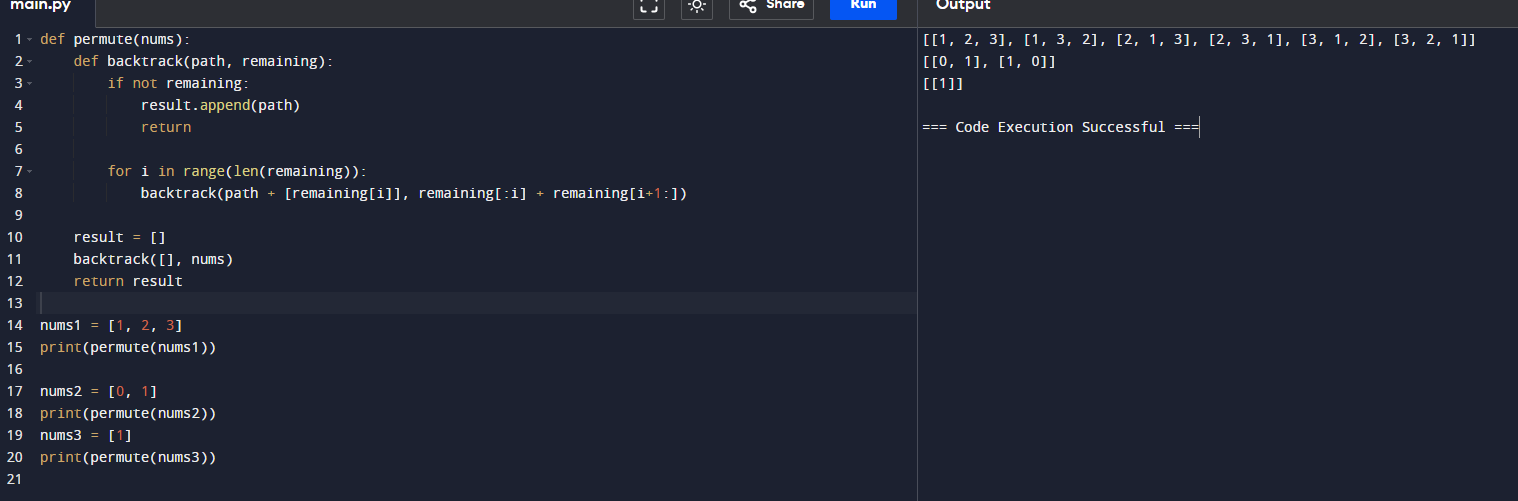
Input: nums = [0,1]

Output: [[0,1],[1,0]]

Example 3:

Input: nums = [1]

Output: [[1]]



**10.** Given a collection of numbers, nums, that might contain duplicates, return all possible unique

permutations in any order.

Example 1:

Input: nums = [1,1,2]

Output:

[[1,1,2],

[1,2,1],

[2,1,1]]

Example 2:

Input: nums = [1,2,3]

Output: [[1,2,3],[1,3,2],[2,1,3],[2,3,1],[3,1,2],[3,2,1]]

