CODE:

import random

import math

def create\_board(n):

return [random.randint(0, n - 1) for \_ in range(n)]

def calculate\_conflicts(board):

n = len(board)

conflicts = 0

for i in range(n):

for j in range(i + 1, n):

if board[i] == board[j] or abs(board[i] - board[j]) == j - i:

conflicts += 1

return conflicts

def get\_neighbors(board):

n = len(board)

neighbors = []

for i in range(n):

for j in range(n):

if board[i] != j:

neighbor = list(board)

neighbor[i] = j

neighbors.append(neighbor)

return neighbors

def simulated\_annealing(n, initial\_temperature, cooling\_rate):

current\_board = create\_board(n)

current\_conflicts = calculate\_conflicts(current\_board)

best\_board = list(current\_board) # Initialize best\_board

best\_conflicts = current\_conflicts

temperature = initial\_temperature

while temperature > 1:

neighbors = get\_neighbors(current\_board)

neighbor = random.choice(neighbors)

neighbor\_conflicts = calculate\_conflicts(neighbor)

delta = neighbor\_conflicts - current\_conflicts

if delta < 0 or random.random() < math.exp(-delta / temperature):

current\_board = neighbor

current\_conflicts = neighbor\_conflicts

if current\_conflicts < best\_conflicts:

best\_board = list(current\_board)

best\_conflicts = current\_conflicts

temperature \*= cooling\_rate

return best\_board, best\_conflicts

n = 4

initial\_temperature = 100

cooling\_rate = 0.99

solution, conflicts = simulated\_annealing(n, initial\_temperature, cooling\_rate)

print("Solution:")

for i in range(n):

line = ""

for j in range(n):

if j == solution[i]:

line += "Q "

else:

line += ". "

print(line)

print("\nConflicts:", conflicts)

OUTPUT:

