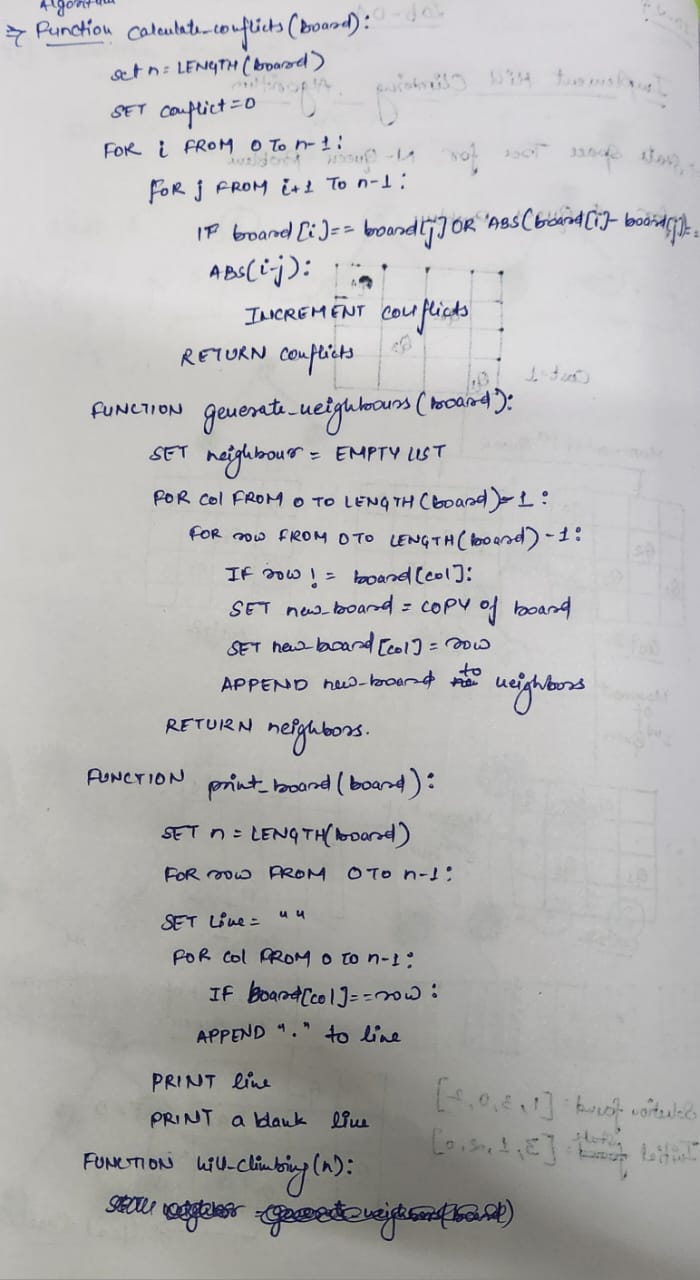
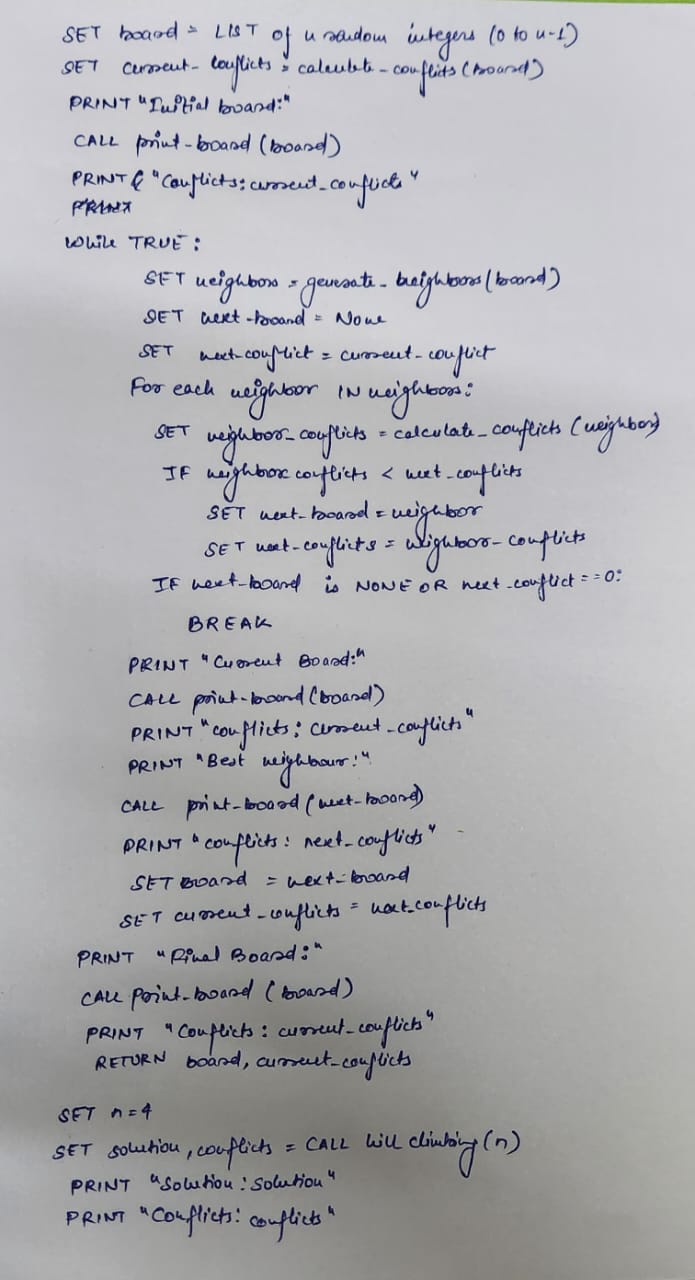
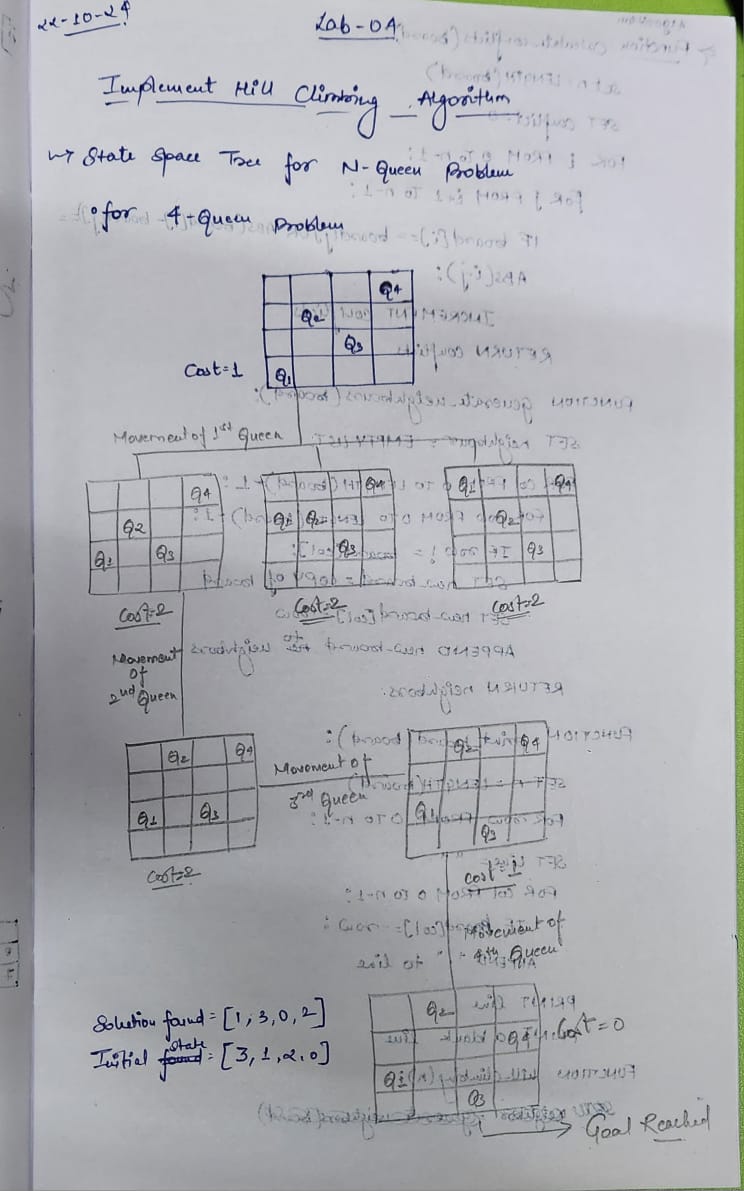
**N-Queens using Hill Climbing**

**Algorithm:**

****

****

**Sate Space Tree:**

****

**Code:**

import random

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]) == abs(i - j):

conflicts += 1

return conflicts

def generate\_neighbors(board):

neighbors = []

for col in range(len(board)):

for row in range(len(board)):

if row != board[col]:

new\_board = board[:]

new\_board[col] = row

neighbors.append(new\_board)

return neighbors

def print\_board(board):

n = len(board)

for row in range(n):

line = ""

for col in range(n):

line += "Q " if board[col] == row else ". "

print(line)

print()

def hill\_climbing(n):

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

current\_conflicts = calculate\_conflicts(board)

print("Initial board:")

print\_board(board)

print(f"Conflicts: {current\_conflicts}")

while True:

neighbors = generate\_neighbors(board)

next\_board = None

next\_conflicts = current\_conflicts

for neighbor in neighbors:

neighbor\_conflicts = calculate\_conflicts(neighbor)

if neighbor\_conflicts < next\_conflicts:

next\_board = neighbor

next\_conflicts = neighbor\_conflicts

if next\_board is None or next\_conflicts == 0:

break

print("Current board:")

print\_board(board)

print(f"Conflicts: {current\_conflicts}")

print("Best neighbor:")

print\_board(next\_board)

print(f"Conflicts: {next\_conflicts}")

board = next\_board

current\_conflicts = next\_conflicts

print("Final board:")

print\_board(board)

print(f"Conflicts: {current\_conflicts}")

return board, current\_conflicts

n = 4

solution, conflicts = hill\_climbing(n)

print("Solution:", solution)

print("Conflicts:", conflicts)

**Output:**

Initial board:

. Q . .

Q . Q .

. . . .

. . . Q

Conflicts: 3

Current board:

. Q . .

Q . Q .

. . . .

. . . Q

Conflicts: 3

Best neighbor:

. Q . .

. . Q .

Q . . .

. . . Q

Conflicts: 1

Final board:

. Q . .

. . Q .

Q . . .

. . . Q

Conflicts: 1

Solution: [2, 0, 1, 3]

Conflicts: 1