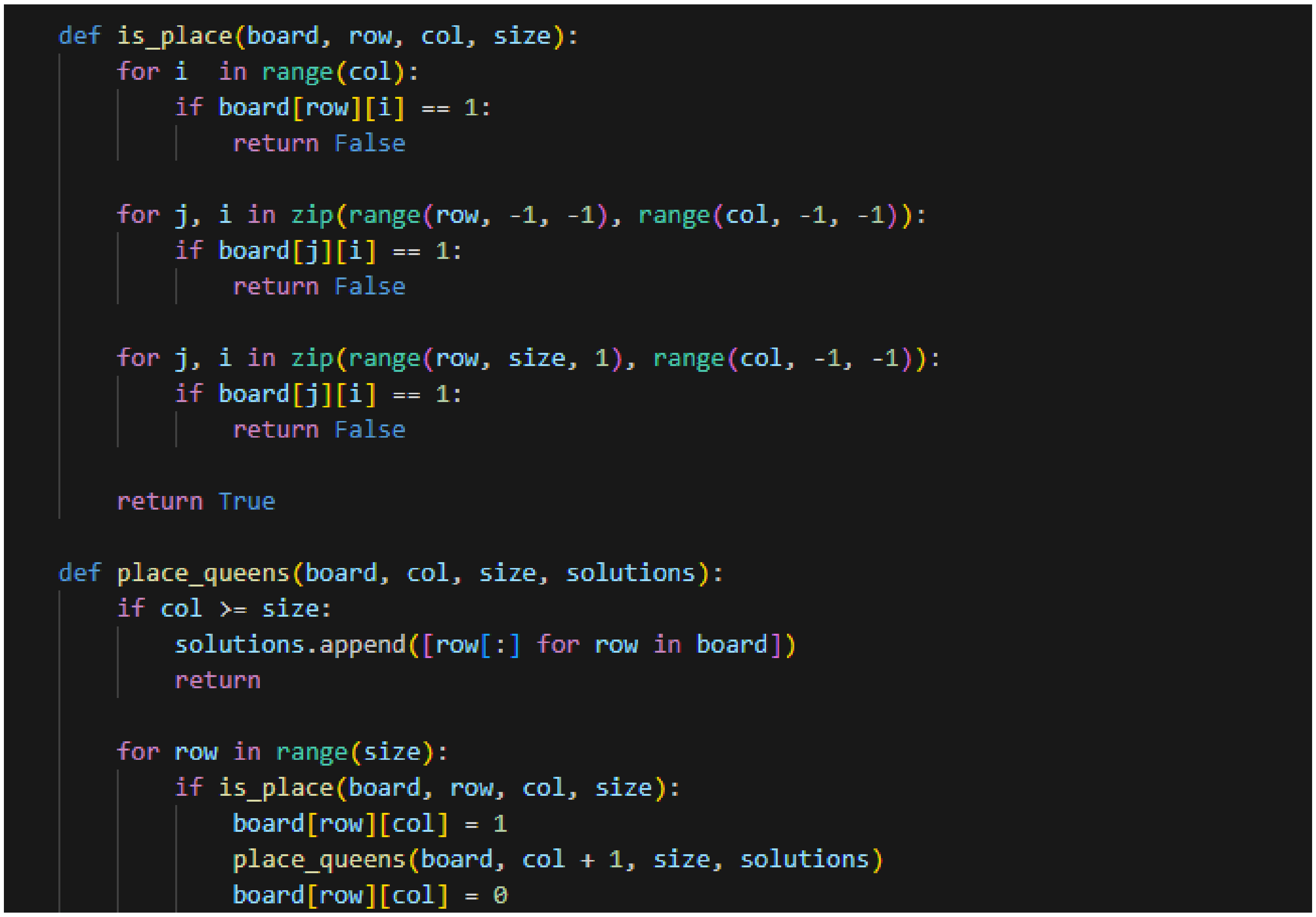
The Superior University, Lahore

Assignment-I (Fall 2024)

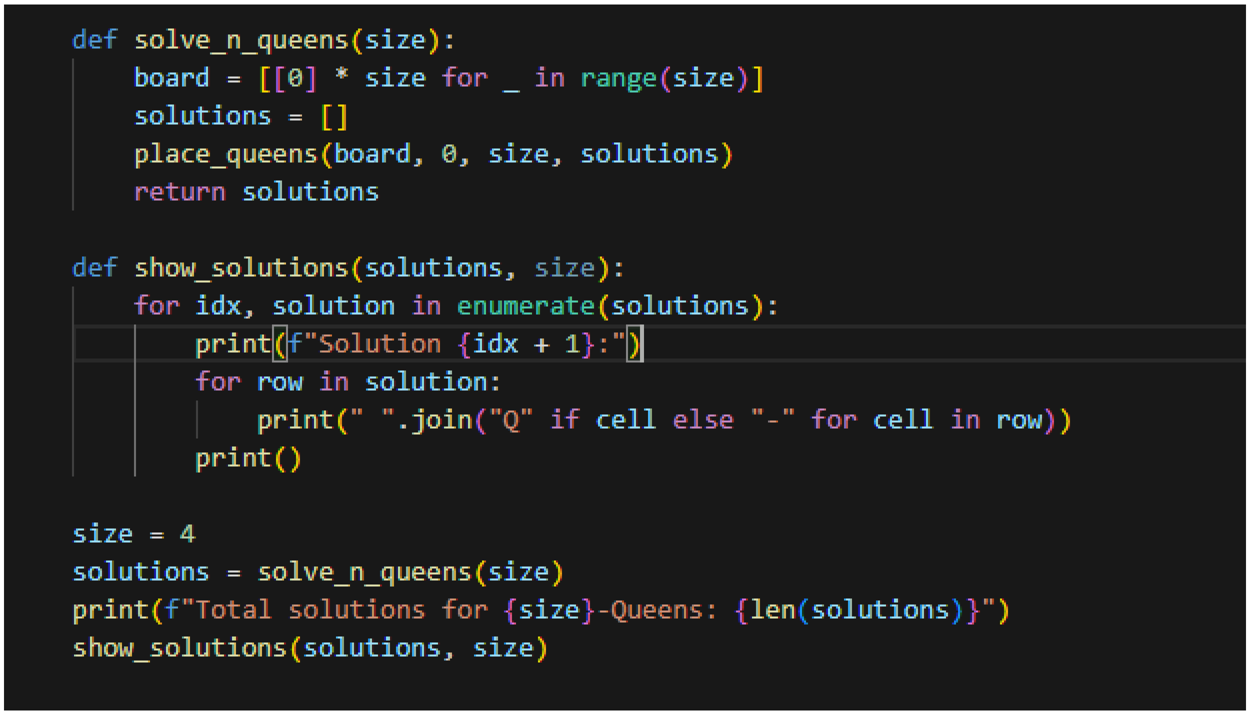
Course Title:	PAI			Course Code:	CPR601260	Credit Hours:	4
Instructor:				Programme Name:	BSDS		
Semester:	4th	Batch:	F23	Section:	BSDSM-4A	Date:	30 January, 2024
Time Allowed:				Maximum Marks:			
Student’s Name:	Fizza Farooq			Reg. No.	Su92-bsdsm-f23-017		
Lab-Task 1: Question 2: Question							

LAB 4, TASK4

Task: N-Queens Problem (Dynamic)



```python
def is_place(board, row, col, size):
    for i  in range(col):
        if board[row][i] == 1:
            return False

    for j, i in zip(range(row, -1, -1), range(col, -1, -1)):
        if board[j][i] == 1:
            return False

    for j, i in zip(range(row, size, 1), range(col, -1, -1)):
        if board[j][i] == 1:
            return False

    return True

def place_queens(board, col, size, solutions):
    if col >= size:
        solutions.append([row[:] for row in board])
        return

    for row in range(size):
        if is_place(board, row, col, size):
            board[row][col] = 1
            place_queens(board, col + 1, size, solutions)
            board[row][col] = 0
```



```python
def solve_n_queens(size):
    board = [[0] * size for _ in range(size)]
    solutions = []
    place_queens(board, 0, size, solutions)
    return solutions

def show_solutions(solutions, size):
    for idx, solution in enumerate(solutions):
        print(f"Solution {idx + 1}:")
        for row in solution:
            print(" ".join("Q" if cell else "-" for cell in row))
        print()

size = 4
solutions = solve_n_queens(size)
print(f"Total solutions for {size}-Queens: {len(solutions)}")
show_solutions(solutions, size)
```

The Python program implements a backtracking solution to solve the N-Queens problem. The solution seeks to put N queens on an NxN chessboard so they do not threaten one another.

1.The function determines the feasibility of queen placement (is_place function) The function examines three conditions before allowing the placement of a queen at (row, col):

- The left section of the chessboard has no queen placed in the same horizontal row.
- The upper-left diagonal does not have a queen placed.
- There is no queen ever located in the lower-left diagonal.

The function returns False when any condition becomes violated. Otherwise, it returns True.

2. The Backtracking algorithm implements the placing queens functionality through the place_queens function.

This operation conducts a queen placement attempt across all rows situated in the active column.Safety of the current position allows the algorithm to both place a queen (1) and advance to the following column.The placement of queens creates a solution since all columns reach the maximum size (col >= size).

Backtracking:

The function will delete (0) the last queen from its previous placement to substitute it with another row while checking for failures.

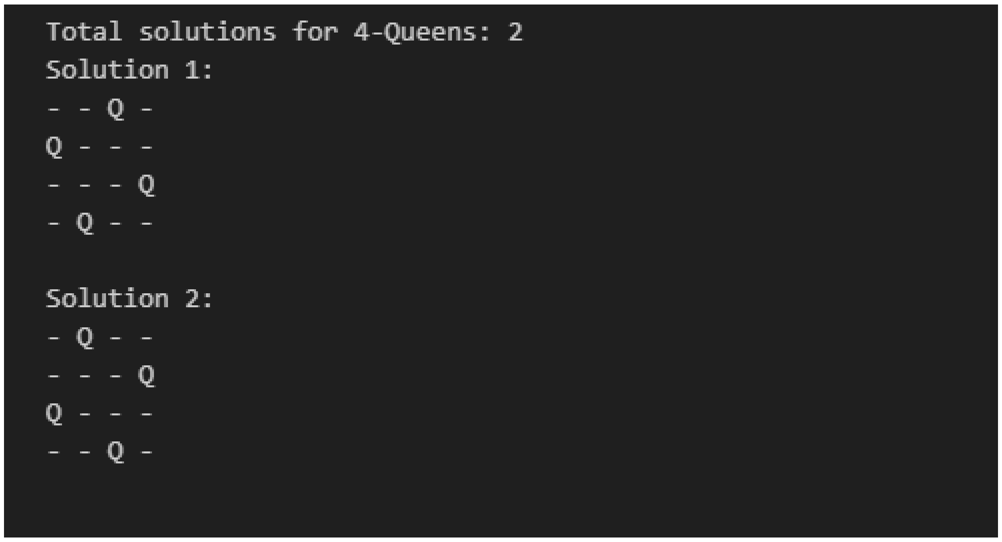
3. The N-Queens Problem solver (solve_n_queens function) allows placing 4 queens into a 4×4 board through two different valid arrangements.

An N × N chessboard receives the initialization of all positions as 0 (empty).

- The place_queens function completes all possible valid solution finding tasks.
- Returns a list of solutions.
- Prints all solutions found.

A queen is shown as Q and an open space is shown using - in the representation.

OUTPUT



```
Total solutions for 4-Queens: 2
Solution 1:
- - Q -
Q - - -
- - - Q
- Q - -

Solution 2:
- Q - -
- - - Q
Q - - -
- - Q -
```

Solves the 4-Queens problem.

Prints the total number of solutions.

Displays each solution.

Output for size = 4