

Lab #01: N-Queens Problem

Place N queens on an $N \times N$ chessboard such that no two queens threaten each other (i.e., no two queens share the same row, column, or diagonal).

Lab #02: 8- puzzle Problem

Solve the eight puzzle problem where,

- **Initial state:** Any configuration
- **Goal State:** Tiles in ascending (specific) order

The solution will be a sequence of moves of the blank that transforms the initial state to the goal state.

Example: 8-puzzle

7	2	4
5		6
8	3	1

Start State

1	2	3
4	5	6
7	8	

Goal State

Lab #03: Water jug problem

You are given two jugs:

- Jug 1: Capacity of X liters ($0 \leq X \leq 9$)
- Jug 2: Capacity of Y liters ($0 \leq Y \leq 9$)

Task: Measure exactly Z liters of water using these two jugs.

Allowed Operations:

1. Fill one of the jugs.
2. Empty one of the jugs.
3. Pour water from one jug into another until one jug is either full or empty.

Solve for: $X = 3$, $Y = 4$, $Z = 2$

Lab #04:

Implement the **Breadth-First Search (BFS)** algorithm and submit the code.

Lab #05:

Implement the **Depth-First Search (DFS)** algorithm and submit the code.

Lab #06:

Solve the **Missionaries and Cannibals Problem** and submit the code. Show the steps as follows:

```
*D:\SUSTV\Artificial Intelligenc  x  +  v
Missionneries and Cannibals problem
-----

M M M C C C  BOAT-L[ , ]-----
M M M C      BOAT-L[C,C]-----
M M M C      -----BOAT-R[C,C]
M M M C      -----BOAT-R[C, ]  C
M M M C      BOAT-L[C, ]-----  C
M M M        BOAT-L[C,C]-----  C
M M M        -----BOAT-R[C,C]  C
M M M        -----BOAT-R[C, ]  C C
M M M        BOAT-L[C, ]-----  C C
M C          BOAT-L[M,M]-----  C C
M C          -----BOAT-R[M,M]  C C
M C          -----BOAT-R[C,M]  M C
M C          BOAT-L[C,M]-----  M C
C C          BOAT-L[M,M]-----  M C
C C          -----BOAT-R[M,M]  M C
C C          -----BOAT-R[C, ]  M M M
C C          BOAT-L[C, ]-----  M M M
C            BOAT-L[C,C]-----  M M M
C            -----BOAT-R[C,C]  M M M
```

Lab #07:

Make a **Sudoku Solver** and submit the code.

5	3	1	2	7	6	8	9	4
6	2	4	1	9	5	2		
	9	8					6	
8				6				3
4			8		3			1
7				2				6
	6					2	8	
			4	1	9			5
				8			7	9