**Title: Simulation of 8-Puzzle Game**

**Problem Statement:**

The 8-puzzle game is a sliding tile puzzle consisting of a 3x3 grid with numbered tiles ranging from 1 to 8, along with one empty space. The objective is to move the tiles to reach a goal configuration by sliding them horizontally or vertically into the empty space. This problem can be generalized as an artificial intelligence challenge, requiring the implementation of search algorithms to efficiently find the shortest solution.

**Algorithm:**

* **Initial State**: A randomized 3x3 grid with 8 numbered tiles and one empty space.
* **Search Algorithms**:
  1. **Breadth-First Search (BFS)**:
     + Explores all nodes at the present depth level before moving on to the next depth level.
     + Guarantees the shortest path but may consume more memory.
  2. *A Algorithm*\*:
     + Utilizes both path cost (g(n)) and a heuristic estimate (h(n)) to guide the search.
     + Common heuristics:
       - **Misplaced Tile Heuristic**: Counts the number of misplaced tiles.
       - **Manhattan Distance**: Calculates the total distance tiles are from their goal positions.
* **Goal State**:

1 2 3

4 5 6

7 8 \_

* **Algorithm Flow**:
  1. Initialize the starting state.
  2. Use the search algorithm to explore possible moves.
  3. For each move, calculate the new state and evaluate the path cost and heuristic value.
  4. Continue until the goal state is reached.
  5. Return the sequence of moves and the total cost.

**Dataset Authentication:**

* **Dataset**: The dataset for the simulation is the set of all possible configurations of the 8-puzzle game. The algorithm will authenticate the dataset by ensuring:
  + **Solvability**: Before running the simulation, the program checks if the given puzzle configuration is solvable. An 8-puzzle is solvable if the number of inversions is even.
  + **Initial and Goal States**: The algorithm verifies that the initial state is valid and that it can lead to the predefined goal state.

**Expected Output:**

* **Optimal Solution Path**: The simulation will produce the shortest path of moves (left, right, up, down) to solve the 8-puzzle from the initial state to the goal state.
* **Solution Cost**: The number of moves taken to reach the goal.
* **Performance Metrics**:
  + Total number of nodes expanded.
  + Time taken to find the solution.
  + Memory used during the search process.

**Example Output:**

* *Input:* Start state:

1 2 3

4 8 5

7 \_ 6

* *Output:*
  + Moves: Right, Down, Left, Up
  + Number of Moves: 4
  + Time taken: X seconds
  + Space complexity: O(X)

**Batch-17**

**2320030182-Sadhana**

**2320030185-Avinash**

**2320030191-Jayakrishna**