---

**How to Solve the 8-Puzzle Using IDS (Iterative Deepening Search)**

**Step 1: Understand the Problem**

The 8-Puzzle consists of a 3x3 grid with 8 numbered tiles and one empty space (0).

- **Initial State**: The starting arrangement of the tiles.
- **Goal State**: Usually the tiles arranged in order with the empty space at the bottom-right:

1 2 3

4 5 6

7 8 0

The empty space (0) can move **Up, Down, Left, or Right** to swap positions with adjacent tiles.

---

**Step 2: Understand IDS**

**Iterative Deepening Search (IDS)** combines **Depth-First Search (DFS)** and **Breadth-First Search (BFS)**:

1. Start with **depth = 0**.
2. Perform DFS limited to the current depth.
3. If goal is not found, **increase depth by 1** and repeat.
4. Continue until the goal state is reached.

**Advantages**:

- Explores all possible moves systematically.
- Avoids using excessive memory like BFS.
- Guaranteed to find the shortest path.

---

**Step 3: Implementing IDS for 8-Puzzle**

1. **Start at Initial State**:

2. (1, 2, 3
3. ,4, 5 ,0
4. ,6 ,7 ,8)
5. **Set depth = 0** → Check if initial state is goal → No → Increase depth.
6. **Depth = 1** → Generate all possible moves of the empty space (0):
    - Move Up
    - Move Left
    - Move Down
7. **Depth = 2** → For each new state from depth 1, generate all possible moves **without revisiting already seen states**.
8. **Continue increasing depth** and exploring nodes in DFS manner until the **goal state** is reached:

    (1, 2, 3

    ,4 ,5, 6

    ,7, 8, 0)

---

**Step 4: Avoiding Repetition**

- Keep a **visited set** to avoid revisiting the same state.
- This reduces unnecessary calculations and speeds up finding the solution.

---

**Step 5: Track Moves**

- Record each move of the empty space.
- Example sequence to solve a puzzle (depending on initial state):

Initial : ( 1, 2, 3

,4 ,5, 0

,6 ,7, 8)

Move Left →

( 1, 2, 3

,4, 0, 5

,6 ,7 ,8)

Move Down →

(1 2 3

4 5 6

7 8 0)

---

**Step 6: Summary**

1. Use IDS to explore depth-limited DFS iteratively.
2. At each depth, generate all possible moves of the empty tile.
3. Avoid revisiting repeated states.
4. Increase depth until the goal is found.
5. Track the moves to reconstruct the solution path.

---