

**Department of Computer Engineering**

**Course: DAA**

**Course Code: BTECCE21501**

**Mini-Project – Report**

**PHASE - III**

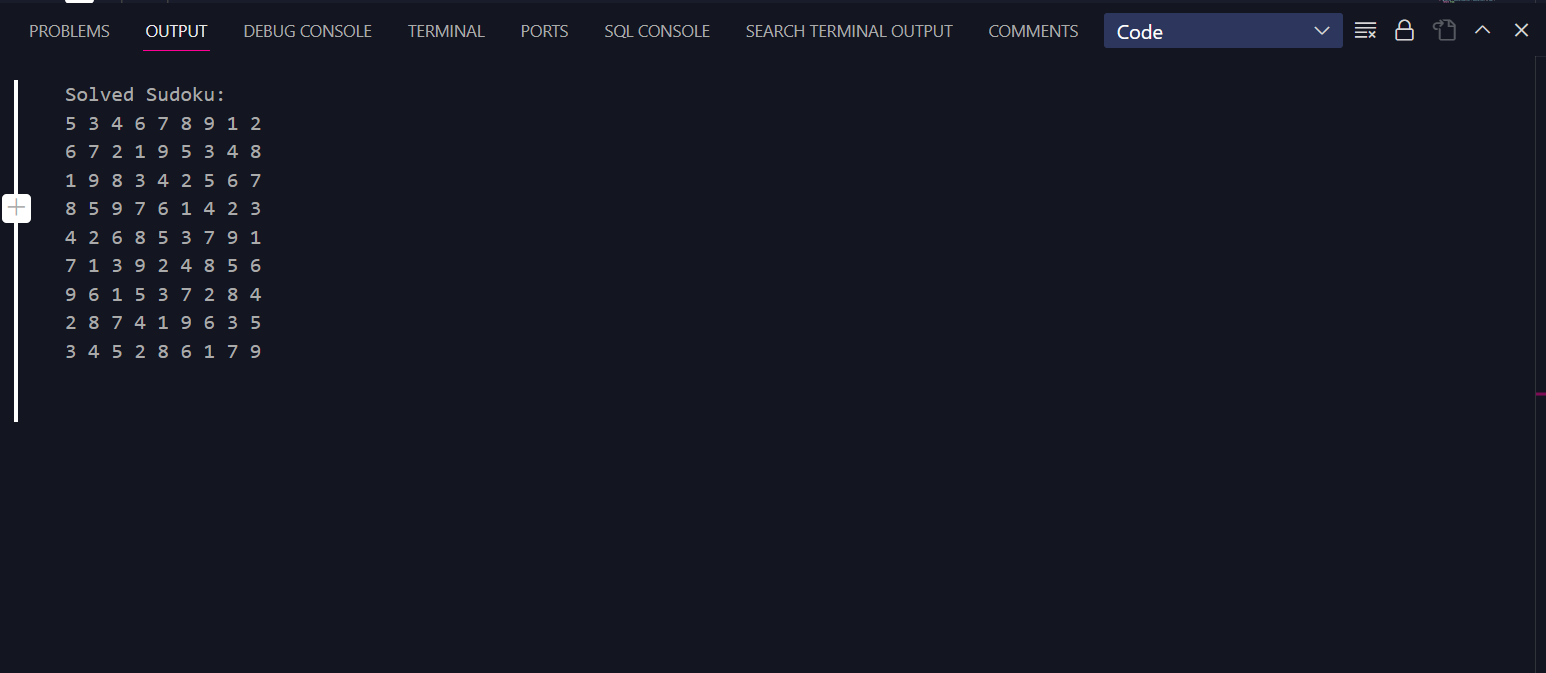
**Guidance By – Prof. Noshir Tarapore**

**Topic: Algorithmic Solutions for Solving Sudoku Puzzles**

**By**

|  |  |  |
| --- | --- | --- |
| **Roll No** | **SRN** | **Name of Student** |
| 38 | 202200930 | Shyamal S. Patil |
| 39 | 202200931 | Shaikh M. Asad |
| 33 | 202101389 | Tanmay Gawali |

**Output:**

****

The detailed outcome of the backtracking algorithm for solving a Sudoku puzzle can be summarized as follows:

1. **Initialization:**
   * The algorithm starts with a partially filled Sudoku grid.
   * It begins the search for an empty cell (a cell with a value of 0) starting from the top-left corner of the grid.
2. **Finding an Empty Cell:**
   * The algorithm searches the grid for the first empty cell.
   * If an empty cell is found, the algorithm proceeds to the next step. If no empty cells are found, the puzzle is considered solved, and the algorithm returns true.
3. **Trying Numbers 1 to 9:**
   * For the current empty cell, the algorithm iterates through numbers from 1 to 9.
   * For each number, it checks if it's a valid move for the empty cell (i.e., it's not already in the same row, column, or 3x3 box).
   * If a valid number is found, it is placed in the empty cell, and the algorithm moves on to the next empty cell (recursively).
4. **Recursion:**
   * The algorithm proceeds with the next empty cell (either the next cell in the same row or the next row).
   * The process of trying numbers is repeated for the new empty cell.
   * This recursive process continues until a solution is found or until the algorithm determines that there is no valid number to place in an empty cell (i.e., backtracking is required).
5. **Backtracking:**
   * If the algorithm reaches an empty cell where no valid number can be placed (all numbers from 1 to 9 have been tried), it returns false to indicate that the current path does not lead to a solution.
   * This triggers backtracking to the previous empty cell, where a different number is tried.
   * If all possible numbers have been tried for a particular empty cell and none lead to a solution, the algorithm continues to backtrack further, revisiting and changing numbers in previous cells until a valid solution is found or all possibilities are exhausted.
6. **Solution Found:**
   * When a valid number is found for each empty cell, and all constraints are satisfied, the algorithm completes and returns true to indicate that a solution has been found.
7. **No Solution Exists:**
   * If the algorithm exhaustively explores all possibilities without finding a valid solution, it returns false to indicate that no solution exists for the given Sudoku puzzle.

The algorithm uses recursive depth-first search and backtracking to explore the solution space of the Sudoku puzzle. It repeatedly tries different numbers in empty cells, backtracks when necessary, and continues until a solution is found or it concludes that no valid solution is possible. This process continues until the entire grid is filled with valid numbers or all possibilities have been exhausted.