DFS WITHOUT HEURISTIC

```
def dfs(start_state, goal_state):
    stack = [start_state]
    visited = set()
    parent = {start_state: None}

    while stack:
        current = stack.pop()
        if current == goal_state:
            path = []
            while current:
                path.append(current)
                current = parent[current]
            return path[::-1]

        if current not in visited:
            visited.add(current)
            # Get neighbors and reverse to explore in a consistent
order (e.g., right, down, left, up)
            neighbors = get_neighbors_dfs(current)
            neighbors.reverse()
            for neighbor in neighbors:
                if neighbor not in visited:
                    parent[neighbor] = current
                    stack.append(neighbor)
    return None

# Get input from the user row by row
print("1BM23CS333")
print("Enter the initial state (enter 3 digits per row, separated by
spaces, 0 for empty):")
initial_state_rows = []
for i in range(3):
    row = input(f"Row {i+1}: ").split()
    initial_state_rows.extend(row)
initial_state = "".join(initial_state_rows)

print("\nEnter the goal state (enter 3 digits per row, separated by
spaces, 0 for empty):")
goal_state_rows = []
for i in range(3):
    row = input(f"Row {i+1}: ").split()
    goal_state_rows.extend(row)
goal_state = "".join(goal_state_rows)

solution = dfs(initial_state, goal_state)

if solution:
```

```
    print("\nDFS solution path:")
    for s in solution:
        print(s[:3])
        print(s[3:6])
        print(s[6:])
        print()
else:
    print("\nNo solution found.")
```

OUTPUT:

```
Streaming output truncated to the last 5000 lines.
164
320
785

160
324
785

106
324
785

126
304
785

126
384
705

126
384
075

126
084
375

026
184
375

206
184
375

286
104
375

286
174
305
```