Lab Report:-06

Experiment Name: 8-Puzzle Problem using Breadth-First Search (BFS)

Objectives:

To comprehend the operation of the Breadth-First Search (BFS) method in state-space issue resolution.
Using BFS to solve the traditional 8-Puzzle problem.
To get familiar with Python programming principles such as condition checking, deep copying, loops, and lists.
To see the tiles moving step-by-step and the route leading to the desired state.

Problem Statement:

The 8-Puzzle problem is a 3x3 grid with one blank area (0) and tiles numbered 1 through 8. The objective is to slide tiles into the vacant area to change an original tile arrangement into the desired target configuration. In order to find the answer in the fewest number of movements, the BFS algorithm investigates every move that might be made, level by level. BFS is used in this experiment to methodically investigate every stage and determine the best move sequence to accomplish the objective.

Algorithm:

Step 1: Initialize the queue with the initial state and mark it as visited.

Step 2: Loop until the queue is empty or the goal state is found:

(1) Take the front state from the queue.
(2) Find the position of the empty tile (0).
(3) Generate all valid moves (up, down, left, right) by swapping the empty tile with adjacent tiles.

Step 3: Check each new state:

(1) If it is already visited, ignore it.
(2) If it matches the goal state, print the solution and stop.
(3) Otherwise, append it to the queue and mark as visited.

Step 4: Continue until the goal state is reached or no new states are left.

Example Execution:

Output:

```
Onik-Howlader@DESKTOP-I4SJT3P MINGW64 /d/Riadul sir assingment
$ python -u "/d/Riadul sir assingment/8-puzzel/8-puzzel.py"
[[1, 2, 3], [4, 5, 6], [0, 7, 8]]
BFS START:
[[1, 2, 3], [4, 5, 6], [7, 8, 0]]
Solution Found:
[[1, 2, 3], [4, 5, 0], [7, 8, 6]]
[[1, 2, 3], [4, 5, 6], [7, 0, 8]]
New State:
[[1, 2, 3], [4, 5, 0], [7, 8, 6]]
Solution Found:
[[1, 2, 3], [4, 5, 6], [7, 8, 0]]
[[1, 2, 0], [4, 5, 3], [7, 8, 6]]
[[1, 2, 3], [4, 0, 5], [7, 8, 6]]
New State:
[[1, 2, 3], [4, 5, 6], [7, 0, 8]]
Solution Found:
[[1, 2, 3], [4, 0, 6], [7, 5, 8]]
[[1, 2, 3], [4, 5, 6], [7, 8, 0]]
[[1, 2, 3], [4, 5, 6], [0, 7, 8]]
[[1, 2, 3], [4, 5, 6], [0, 7, 8]]
result Found
```

Conclusion:

1) The 8-Puzzle problem's state-space is methodically investigated by BFS.
2) If there is a solution, it ensures that the shortest route to the desired state will be found.
3) Tile movements are accurately replicated by using deep copy and Python's list operations.
4) This experiment improves knowledge of state-space traversal, backtracking avoidance, and search methods.