EXPERIMENT 5

PROBLEM TITLE

Solve the 8-Puzzle Problem Using Best-First Search.

OBJECTIVE OF THE PROGRAM

To implement the Best-First Search algorithm to solve the 8-puzzle problem and understand heuristic-driven search strategies.

DESCRIPTION

The 8-puzzle is a classic problem in AI, where the goal is to move tiles on a 3x3 grid to achieve a specific configuration. Best-First Search selects the most promising node based on a heuristic function (h(n)), exploring nodes with the lowest heuristic cost first.

ALGORITHM

1. Initialize the open list with the start node.
2. Loop until the open list is empty:
    a. Select the node with the lowest heuristic value.
    b. If this node is the goal, reconstruct the path and return it.
    c. Otherwise, generate the node's neighbors by sliding tiles.
    d. Add the neighbors to the open list based on their heuristic values.
3. If the open list is empty and no solution is found, return failure.

EXPERIMENT 6

PROBLEM TITLE

Implementation of the Travelling Salesman Problem (TSP).

OBJECTIVE OF THE PROGRAM

To implement a solution to the Travelling Salesman Problem (TSP) using a simple greedy algorithm and understand the challenges of combinatorial optimization.

DESCRIPTION

The Travelling Salesman Problem (TSP) is a classic optimization problem where the goal is to find the shortest possible route that visits each city exactly once and returns to the origin city. It is NP-hard, meaning there is no efficient solution for large instances, but heuristic or approximation algorithms can provide good solutions.

ALGORITHM

1. Start at any city as the current city.
2. Find the nearest unvisited city and move to it.
3. Repeat until all cities are visited.
4. Return to the starting city to complete the tour.

EXPERIMENT 7

PROBLEM TITLE

Write a Program to Implement the Towers of Hanoi Problem.

OBJECTIVE OF THE PROGRAM

To implement a recursive solution for the Towers of Hanoi problem, illustrating the principles of recursion in problem-solving.

DESCRIPTION

The Towers of Hanoi is a classic problem where three pegs and a number of disks are involved. The goal is to move all the disks from the source peg to the destination peg following these rules:

- Only one disk can be moved at a time.
- A disk can only be placed on top of a larger disk.
- All disks must end up on the destination peg in the same order.

ALGORITHM

1. Move n-1 disks from the source peg to the auxiliary peg.
2. Move the nth disk directly to the destination peg.
3. Move the n-1 disks from the auxiliary peg to the destination peg.