

Intelligent Systems

Project 1

Name	Student Id
Nilanjan Mhatre	801045013
Akash	801043608

8-puzzle Formulation

The 8-puzzle is solved by using heuristic function and generating a tree by expanding the nodes having the least heuristic.

Manhattan distance or Misplaced tiles count can be used as a heuristic.

The A* algorithm aims at using the heuristic as well as the distance travelled so far, combined to make a selection of the nodes from the fringe.

Program Structure

```
*Puzzle8.java TestHashMethod.java AStar.java
7 public class Puzzle8 {
8     private List<Integer> a;
9     private Integer heuristicValue;
10    private Integer depth;
11    private Puzzle8 parentPuzzle;
12
13    public Puzzle8(List<Integer> a) {}
14
15    public Puzzle8(List<Integer> a, Integer depth) {}
16
17    // Operations up, down, left, right
18    * Move the blank space or 0 up, if possible
19    public Puzzle8 moveUp() {
20        Puzzle8 p = null;
21        int index = a.indexOf(0);
22        if(index > 2) {
23            List<Integer> b = new ArrayList<>(a);
24            Collections.swap(b, index, index - 3);
25            p = new Puzzle8(b);
26            p.setParentPuzzle(this);
27        }
28        return p;
29    }
30
31    * Move the blank space or 0 down, if possible
32    public Puzzle8 moveDown() {}
33
34    * Move the blank space or 0 to the right, if possible
35    public Puzzle8 moveRight() {}
36
37    * Move the blank space or 0 to the left, if possible
38    public Puzzle8 moveLeft() {}
39
40    //..... display the puzzle .....//
41    public String display() {}
42    //.....//
43
44    // Getters and Setters
45    /**
46     * The puzzle array stored as a 1D - array
47     * @return List<Integer>
48     */
49    public List<Integer> getA() {}
50
51    public void setA(List<Integer> a) {}
52
53    * The Heuristic value as per the applied heuristic function
54    public Integer getHeuristicValue() {}
55
56    public void setHeuristicValue(Integer heuristicValue) {}
57
58    * The distance travelled to get to this puzzle/node, g()
59    public Integer getDepth() {}
60
61    public void setDepth(Integer depth) {}
62
63    * Parent puzzle will help to determine the path
64    public Puzzle8 getParentPuzzle() {}
65
66    public void setParentPuzzle(Puzzle8 parentPuzzle) {}
67
68    /**
69     * The total heuristic value
70     * @return Integer g() + f(), total heuristic value
71     */
72    public Integer getTotalHeuristicValue() {}
73
74    // Equals and hashCode
75    @Override
76    public int hashCode() {
77        String str = "";
78        for(int i=0; i<a.size(); i++) {
79            str += a.get(i);
80        }
81
82        return new Integer(str);
83    }
84
85    public boolean equals(Object obj) {}
86
87
88
89
90
91
92
93
```

1. An **instance** of class 'Puzzle8' stores a particular **state**, along with its heuristic value (h), the distance/depth travelled (g) to get to that state and the parent 'Puzzle8'.
2. The array 'a' defines the number arrangement as a 1D array.
e.g. a= {1, 2, 3, 4, 5, 6, 7, 8, 0}
implies $\begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & - \end{bmatrix}$
3. '**hashCode**' method returns '**hash**' of every state, which would be unique for each state of all 9! i.e. 362880 combinations possible (Checked using **TestHashMethod.java**). This value will be used to store states that were already **visited** in the A-star algorithm, rather than storing the whole 'Puzzle8' object or the array
4. 'equals' method uses 'hashCode' rather than comparing for every array element
5. **Operations** include moving blank space - **up, down, right** or **left**. The operation will return a new instance of 'Puzzle8', which would have the parent as the current instance
6. Correct logic is applied for each of the operations while dealing with a 1D representation
7. E.g. Up operation swaps the 'blank' at (position) with element at (position - 3)
8. If the operation is not possible, null is returned

```

Puzzle8.java TestHashMethod.java AStar.java
1 package project;
2
3 import java.util.ArrayList;
4
5
6
7
8
9
10 public class AStar {
11
12     List<Integer> initial;
13     List<Integer> goal;
14     private List<Puzzle8> fringe;
15     private Set<Integer> visited;
16
17     public AStar(List<Integer> initial, List<Integer> goal) {
18         super();
19         this.initial = initial;
20         this.goal = goal;
21         Puzzle8 puzzle = new Puzzle8(initial);
22         this.fringe = new ArrayList<>();
23         this.visited = new HashSet<>();
24         insertPuzzleNode(puzzle, 0);
25     }
26

```

```

public static void main(String[] args) {
    // Integer[] a1 = {0, 1, 3, 4, 2, 5, 7, 8, 6}; Integer[] b1 = {1, 2, 3, 4, 5, 6, 7, 8, 0};
    // Integer[] a1 = {1, 2, 3, 7, 4, 5, 6, 8, 0}; Integer[] b1 = {1, 2, 3, 8, 6, 4, 7, 5, 0};
    // Integer[] a1 = {2, 8, 1, 3, 4, 6, 7, 5, 0}; Integer[] b1 = {3, 2, 1, 8, 0, 4, 7, 5, 6};
    // List<Integer> a = new ArrayList<>(Arrays.asList(a1));
    // List<Integer> b = new ArrayList<>(Arrays.asList(b1));
    // AStar star = new AStar(a, b);

    List<Integer> a = new ArrayList<>();
    List<Integer> b = new ArrayList<>();

    Scanner src = new Scanner(System.in);

    System.out.println("Add numbers from 0-9 to INITIAL puzzle state serially, separated by SPACE and/or NEW LINE.");
    while(a.size() < 9) {
        Integer n = src.nextInt();
        if(n >= 0 && n < 9 && !a.contains(n)) {
            a.add(n);
        } else {
            System.out.println("Invalid. Add numbers from 0-9.");
        }
    }

    System.out.println("Add numbers from 0-9 to GOAL puzzle state serially.");
    while(b.size() < 9) {
        Integer n = src.nextInt();
        if(n >= 0 && n < 9 && !b.contains(n)) {
            b.add(n);
        } else {
            System.out.println("Invalid. Add numbers from 0-9.");
        }
    }

    AStar star = new AStar(a, b);

    star.runAlgorithm();
    src.close();
}
}

```

```

/**
 * Insert a new puzzle arrangement in the fringe at the right position,
 * so the fringe stays sorted.
 * Insertion sort technique is used to insert the new puzzle
 */
private void insertPuzzleNode(Puzzle8 puzzle, int depth) {
    int position = fringe.size();
    Integer heuristicValue = runHeuristic2(puzzle.getA());
    puzzle.setHeuristicValue(heuristicValue);
    puzzle.setDepth(depth);

    for(int i=0; i<fringe.size(); i++) {
        if(puzzle.getTotalHeuristicValue() < fringe.get(i).getTotalHeuristicValue()) {
            position = i;
            break;
        }
    }
    fringe.add(position, puzzle);
}

/**
 * Manhattan distance
 * Heuristic function on the 8-puzzle
 */
public Integer runHeuristic1(List<Integer> a) {
    Integer heuristic = Integer.MAX_VALUE;
    if(a != null) {
        heuristic = 0;
        for(int i=0; i<a.size(); i++) {
            int index = goal.indexOf(a.get(i));
            heuristic += Math.abs(i*3 - index*3) + Math.abs(i/3 - index/3);
        }
    }

    return heuristic;
}

/**
 * Missing pieces
 * Heuristic function on the 8-puzzle
 */
public Integer runHeuristic2(List<Integer> a) {
    Integer heuristic = null;
    if(a != null) {
        heuristic = 0;
        for(int i=0; i<a.size(); i++) {
            if(!a.get(i).equals(goal.get(i))) {
                heuristic++;
            }
        }
    }

    return heuristic;
}
}

```

1. 'Astar' class instance will take the **initial** and the **goal** state
2. A '**fringe**' of puzzles will be created, that will include the initial state already present
3. Again, the states are 1D arrays, represented for the 8-puzzle problem, to improve performance by keeping minimal references

4. User input will be taken as initial and goal state

5. '**insertPuzzleNode**' inserts every expanded node/state to the fringe
6. '**Insertion sort**' technique is used to determine the right position to insert the new puzzle based on '**totalHeuristic=(g)+(h)**'
7. Two heuristic functions are defined, that are called in '**insertPuzzleNode**'
8. '**runHeuristic1**' is **Manhattan distance**
"Formula for 1D array =
 $\text{abs}|\text{index}\%3 - \text{correctIndex}\%3| + \text{abs}|\text{index}/3 - \text{correctIndex}/3|$ "
9. '**runHeuristic2**' is **Misplaced tiles**
Formula for 1D array
=**(counter++)** for every mismatched index

```

/**
 * Run the A-star algorithm and display the path
 */
public void runAlgorithm() {
    int totalExpandedNodes = 0;
    int totalVisitedNodes = 0;
    Puzzle8 puzzle = null;

    while(true) {
        puzzle = fringe.remove(0);
        puzzle.display();
        if(puzzle.getHeuristicValue().intValue() == 0) {
            break;
        }

        visited.add(puzzle.hashCode());
        totalVisitedNodes++;

        Puzzle8 puzzleUp = puzzle.moveUp();
        Puzzle8 puzzleDown = puzzle.moveDown();
        Puzzle8 puzzleRight = puzzle.moveRight();
        Puzzle8 puzzleLeft = puzzle.moveLeft();

        if(puzzleUp != null && !visited.contains(puzzleUp.hashCode())) {
            totalExpandedNodes++;
            insertPuzzleNode(puzzleUp, puzzle.getDepth() + 1);
        }
        if(puzzleDown != null && !visited.contains(puzzleDown.hashCode())) {
            totalExpandedNodes++;
            insertPuzzleNode(puzzleDown, puzzle.getDepth() + 1);
        }
        if(puzzleRight != null && !visited.contains(puzzleRight.hashCode())) {
            totalExpandedNodes++;
            insertPuzzleNode(puzzleRight, puzzle.getDepth() + 1);
        }
        if(puzzleLeft != null && !visited.contains(puzzleLeft.hashCode())) {
            totalExpandedNodes++;
            insertPuzzleNode(puzzleLeft, puzzle.getDepth() + 1);
        }
    }

    System.out.println("Total Expanded nodes: " + totalExpandedNodes);
    System.out.println("Total Visited nodes: " + totalVisitedNodes);

    StringBuilder builder = new StringBuilder();
    while(puzzle != null) {
        builder.insert(0, puzzle.display());
        puzzle = puzzle.getParentPuzzle();
    }
    System.out.println();
    System.out.print("Path: -");
    System.out.println(builder.toString());
}

/**
 * Insert a new puzzle arrangement in the fringe at the right position
 */

```

Algorithm: -

1. The loop breaks when the '**heuristic value**' equals 0
2. The '**fringe**' is always sorted as the sorting is carried while inserting
3. Hence, the least heuristic node/puzzle is at the 0th index
4. Add its '**hashCode**' to the '**visited**' list and increment '**totalVisited**' counter
5. If it is not the goal, **expand** to generate more nodes by performing operations – **up, down, right, left**, whichever is possible, and add the generated nodes/puzzles to the fringe by insert operation maintaining the sorting
6. Increment the '**totalExpanded**' counter
7. When the loop breaks, the '**puzzle**' object will contain the goal state
8. **Trace the complete path by '**parentPuzzle**' link**

Sample results for Misplaced Tile

Heuristics

Add numbers from 0-9 to INITIAL
puzzle state serially, separated by
SPACE and/or NEW LINE.

0 1 3 4 2 5 7 8 6

Add numbers from 0-9 to GOAL
puzzle state serially.

1 2 3 4 5 6 7 8 0

Total Expanded nodes: 9

Total Visited nodes: 4

Path: -

_ 1 3

4 2 5

7 8 6

1 _ 3

4 2 5

7 8 6

1 2 3

4 _ 5

7 8 6

1 2 3

4 5 _

7 8 6

1 2 3

4 5 6

7 8 _

Process finished with exit code 0

Add numbers from 0-9 to INITIAL
puzzle state serially, separated by
SPACE and/or NEW LINE.

1 2 3 7 4 5 6 8 0

Add numbers from 0-9 to GOAL
puzzle state serially.

1 2 3 8 6 4 7 5 0

Total Expanded nodes: 43

Total Visited nodes: 23

Path: -

1 2 3

7 4 5

6 8 _

1 2 3

7 4 _

6 8 5

1 2 3

7 _ 4

6 8 5

1 2 3

7 8 4

6 _ 5

1 2 3

7 8 4

_ 6 5

1 2 3

_ 8 4

7 6 5

1 2 3

8 _ 4

7 6 5

1 2 3

8 6 4

7 _ 5

1 2 3

8 6 4

7 5 _

Process finished with exit code 0

Add numbers from 0-9 to INITIAL
puzzle state serially, separated by
SPACE and/or NEW LINE.

2 8 1 3 4 6 7 5 0

Add numbers from 0-9 to GOAL
puzzle state serially.

3 2 1 8 0 4 7 5 6

Total Expanded nodes: 14

Total Visited nodes: 7

Path: -

2 8 1

3 4 6

7 5 _

2 8 1

3 4 _

7 5 6

2 8 1

3 _ 4

7 5 6

2 _ 1

3 8 4

7 5 6

_ 2 1

3 8 4

7 5 6

3 2 1

_ 8 4

7 5 6

3 2 1

8 _ 4

7 5 6

Process finished with exit code 0

Add numbers from 0-9 to INITIAL
puzzle state serially, separated by
SPACE and/or NEW LINE.

2 8 3 1 6 4 7 0 5

Add numbers from 0-9 to GOAL
puzzle state serially.

1 2 3 8 0 4 7 6 5

Total Expanded nodes: 13

Total Visited nodes: 6

Path: -

2 8 3

1 6 4

7 _ 5

2 8 3

1 _ 4

7 6 5

2 _ 3

1 8 4

7 6 5

_ 2 3

1 8 4

7 6 5

1 2 3

_ 8 4

7 6 5

1 2 3

8 _ 4

7 6 5

Process finished with exit code 0

Add numbers from 0-9 to INITIAL
puzzle state serially, separated by
SPACE and/or NEW LINE.

8 1 3 4 0 2 7 6 5

Add numbers from 0-9 to GOAL
puzzle state serially.

1 2 3 4 5 6 7 8 0

Total Expanded nodes: 555

Total Visited nodes: 338

Path: -

8 1 3

4 _ 2

7 6 5

8 1 3

4 2 _

7 6 5

8 1 3

4 2 5

7 6 _

8 1 3

4 2 5

7 _ 6

8 1 3

4 2 5

_ 7 6

8 1 3

_ 2 5

4 7 6

_ 1 3

8 2 5

4 7 6

1 _ 3

8 2 5

4 7 6

1 2 3

8 _ 5

4 7 6

1 2 3

_ 8 5

4 7 6

1 2 3

4 8 5

_ 7 6

1 2 3

4 8 5

7 _ 6

1 2 3

4 _ 5

7 8 6

1 2 3

4 5 _

7 8 6

1 2 3

4 5 6

7 8 _

Process finished with exit code 0

Add numbers from 0-9 to INITIAL
puzzle state serially, separated by
SPACE and/or NEW LINE.

7 2 4 5 0 6 8 3 1

Add numbers from 0-9 to GOAL
puzzle state serially.

1 2 3 4 5 6 7 8 0

Total Expanded nodes: 6790

Total Visited nodes: 4286

Path: -

7 2 4

5 _ 6

8 3 1

7 2 4

5 3 6

8 _ 1

7 2 4

5 3 6

8 1 _

7 2 4

5 3 _

8 1 6

7 2 4

5 _ 3

8 1 6

7 2 4

_ 5 3

8 1 6

_ 2 4

7 5 3

8 1 6

2 _ 4

7 5 3

8 1 6

2 4 _

7 5 3

8 1 6

2 4 3

7 5 _

8 1 6

2 4 3

7 _ 5

8 1 6

2 4 3

7 1 5

8 _ 6

2 4 3

7 1 5

_ 8 6

2 4 3

_ 1 5

7 8 6

2 4 3

1 _ 5

7 8 6

2 _ 3

1 4 5

7 8 6

_ 2 3

1 4 5

7 8 6

1 2 3

_ 4 5

7 8 6

1 2 3

4 _ 5

7 8 6

1 2 3

4 5 _

7 8 6

1 2 3

4 5 6

7 8 _

Process finished with exit code 0

Add numbers from 0-9 to INITIAL
puzzle state serially, separated by
SPACE and/or NEW LINE.

1 2 0 4 5 3 7 8 6

Add numbers from 0-9 to GOAL
puzzle state serially.

1 2 3 4 5 6 7 8 0

Total Expanded nodes: 4

Total Visited nodes: 2

Path: -

1 2 _

4 5 3

7 8 6

1 2 3

4 5 _

7 8 6

1 2 3

4 5 6

7 8 _

Process finished with exit code 0

Add numbers from 0-9 to INITIAL
puzzle state serially, separated by
SPACE and/or NEW LINE.

1 2 3 0 4 6 7 5 8

Add numbers from 0-9 to GOAL
puzzle state serially.

1 2 3 4 5 6 7 8 0

Total Expanded nodes: 8

Total Visited nodes: 3

Path: -

1 2 3

_ 4 6

7 5 8

1 2 3

4 _ 6

7 5 8

1 2 3

4 5 6

7 _ 8

1 2 3

4 5 6

7 8 _

Process finished with exit code 0

Add numbers from 0-9 to INITIAL
puzzle state serially, separated by
SPACE and/or NEW LINE.

3 8 2 4 5 6 1 7 0

Add numbers from 0-9 to GOAL
puzzle state serially.

1 2 3 4 5 6 7 8 0

Total Expanded nodes: 14780

Total Visited nodes: 9385

Path: -

3 8 2

4 5 6

1 7 _

3 8 2

4 5 _

1 7 6

3 8 2

4 _ 5

1 7 6

3 8 2

_ 4 5

1 7 6

3 8 2

1 4 5

_ 7 6

3 8 2

1 4 5

7 _ 6

3 8 2

1 4 5

7 6 _

3 8 2

1 4 _

7 6 5

3 8 _

1 4 2

7 6 5

3 _ 8

1 4 2

7 6 5

_ 3 8

1 4 2

7 6 5

1 3 8

_ 4 2

7 6 5

1 3 8

4 _ 2

7 6 5

1 3 8

4 2 _

7 6 5

1 3 _

4 2 8

7 6 5

1 _ 3

4 2 8

7 6 5

1 2 3

4 _ 8

7 6 5

1 2 3

4 6 8

7 _ 5

1 2 3

4 6 8

7 5 _

1 2 3

4 6 _

7 5 8

1 2 3

4 _ 6

7 5 8

1 2 3

4 5 6

7 _ 8

1 2 3

4 5 6

7 8 _

Process finished with exit code 0

Sample Solution for Manhattan
distance heuristics

Add numbers from 0-9 to INITIAL
puzzle state serially, separated by
SPACE and/or NEW LINE.

0 1 3

4 2 5

7 8 6

Add numbers from 0-9 to GOAL
puzzle state serially.

1 2 3

4 5 6

7 8 0

Total Expanded nodes: 9

Total Visited nodes: 4

Path: -

_ 1 3

4 2 5

7 8 6

1 _ 3

4 2 5

7 8 6

1 2 3

4 _ 5

7 8 6

1 2 3

4 5 _

7 8 6

1 2 3

4 5 6

7 8 _

Process finished with exit code 0

Add numbers from 0-9 to INITIAL
puzzle state serially, separated by
SPACE and/or NEW LINE.

1 2 3

7 4 5

6 8 0

Add numbers from 0-9 to GOAL
puzzle state serially.

1 2 3 8 6

4 7 5 0

Total Expanded nodes: 22

Total Visited nodes: 11

Path: -

1 2 3

7 4 5

6 8 _

1 2 3

7 4 _

6 8 5

1 2 3

7 _ 4

6 8 5

1 2 3

7 8 4

6 _ 5

1 2 3

7 8 4

_ 6 5

1 2 3

_ 8 4

7 6 5

1 2 3

8 _ 4

7 6 5

1 2 3

8 6 4

7 _ 5

1 2 3

8 6 4

7 5 _

Process finished with exit code 0

Add numbers from 0-9 to INITIAL
puzzle state serially, separated by
SPACE and/or NEW LINE.

2 8 1 3 4 6 7 5 0

Add numbers from 0-9 to GOAL
puzzle state serially.

3 2 1 8 0 4 7 5 6

Total Expanded nodes: 12

Total Visited nodes: 6

Path: -

2 8 1

3 4 6

7 5 _

2 8 1

3 4 _

7 5 6

2 8 1

3 _ 4

7 5 6

2 _ 1

3 8 4

7 5 6

_ 2 1

3 8 4

7 5 6

3 2 1

_ 8 4

7 5 6

3 2 1

8 _ 4

7 5 6

Process finished with exit code 0

Add numbers from 0-9 to INITIAL
puzzle state serially, separated by
SPACE and/or NEW LINE.

8 1 3 4 0 2 7 6 5

Add numbers from 0-9 to GOAL
puzzle state serially.

1 2 3 4 5 6 7 8 0

Total Expanded nodes: 187

Total Visited nodes: 110

Path: -

8 1 3

4 _ 2

7 6 5

8 1 3

4 2 _

7 6 5

8 1 3

4 2 5

7 6 _

8 1 3

4 2 5

7 _ 6

8 1 3

4 2 5

_ 7 6

8 1 3

_ 2 5

4 7 6

_ 1 3

8 2 5

4 7 6

1 _ 3

8 2 5

4 7 6

1 2 3

8 _ 5

4 7 6

1 2 3

_ 8 5

4 7 6

1 2 3

4 8 5

_ 7 6

1 2 3

4 8 5

7 _ 6

1 2 3
4 _ 5
7 8 6

1 2 3
4 5 _
7 8 6

1 2 3
4 5 6
7 8 _

Process finished with exit code 0

Add numbers from 0-9 to INITIAL
puzzle state serially, separated by
SPACE and/or NEW LINE.

2 8 3 1 6 4 7 0 5

Add numbers from 0-9 to GOAL
puzzle state serially.

1 2 3 8 0 4 7 6 5

Total Expanded nodes: 11

Total Visited nodes: 5

Path: -

2 8 3

1 6 4

7 _ 5

2 8 3

1 _ 4

7 6 5

2 _ 3

1 8 4

7 6 5

_ 2 3

1 8 4

7 6 5

1 2 3

_ 8 4

7 6 5

1 2 3

8 _ 4

7 6 5

Process finished with exit code 0

Add numbers from 0-9 to INITIAL
puzzle state serially, separated by
SPACE and/or NEW LINE.

7 2 4 5 0 6 8 3 1

Add numbers from 0-9 to GOAL
puzzle state serially.

1 2 3 4 5 6 7 8 0

Total Expanded nodes: 746

Total Visited nodes: 460

Path: -

7 2 4

5 _ 6

8 3 1

7 2 4

5 3 6

8 _ 1

7 2 4

5 3 6

8 1 _

7 2 4

5 3 _

8 1 6

7 2 4

5 _ 3

8 1 6

7 2 4

_ 5 3

8 1 6

_ 2 4

7 5 3

8 1 6

2 _ 4

7 5 3

8 1 6

2 4 _

7 5 3

8 1 6

2 4 3

7 5 _

8 1 6

2 4 3

7 _ 5

8 1 6

2 4 3

7 1 5

8 _ 6

2 4 3

7 1 5

_ 8 6

2 4 3

_ 1 5

7 8 6

2 4 3

1 _ 5

7 8 6

2 _ 3

1 4 5

7 8 6

_ 2 3

1 4 5

7 8 6

1 2 3

_ 4 5

7 8 6

1 2 3

4 _ 5

7 8 6

1 2 3

4 5 _

7 8 6

1 2 3

4 5 6

7 8 _

Process finished with exit code 0

Add numbers from 0-9 to INITIAL
puzzle state serially, separated by
SPACE and/or NEW LINE.

1 2 0 4 5 3 7 8 6

Add numbers from 0-9 to GOAL
puzzle state serially.

1 2 3 4 5 6 7 8 0

Total Expanded nodes: 4

Total Visited nodes: 2

Path: -

1 2 _

4 5 3

7 8 6

1 2 3

4 5 _

7 8 6

1 2 3

4 5 6

7 8 _

Process finished with exit code 0

Add numbers from 0-9 to INITIAL
puzzle state serially, separated by
SPACE and/or NEW LINE.

1 2 3 0 4 6 7 5 8

Add numbers from 0-9 to GOAL
puzzle state serially.

1 2 3 4 5 6 7 8 0

Total Expanded nodes: 8

Total Visited nodes: 3

Path: -

1 2 3

_ 4 6

7 5 8

1 2 3

4 _ 6

7 5 8

1 2 3

4 5 6

7 _ 8

1 2 3

4 5 6

7 8 _

Process finished with exit code 0

Add numbers from 0-9 to INITIAL
puzzle state serially, separated by
SPACE and/or NEW LINE.

3 8 2 4 5 6 1 7 0

Add numbers from 0-9 to GOAL
puzzle state serially.

1 2 3 4 5 6 7 8 0

Total Expanded nodes: 2881

Total Visited nodes: 1789

Path: -

3 8 2

4 5 6

1 7 _

3 8 2

4 5 _

1 7 6

3 8 2

4 _ 5

1 7 6

3 8 2

_ 4 5

1 7 6

3 8 2

1 4 5

_ 7 6

3 8 2

1 4 5

7 _ 6

3 8 2	1 3 _	1 2 3
1 4 5	4 2 8	4 5 6
7 6 _	7 6 5	7 8 _
3 8 2	1 _ 3	Process finished with exit code 0
1 4 _	4 2 8	
7 6 5	7 6 5	
3 8 _	1 2 3	
1 4 2	4 _ 8	
7 6 5	7 6 5	
3 _ 8	1 2 3	
1 4 2	4 6 8	
7 6 5	7 _ 5	
_ 3 8	1 2 3	
1 4 2	4 6 8	
7 6 5	7 5 _	
1 3 8	1 2 3	
_ 4 2	4 6 _	
7 6 5	7 5 8	
1 3 8	1 2 3	
4 _ 2	4 _ 6	
7 6 5	7 5 8	
1 3 8	1 2 3	
4 2 _	4 5 6	
7 6 5	7 _ 8	