

# AI

# Assignment 1

Name	ID
Mai Ahmed Hussein	19016736
Habiba Osama Zaki	19015575
Basel Ahmed Awad	19015513
Ali Hassan Alsharawy	19016013



## Problem statement:

Solving the 8-puzzle game using BFS, DFS, and A\* with the heuristic function as euclidean distance or manhattan distance to reach goal state "012345678".

## Assumptions and Details:

- The default algorithm will be **BFS** if the user didn't choose an algorithm.
- Code detects invalid and unsolvable states and doesn't attempt to solve them.

# Data Structures:

- **HashSet:**

Used to store the visited (explored) states.

Also used for the frontier list in BFS and DFS to check if it contains a certain state or not in  $O(1)$  instead of  $O(n)$  of the Queue and Stack (.contains) method.

- **Queue:**

Used in BFS as the frontier list.

- **Stack:**

Used in DFS as the frontier list.

- **PriorityQueue:**

Used in A\* as the frontier list.

- **HashMap:**

Used in A\* to store the lowest cost for the states currently in the frontier only.

- **LinkedList:**

Used to store the final path states.

Also used to store the state's children list(neighbour list).

- **Node:**

Added data structure that contains puzzle state stored as int, depth, reference to parent node, reference to children linkedlist(neighbour list), and the index of the zero in the puzzle state.

# **Algorithms:**

## **BFS Algorithm:**

Start by checking the shallowest node first, level by level using Queue as frontier.

## **DFS Algorithm:**

Start by checking the deepest node first, using Stack as frontier.

## **A\* Algorithm:**

Searching according to the  $\min(\text{Cost}(x) + \text{heuristic}(x))$  using Priority Queue as frontier.

The heuristic is calculated using 2 different methods Manhattan and Euclidean.

## **public void tracePath(Node n) :**

Getting the final path starting from goal state node to the root node (initial state).

## Manhattan Vs Euclidean:

**EX 1: 125340678**

POV	Manhattan	Euclidean
Max Depth	3	3
Cost of Path	3	3
Search Path	3	3
No. of Nodes Visited	4	4
No. of Nodes Expanded	8	8
Running Time in $\mu$ s	432	1515

**EX 2: 768243105**

POV	Manhattan	Euclidean
Max Depth	27	27
Cost of Path	27	27
Search Path	27	27
No. of Nodes Visited	4984	10629
No. of Nodes Expanded	13567	28775
Running Time in $\mu$ s	59975	320786

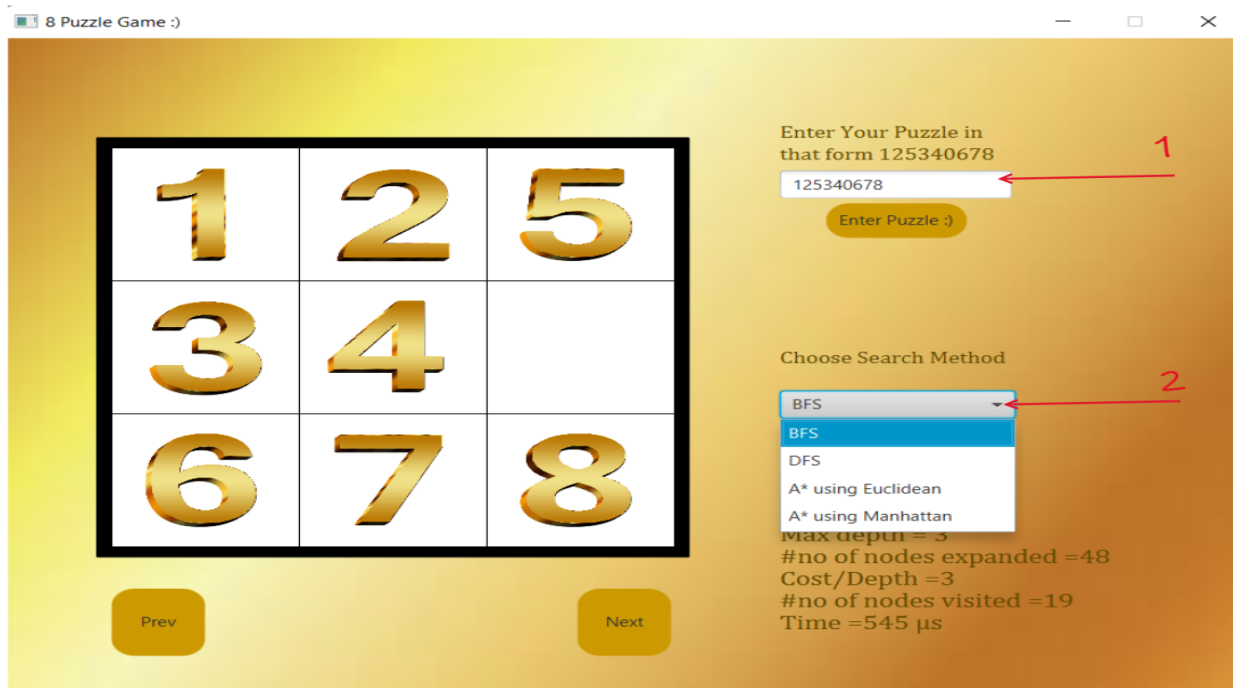
**EX 3: 182043765**

POV	Manhattan	Euclidean
Max Depth	21	21
Cost of Path	21	21
Search Path	21	21
No. of Nodes Visited	1388	2276
No. of Nodes Expanded	3783	6143
Running Time in $\mu$ s	7940	26980

From the table shown above, it appears that Manhattan expands less nodes and has shorter running time than Euclidean. So,

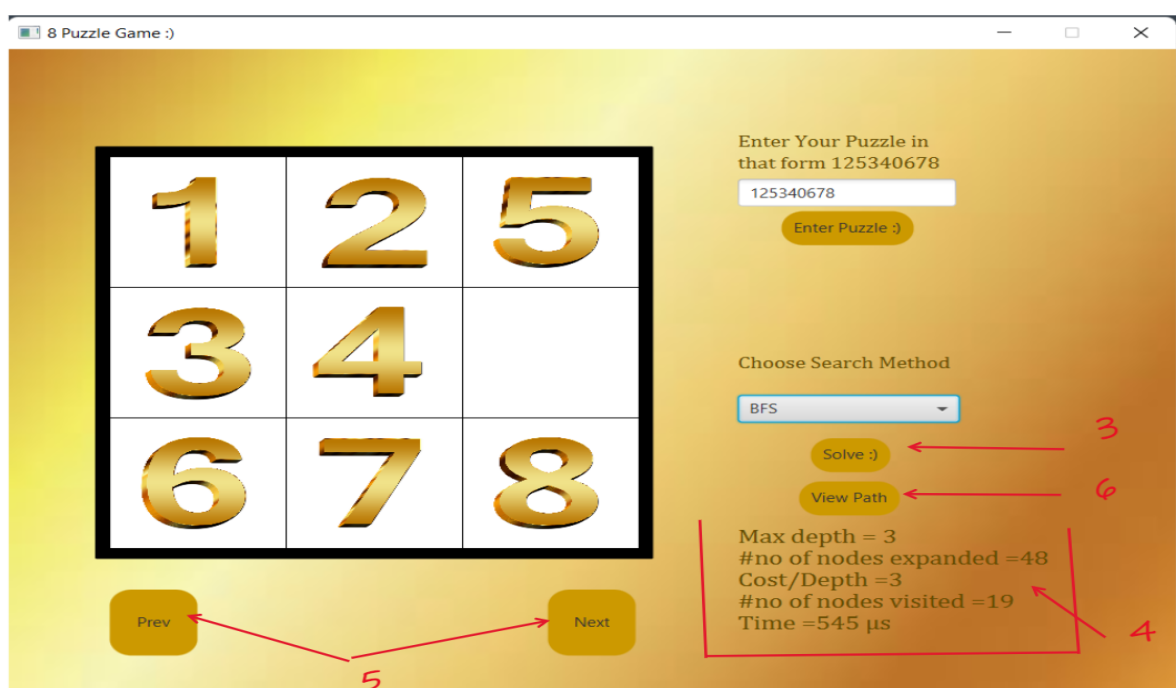
**Manhattan Distance is more admissible.**

# How To Use:



1- first user enters the initial case in that form 125340678 and then presses enter the puzzle. The puzzle will be viewed as shown.

2- choose the search method from the drop down menu . The default is BFS.



3- enter, then the results of depth ,time and cost will appear (number4).

4- the information wanted for this search.

5- you can press next to know the next move of the puzzle and keep moving till reach our goal state 012345678.

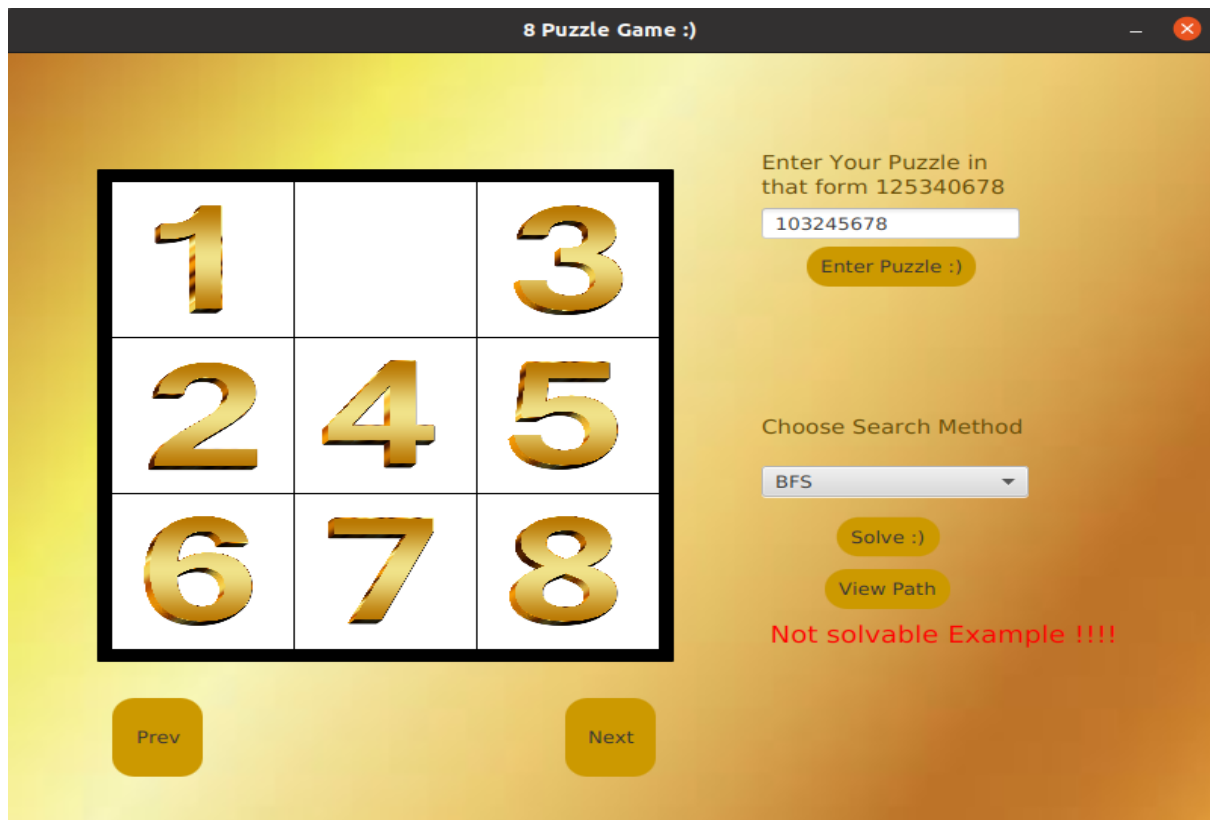
6- View path will open a new window that shows the whole path as in the following photo, also the path is printed in "path.txt" file.

The screenshot shows a web-based application for solving a 3x3 puzzle. On the left, there are three 3x3 grids. The top grid contains the numbers 1, 2, 5 in the first row, 3, 4, and an empty space in the second row, and 6, 7, 8 in the third row. The middle grid contains 1, 2, and an empty space in the first row, 3, 4, 5 in the second row, and 6, 7, 8 in the third row. The bottom grid contains 1, an empty space, 2 in the first row, 3, 4, 5 in the second row, and 6, 7, 8 in the third row. Below these grids are two buttons: 'Prev' and 'Next'. On the right, there is a form with the following elements:

- A text input field labeled 'Enter Your Puzzle in that form 125340678' containing the text '125340678'.
- A button labeled 'Enter Puzzle :)'. Below this is a 'Choose Search Method' dropdown menu with 'BFS' selected.
- Buttons labeled 'Solve :)' and 'View Path'.
- A section showing search results: 'Max depth = 3', '#no of nodes expanded =48', 'Cost/Depth =3', '#no of nodes visited =19', and 'Time =545 μs'.

## Cases that program handle it:

If the example is not solvable will print not solvable



The screenshot shows the '8 Puzzle Game :)' window. On the left is a 3x3 grid with tiles containing numbers 1 through 8. The tiles are arranged in a 3x3 grid with the following values: Row 1: 1, empty, 3; Row 2: 2, 4, 5; Row 3: 6, 7, 8. On the right, the text 'Enter Your Puzzle in that form 125340678' is displayed above a text input field containing '103245678'. Below the input field is a yellow button labeled 'Enter Puzzle :)'. Further down, the text 'Choose Search Method' is above a dropdown menu showing 'BFS'. Below the dropdown are two yellow buttons: 'Solve :)' and 'View Path'. At the bottom of the right panel, the text 'Not solvable Example !!!!' is displayed in red. At the bottom of the window are two yellow buttons: 'Prev' and 'Next'.

1		3
2	4	5
6	7	8

Enter Your Puzzle in that form 125340678

103245678

Enter Puzzle :)

Choose Search Method

BFS

Solve :)

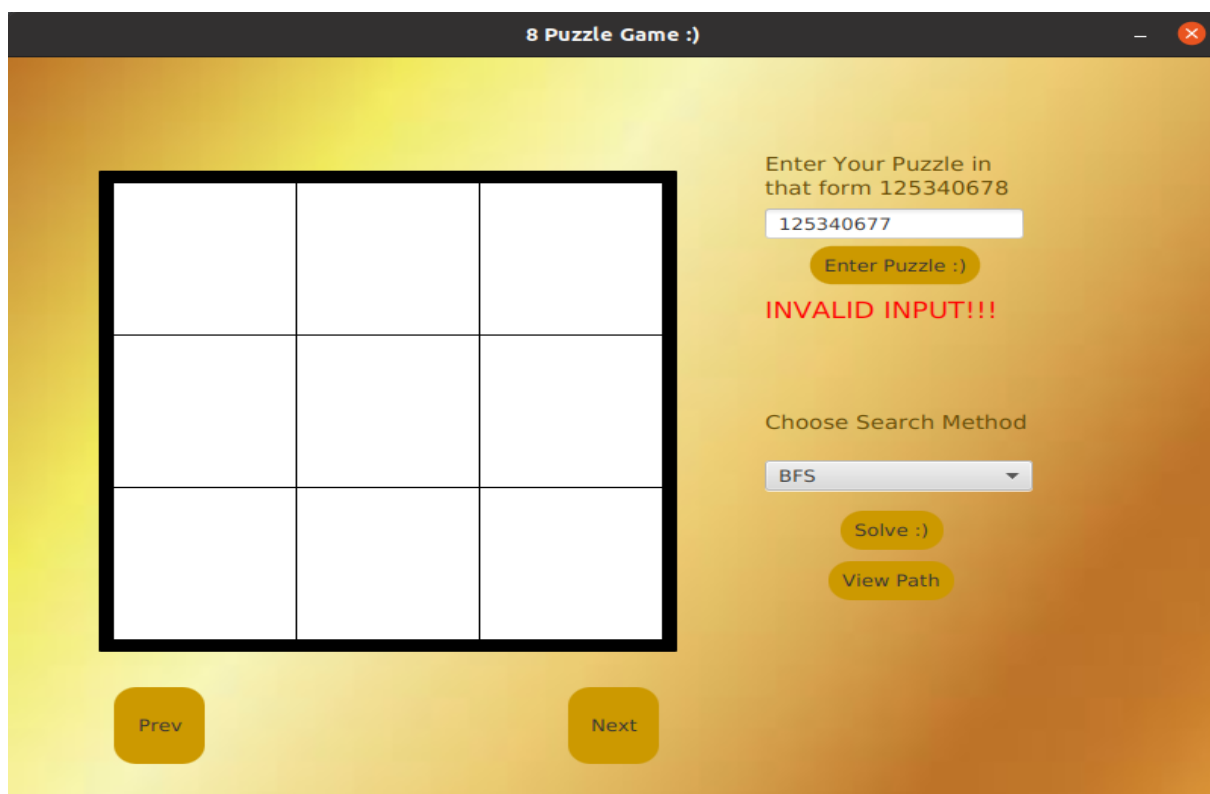
View Path

Not solvable Example !!!!

Prev

Next

If user input incorrect input will print "INVALID INPUT"



The screenshot shows the '8 Puzzle Game :)' window. On the left is a 3x3 grid that is currently empty. On the right, the text 'Enter Your Puzzle in that form 125340678' is displayed above a text input field containing '125340677'. Below the input field is a yellow button labeled 'Enter Puzzle :)'. Further down, the text 'Choose Search Method' is above a dropdown menu showing 'BFS'. Below the dropdown are two yellow buttons: 'Solve :)' and 'View Path'. At the bottom of the right panel, the text 'INVALID INPUT!!!' is displayed in red. At the bottom of the window are two yellow buttons: 'Prev' and 'Next'.


Enter Your Puzzle in that form 125340678

125340677

Enter Puzzle :)

INVALID INPUT!!!

Choose Search Method

BFS

Solve :)

View Path

Prev

Next



## Sample Runs:

**EX 1: 125340678**

**BFS:**

8 Puzzle Game :)

1	2	5
3	4	
6	7	8

Enter Your Puzzle in that form 125340678  
125340678  
Enter Puzzle :)

Choose Search Method  
BFS

Solve :)  
View Path

Max depth = 3  
#no of nodes expanded = 48  
Cost/Depth = 3  
#no of nodes visited = 19  
Time = 2586  $\mu$ s

Prev Next

**Path:**

1	2	5
3	4	
6	7	8

1	2	
3	4	5
6	7	8

1		2
3	4	5
6	7	8

	1	2
3	4	5
6	7	8

## DFS:

8 Puzzle Game :)

1	2	5
3	4	
6	7	8

Enter Your Puzzle in that form 125340678

125340678

Enter Puzzle :)

Choose Search Method

DFS

Solve :)

View Path

Max depth = 3  
#no of nodes expanded = 8  
Cost/Depth = 3  
#no of nodes visited = 4  
Time = 181  $\mu$ s

Prev Next

## A\* using Euclidean:

8 Puzzle Game :)

1	2	5
3	4	
6	7	8

Enter Your Puzzle in that form 125340678

125340678

Enter Puzzle :)

Choose Search Method

A\* using Euclidean

Solve :)

View Path

Max depth = 3  
#no of nodes expanded = 8  
Cost/Depth = 3  
#no of nodes visited = 4  
Time = 207  $\mu$ s

Prev Next

## A\* using Manhattan:

8 Puzzle Game :)

1	2	5
3	4	
6	7	8

Enter Your Puzzle in that form 125340678  
125340678  
Enter Puzzle :)

Choose Search Method  
A\* using Manhattan

Solve :)  
View Path

Max depth = 3  
#no of nodes expanded = 8  
Cost/Depth = 3  
#no of nodes visited = 4  
Time = 201  $\mu$ s

Prev Next

**EX 2: 012435687**

**BFS:**

8 Puzzle Game :)

	1	2
4	3	5
6	8	7

Enter Your Puzzle in that form 125340678  
012435687  
Enter Puzzle :)

Choose Search Method  
BFS

Solve :)  
View Path

Max depth = 18  
#no of nodes expanded = 55860  
Cost/Depth = 18  
#no of nodes visited = 20504  
Time = 25514  $\mu$ s

Prev Next

## DFS:

8 Puzzle Game :)

	1	2
4	3	5
6	8	7

Prev

Next

Enter Your Puzzle in that form 125340678

012435687

Enter Puzzle :)

Choose Search Method

DFS

Solve :)

View Path

Max depth = 49930  
#no of nodes expanded = 165415  
Cost/Depth = 49930  
#no of nodes visited = 57472  
Time = 250124  $\mu$ s

## A\* using Euclidean:

8 Puzzle Game :)

	1	2
4	3	5
6	8	7

Prev

Next

Enter Your Puzzle in that form 125340678

012435687

Enter Puzzle :)

Choose Search Method

A\* using Euclidean

Solve :)

View Path

Max depth = 18  
#no of nodes expanded = 2001  
Cost/Depth = 18  
#no of nodes visited = 735  
Time = 56481  $\mu$ s

## A\* using Manhattan:

8 Puzzle Game :)

	1	2
4	3	5
6	8	7

Prev

Next

Enter Your Puzzle in that form 125340678

012435687

Enter Puzzle :)

Choose Search Method

A\* using Manhattan

Solve :)

View Path

Max depth = 18  
#no of nodes expanded = 1392  
Cost/Depth = 18  
#no of nodes visited = 512  
Time = 39774  $\mu$ s

## Path:

1	2	
4	3	5
6	8	7

1	2	5
4	3	7
6		8

1	2	
3	7	5
4	6	8

3	1	2
4	7	5
6		8

1		2
4	3	5
6	8	7

1	2	5
4		7
6	8	

1		2
3	7	5
4	6	8

3	1	2
4		5
6	7	8

1	2	
4	3	5
6	8	7

1	2	5
4	3	7
6		8

	1	2
3	7	5
4	6	8

3	1	2
	7	5
4	6	8

1	2	5
4	3	7
6	8	

1	2	5
3	7	
4	6	8

3	1	2
4	7	5
	6	8

1	2	5
4	3	7
6	8	

1	2	5
3	7	
4	6	8

3	1	2
4	7	5
	6	8

1	2	5
4	3	7
6	8	

1	2	5
3	7	
4	6	8

3	1	2
4	7	5
	6	8

1	2	5
4	3	7
6	8	

1	2	5
3	7	
4	6	8

3	1	2
4	7	5
	6	8

1	2	5
4	3	7
6	8	

1	2	5
3	7	
4	6	8

3	1	2
4	7	5
	6	8

1	2	5
4	3	7
6	8	

1	2	5
3	7	
4	6	8

3	1	2
4	7	5
	6	8

1	2	5
4	3	7
6	8	

1	2	5
3	7	
4	6	8

3	1	2
4	7	5
	6	8

1	2	5
4	3	7
6	8	

1	2	5
3	7	
4	6	8

3	1	2
4	7	5
	6	8

1	2	5
4	3	7
6	8	

1	2	5
3	7	
4	6	8

3	1	2
4	7	5
	6	8

1	2	5
4	3	7
6	8	

1	2	5
3	7	
4	6	8

3	1	2
4	7	5
	6	8

1	2	5
4	3	7
6	8	

1	2	5
3	7	
4	6	8

3	1	2
4	7	5
	6	8

1	2	5
4	3	7
6	8	

1	2	5
3	7	
4	6	8

3	1	2
4	7	5
	6	8

1	2	5
4	3	7
6	8	

1	2	5
3	7	
4	6	8

3	1	2
4	7	5
	6	8

1	2	5
4	3	7
6	8	

1	2	5
3	7	
4	6	8

3	1	2
4	7	5
	6	8

1	2	5
4	3	7
6	8	

1	2	5
3	7	
4	6	8

3	1	2
4	7	5
	6	8

1	2	5
4	3	7
6	8	

1	2	5
3	7	
4	6	8

3	1	2
4	7	5
	6	8

1	2	5
4	3	7
6	8	

1	2	5
3	7	
4	6	8

3	1	2
4	7	5
	6	8

1	2	5
4	3	7
6	8	

1	2	5
3	7	
4	6	8

3	1	2
4	7	5
	6	8

1	2	5
4	3	7
6	8	

1	2	5
3	7	
4	6	8

3	1	2
4	7	5
	6	8

1	2	5
4	3	7
6	8	

1	2	5
3	7	
4	6	8

3	1	2
4	7	5
	6	8

1	2	5
4	3	7
6	8	

1	2	5
3	7	
4	6	8

3	1	2
4	7	5
	6	8

1	2	5
4	3	7
6	8	

1	2	5
3	7	
4	6	8

3	1	2
4	7	5
	6	8

1	2	5
4	3	7
6	8	

1	2	5
3	7	
4	6	8

3	1	2
4	7	5
	6	8

1	2	5
4	3	7
6	8	

1	2	5
3	7	
4	6	8

3	1	2
4	7	5
	6	8

1	2	5
4	3	7
6	8	

1	2	5
3	7	
4	6	8

3	1	2
4	7	5
	6	8

1	2	5
4	3	7
6	8	

1	2	5
3	7	
4	6	8

3	1	2
4	7	5
	6	8

1	2	5
4	3	7
6	8	

1	2	5
3	7	
4	6	8

3	1	2
4	7	5
	6	8

1	2	5
4	3	7
6	8	

1	2	5
3	7	
4	6	8

3	1	2
4	7	5
	6	8

1	2	5
4	3	7
6	8	

1	2	5
3	7	
4	6	8

3	1	2
4	7	5
	6	8

1	2	5
4	3	7
6	8	

1	2	5
3	7	
4	6	8

3	1	2
4	7	5
	6	8

1	2	5
4	3	7
6	8	

1	2	5
3	7	
4	6	8

3	1	2
4	7	5
	6	8

1	2	5
4	3	7
6	8	

1	2	5
3	7	
4	6	8

3	1	2
4	7	5
	6	8

1	2	5
4	3	7
6	8	

1	2	5
3	7	
4	6	8

3	1	2
4	7	5
	6	8

1	2	5
4	3	7
6	8	

1	2	5
3	7	
4	6	8

3	1	2
4	7	5
	6	8

1	2	5
4	3	7
6	8	

1	2	5
3	7	
4	6	8

3	1	2
4	7	5
	6	8

1	2	5
4	3	7
6	8	

1	2	5
3	7	
4	6	8

3	1	2
4	7	5
	6	8

1	2	5
4	3	7
6	8	

1	2	5
3	7	
4	6	8

3	1	2
4	7	5
	6	8

1	2	5
4	3	7
6	8	

1	2	5
3	7	
4	6	8

3	1	2
4	7	5
	6	8

1	2	5
4	3	7
6	8	

1	2	5
3	7	
4	6	8

3	1	2
4	7	5
	6	8

1	2	5
4	3	7
6	8	

1	2	5
3	7	
4	6	8

3	1	2
4	7	5
	6	8

1	2	5
4	3	7
6	8	

1	2	5
3	7	
4	6	8

3	1	2
4	7	5
	6	8

1	2	5
4	3	7
6	8	

1	2	5
3	7	
4	6	8

3	1	2
4	7	5
	6	8

1	2	5
4	3	7
6	8	

1	2	5
3	7	
4	6	8

3	1	2
4	7	5
	6	8

1	2	5
4	3	7
6	8	

1	2	5
3	7	
4	6	8

3	1	2
4	7	5
	6	8

1	2	5
4	3	7
6	8	

1	2	5
3	7	
4	6	8

3	1	2
4	7	5
	6	8

1	2	5
4	3	7
6	8	

1	2	5
3	7	
4	6	8

3	1	2
4	7	5
	6	8

1	2	5
4	3	7
6	8	

1	2	5
3	7	
4	6	8

3	1	2
4	7	5
	6	8

1	2	5
4	3	7
6	8	

1	2	5
3	7	
4	6	8

3	1	2
4	7	5
	6	8

1	2	5
4	3	7
6	8	

1	2	5
3	7	
4	6	8

3	1	2
4	7	5
	6	8

1	2	5
4	3	7
6	8	

1	2	5
3	7	
4	6	8

3	1	2
4	7	5
	6	8

1	2	5
4	3	7
6	8	

1	2	5
3	7	
4	6	8

3	1	2
4	7	5
	6	8

1	2	5
4	3	7
6	8	

1	2	5
3	7	
4	6	8

3	1	2
4	7</	

## Invalid Input:

8 Puzzle Game :)


Enter Your Puzzle in that form 125340678

Enter Puzzle :)

**INVALID INPUT!!!**

Choose Search Method

BFS

Solve :)

View Path

Prev Next

## Unsolvable State:

8 Puzzle Game :)

1		3
2	4	5
6	7	8

Enter Your Puzzle in that form 125340678

Enter Puzzle :)

Choose Search Method

BFS

Solve :)

View Path

**Not solvable Example !!!!**

Prev Next