

AI Report

NAME : Ahmed Hatem

ID : 4593

Description:

It is required to implement the famous 8 puzzle solver using BFS, DFS, A* where an empty block can be swapped up, down, left, right.

The initial state is entered by the user which is represented with 9 numbers represented on 3*3 board

The action on each state is swapping the with any of the adjacent blocks which results to a new state of the puzzle.

By using the 3 search algorithms mentioned before we reach the goal state and the puzzle is solved.

Implementation:

When a user enters the initial state of 9 numbers separated with commas, the program removes the string and the state is represented with 9 numbers. A string is easy for acting on the state, then calling any of the search algorithm depending on the user desire giving it the initial state and goal state entered by the user, BFS expands the possible states of the initial state adding them to queue then choosing the shallowest node and the loop repeats this operation until the goal is found, DFS expands the possible states of the initial state adding them to Stack then choosing the last node entered to stack and the loop repeats this operation until the goal is found, A* expands the possible states of the initial state adding them to priority queue with priority to minimum total cost, Where $\text{total cost} = \text{distanceCovered} + \text{heuristic function}$, heuristic could be Manhattan or Euclidean.

Data structures used:

- **Queue in BFS algorithm.**
- **Stack in DFS algorithm.**
- **Priority queue in A* algorithm.**
- **Array List to store neighbours of each state.**
- **Set to store visited states.**

Sample runs :

- The user is asked to enter initial state ,goal state, type of search algorithm to be used.

Output - puzzleSolver (run) X



run:



Enter your initial state:

1,2,5,3,4,0,6,7,8



Enter your goal state:



0,1,2,3,4,5,6,7,8

Choose a search algorithm:

1.BFS

2.DFS

3.A*

1

BFS :

```
MAIN PATH:

125
340
678

120
345
678                Swapped Up with 5

102
345
678                Swapped Left with 2

012
345
678                Swapped Left with 1

Totalcost = 3
Expanded nodes= 21
Running Time = 11
Execution time= 0 millisec
```

```
output - Notepad
File Edit Format View Help
Initial state is 125340678

New State is : 125340678
125340678 expands: 120345678 125348670 125304678

New State is : 120345678
120345678 expands: 102345678

New State is : 125348670
125348670 expands: 125348607

New State is : 125304678
125304678 expands: 105324678 125374608 125034678

New State is : 102345678
102345678 expands: 142305678 012345678

New State is : 125348607
125348607 expands: 125308647 125348067

New State is : 105324678
105324678 expands: 015324678 150324678

New State is : 125374608
125374608 expands: 125374068 125374680

New State is : 125034678
125034678 expands: 025134678 125634078

New State is : 142305678
142305678 expands: 142375608 142035678 142350678

**** Goal state is reached : 012345678 ****
```

DFS:

```
MAIN PATH:

125
340
678

125
304
678          Swapped Left with 4

125
034
678          Swapped Left with 3

125
634
078          Swapped Down with 6

125
634
708          Swapped Right with 7
```

```
312
645
708          Swapped Left with 8

312
645
078          Swapped Left with 7

312
045
678          Swapped Up with 6

012
345
678          Swapped Up with 3

Totalcost = 431
Expanded nodes= 774
Running Time = 94
```



output - Notepad

File Edit Format View Help

312876054 expands: 312076854

New State is : 312076854

312076854 expands: 012376854 312706854

New State is : 312706854

312706854 expands: 302716854 312756804 312760854

New State is : 312760854

312760854 expands: 310762854 312764850

New State is : 312764850

312764850 expands: 312764805

New State is : 312764805

312764805 expands: 312704865 312764085

New State is : 312764085

312764085 expands: 312064785

New State is : 312064785

312064785 expands: 012364785 312604785

New State is : 312604785

312604785 expands: 302614785 312684705 312640785

New State is : 312640785

312640785 expands: 310642785 312645780

New State is : 312645780

312645780 expands: 312645708

New State is : 312645708

312645708 expands: 312605748 312645078

New State is : 312645078

312645078 expands: 312045678

New State is : 312045678

312045678 expands: 012345678

**** Goal state is reached : 012345678 ****

A*:

Manhattan:

```
output - Notepad
File Edit Format View Help
Initial state is 125340678

New State is : 125340678 (h=6,g=0,f=6)
125340678 expands: 120345678-->(h=4,g=1,f=5) 125348670-->(h=8,g=1,f=9) 125304678-->(h=6,g=1,f=7)

New State is : 120345678 (h=4,g=1,f=5)
120345678 expands: 102345678-->(h=2,g=2,f=4)

New State is : 102345678 (h=2,g=2,f=4)
102345678 expands: 142305678-->(h=4,g=3,f=7) 012345678-->(h=0,g=3,f=3)

**** Goal state is reached : 012345678 ****
```

```
MAIN PATH:

125
340
678

120
345
678          Swapped Up with 5

102
345
678          Swapped Left with 2

012
345
678          Swapped Left with 1

Totalcost = 3
Expanded Nodes: 6
Running Time: 4
Execution time: 0 millisec
```

● Euclidean :

output - Notepad

File Edit Format View Help

Initial state is 125340678

New State is : 125340678 (h=5,g=0,f=5)

125340678 expands: 120345678-->(h=4,g=1,f=5) 125348670-->(h=6,g=1,f=7) 125304678-->(h=5,g=1,f=6)

New State is : 120345678 (h=4,g=1,f=5)

120345678 expands: 102345678-->(h=2,g=2,f=4)

New State is : 102345678 (h=2,g=2,f=4)

102345678 expands: 142305678-->(h=3,g=3,f=6) 012345678-->(h=0,g=3,f=3)

**** Goal state is reached : 012345678 ****

MAIN PATH:

125

340

678

120

345

678

Swapped Up with 5

102

345

678

Swapped Left with 2

012

345

678

Swapped Left with 1

Totalcost = 3

Expanded Nodes: 6

Running Time: 4

Execution time: 15 millisec

Euclidean and Manhattan :

Since that states cannot be swapped diagonally then Euclidean is a bad idea, As long that Euclidean gives the shortest paths but A^* will take longer time to reach the goal so Manhattan in this case is more admissible.