8 – Puzzle using A* Algorithm

INTELLIGENT SYSTEMS

PROJECT DOCUMENTATION REPORT

Submitted To - Dewan T. Ahmed, Ph.D.

Implemented By – Ayush Shetty & Gautham Krishna R

8-PUZZLE FORMULATION:

- The 8-puzzle consists of an area divided into a grid, 3 by 3 for the 8-puzzle.
- On each grid square is a tile, except for one square which remains empty. Thus, there are eight tiles in the 8-puzzle. Tiles are numbered to 8 for the 8-puzzle, so that each tile can be uniquely identified.
- The aim of the puzzle is to get to the final configuration of tiles i.e. goal state from another given configuration i.e. initial state by sliding the individual tiles around the grid until the goal state is achieved.

***Four Directional Moves*:** The empty tile can be swapped with another tile by moving it in four possible directions i.e. up, down, left, right depending on the positions of the rest of the tiles. The tiles are moved in every iteration until the goal state is obtained.

- The 8 puzzle problem can be solved using Blind Search Algorithms or Heuristics Search Techniques.

- Breadth First Search(BFS) and Depth First Search(DFS) algorithms are the two commonly used blind search techniques.

There are two heuristic search techniques used to solve the 8 puzzle problem:

1. **Misplaced Tiles Heuristic:** The heuristic is based upon number of misplaced tiles from their goal state.

2.**Manhattan Distance Heuristic:** The distance between two points measured along axes at right angles.

This project shows the implementation of both the heuristics using the A* algorithm.

A * search algorithm is used to solve this puzzle which illustrates a general artificial intelligence methodology. It is an informed search algorithm which is used in path findings and graph traversals. It is a combination of uniform cost search (UCS) and best first search (greedy), which avoids expanding expensive paths. A* star uses admissible heuristics which is optimal as it avoids over-estimating the path to goal state. The evaluation function A* uses for calculating the distance is:

- f(n) = g(n) + h(n)
- where g(n) = cost so far to reach n,
- h(n) = estimated cost from n to goal,
- f(n) = estimated total cost of path through n to goal

PROGRAM STRUCTURE

The programming language used for this project is Python 3. The program consists of one Class, various methods and a main function .

Variables :

- puzzle -- stores the user input values of initial state from 0-8
- goal -- stores the user input values of goal state from 0-8
- Optimal_path --- finds the Optimal path from sample space after every move.
- expand -- stores the total number of nodes generated.
- state -- retrieving the number of nodes expanded using this variable.

Class :

- Execution –contains a method named heuristic_method which takes an integer input and calculates heuristics i.e., Manhattan distance or Misplaced tiles based on user’s input. It also calculates and prints the number of nodes expanded and generated.

Methods:

- heuristic_method – This method takes an integer input from user for the heuristic function to be used i.e Manhattan distance or Misplaced tiles and calculates the solution from initial state to goal state accordingly.

Functions:

- Manhattan_evaluate -- This function solves the 8-puzzle problem using manhattan heuristic. In this function g(n), h(n) and f(n) are calculated and a priority queue is used to store the node fn and position value as key value pair.

  Here, f(n)= g(n)+h(n), h(n) calculates the sum of number of steps each tile in the current state needs to take to reach to goal state. g(n) is the step cost of each step made.

- Misplaced_evaluate -- This function solves the 8-puzzle problem using Misplaced Tiles heuristic. In this function g(n), h(n) and f(n) are calculated and a priority queue is used to store the node fn and position value as key value pair.
  Here, f(n)= g(n)+h(n) h(n) calculates the number of misplaced tiles in current state as compared to goal state. g(n) is the path cost for each step made.

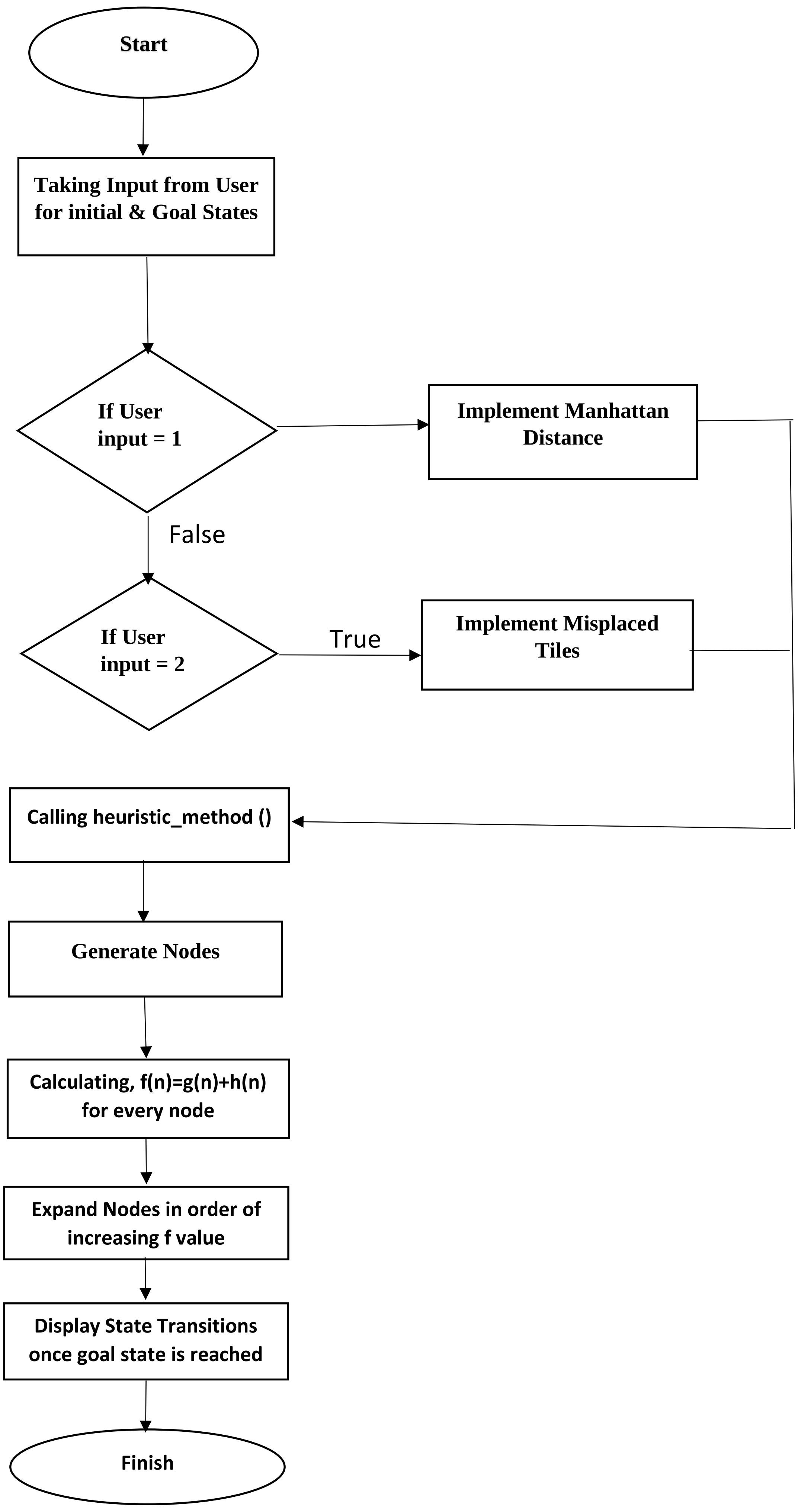
- Optimal_solution -- This function takes all the states as input and traverses through each of the states to find the best solution and returns the best optimal solution.

- manhattan_distance -- This function calculates the Manhattan distance of each element in the initial matrix by calculating the absolute difference between the initial state and the goal state.

- misplaced_tiles – This function calculates the misplaced tiles of each element in the initial matrix by simply calculating the number of tiles that differs from the goal position.

- main – This function takes user inputs for both the initial state and goal state respectively and then calls the heuristic method in Execution class such that it calculates heuristics based on user's input.

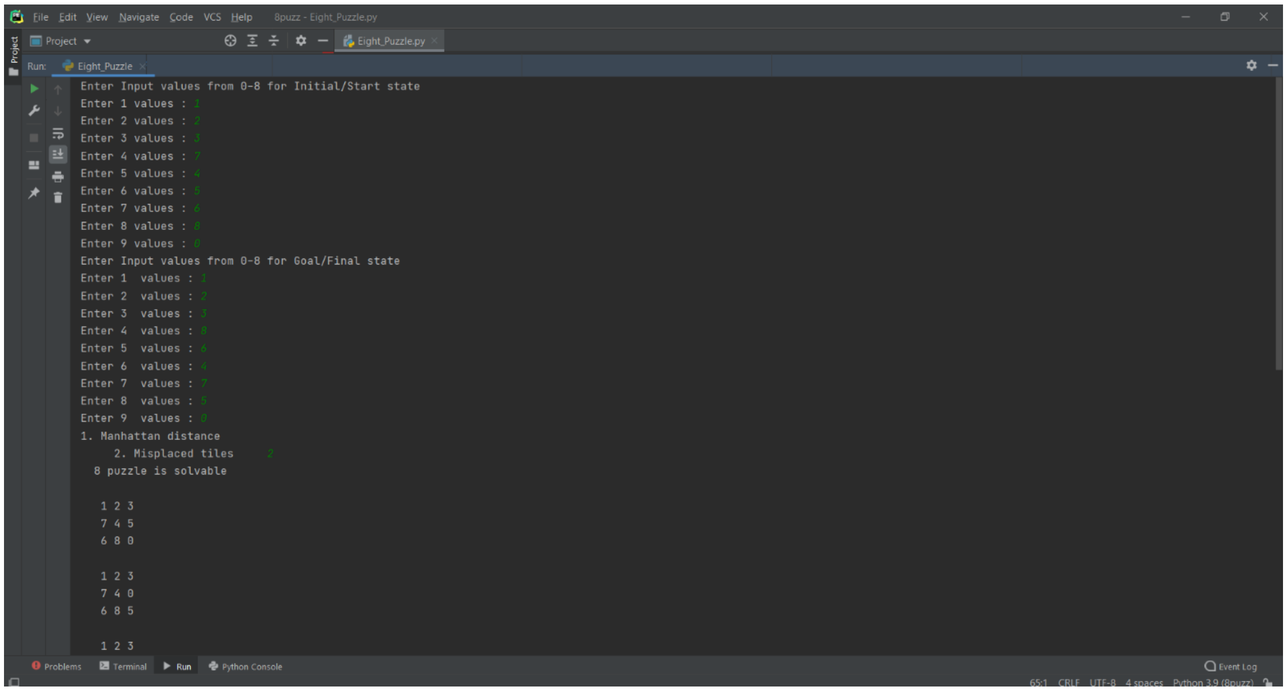
Program Flow – The flow chart in the next page illustrates the diagrammatic execution of the workflow of 8puzzle using A* Star Search algorithm.


Start
Taking Input from User for initial & Goal States
If User input = 1
Implement Manhattan Distance
False
If User input = 2
True
Implement Misplaced Tiles
Calling heuristic_method ()
Generate Nodes
Calculating, f(n)=g(n)+h(n) for every node
Expand Nodes in order of increasing f value
Display State Transitions once goal state is reached
Finish

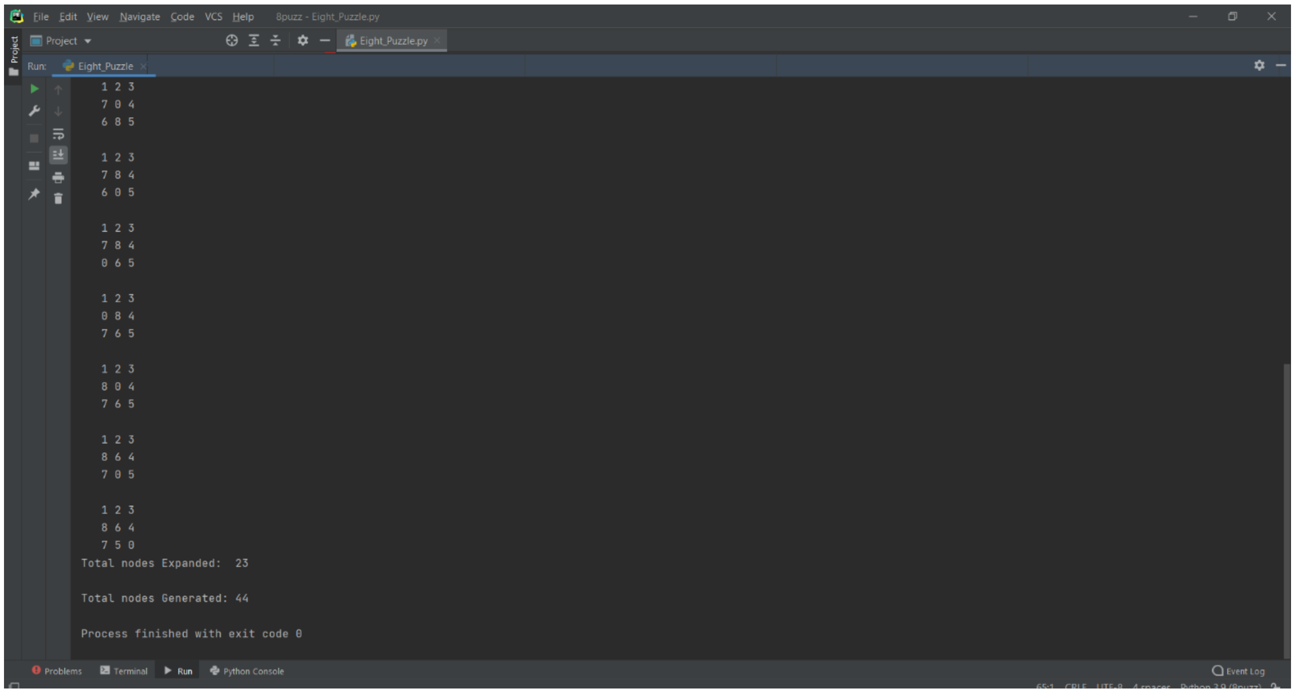
EXECUTION RESULTS:

Example 1
Initial state: 1 2 3 7 4 5 6 8 0
Goal State: 1 2 3 8 6 4 7 5 0

Misplaced Tiles:

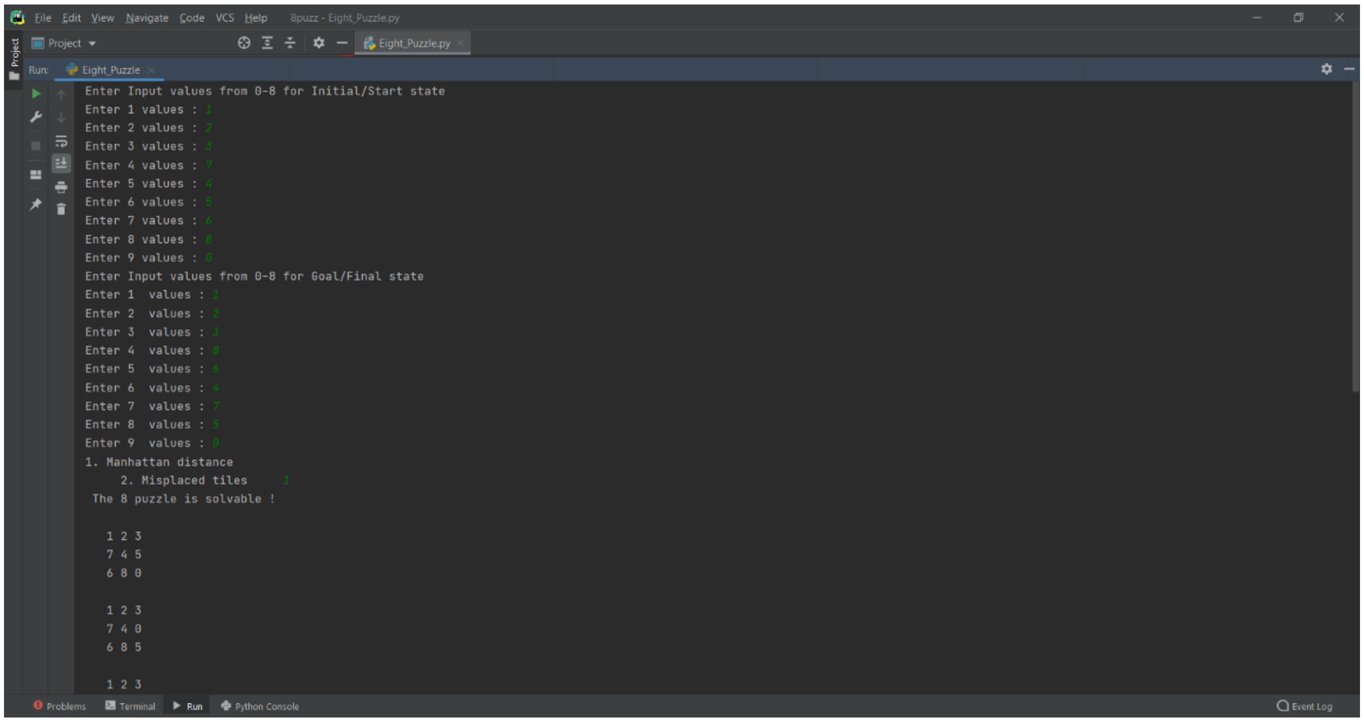


```
Enter Input values from 0-8 for Initial/Start state
Enter 1 values : 1
Enter 2 values : 2
Enter 3 values : 3
Enter 4 values : 7
Enter 5 values : 4
Enter 6 values : 5
Enter 7 values : 6
Enter 8 values : 8
Enter 9 values : 0
Enter Input values from 0-8 for Goal/Final state
Enter 1  values : 1
Enter 2  values : 2
Enter 3  values : 3
Enter 4  values : 8
Enter 5  values : 6
Enter 6  values : 4
Enter 7  values : 7
Enter 8  values : 5
Enter 9  values : 0
1. Manhattan distance
     2. Misplaced tiles     2
  8 puzzle is solvable

   1 2 3
   7 4 5
   6 8 0

   1 2 3
   7 4 0
   6 8 5

   1 2 3
```

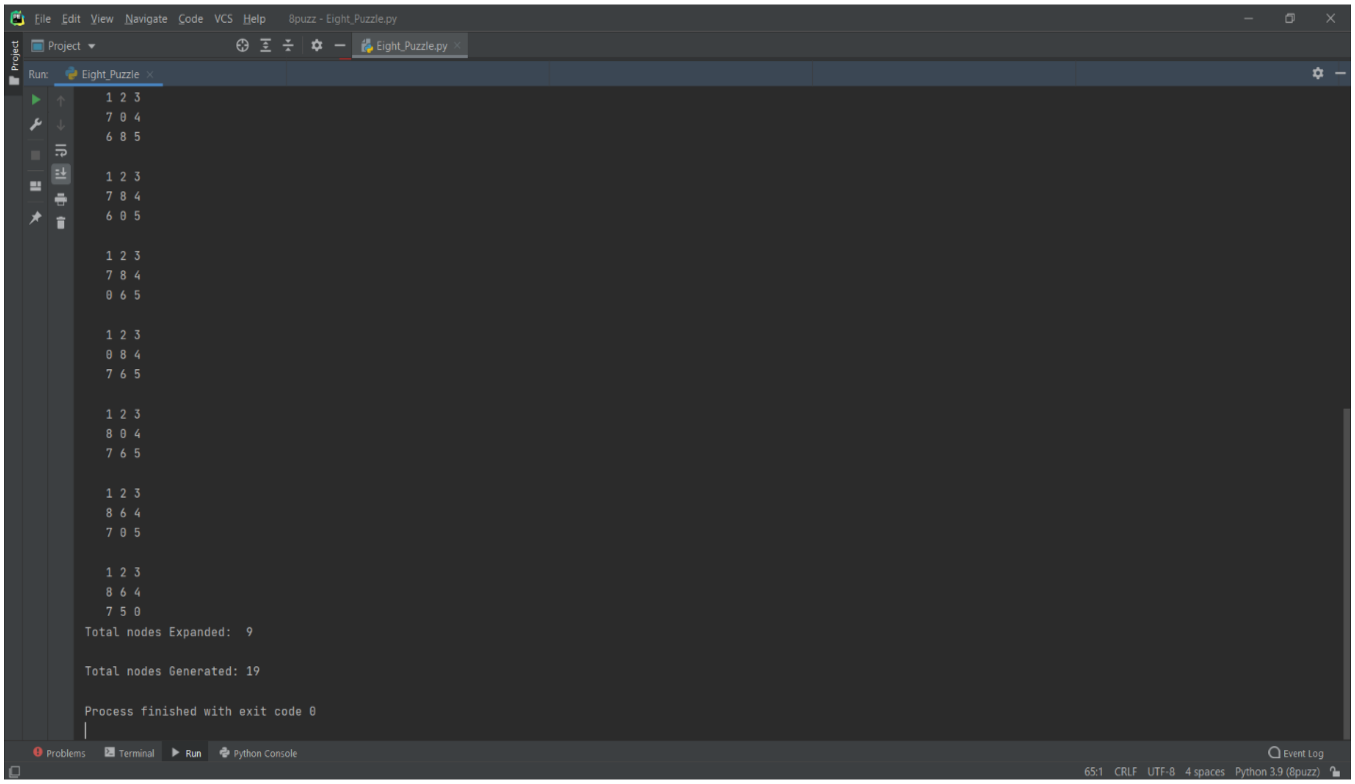


```
   1 2 3
   7 0 4
   6 8 5

   1 2 3
   7 8 4
   6 0 5

   1 2 3
   7 8 4
   0 6 5

   1 2 3
   0 8 4
   7 6 5

   1 2 3
   8 0 4
   7 6 5

   1 2 3
   8 6 4
   7 0 5

   1 2 3
   8 6 4
   7 5 0
Total nodes Expanded:  23

Total nodes Generated: 44

Process finished with exit code 0
```

Manhattan Distance:



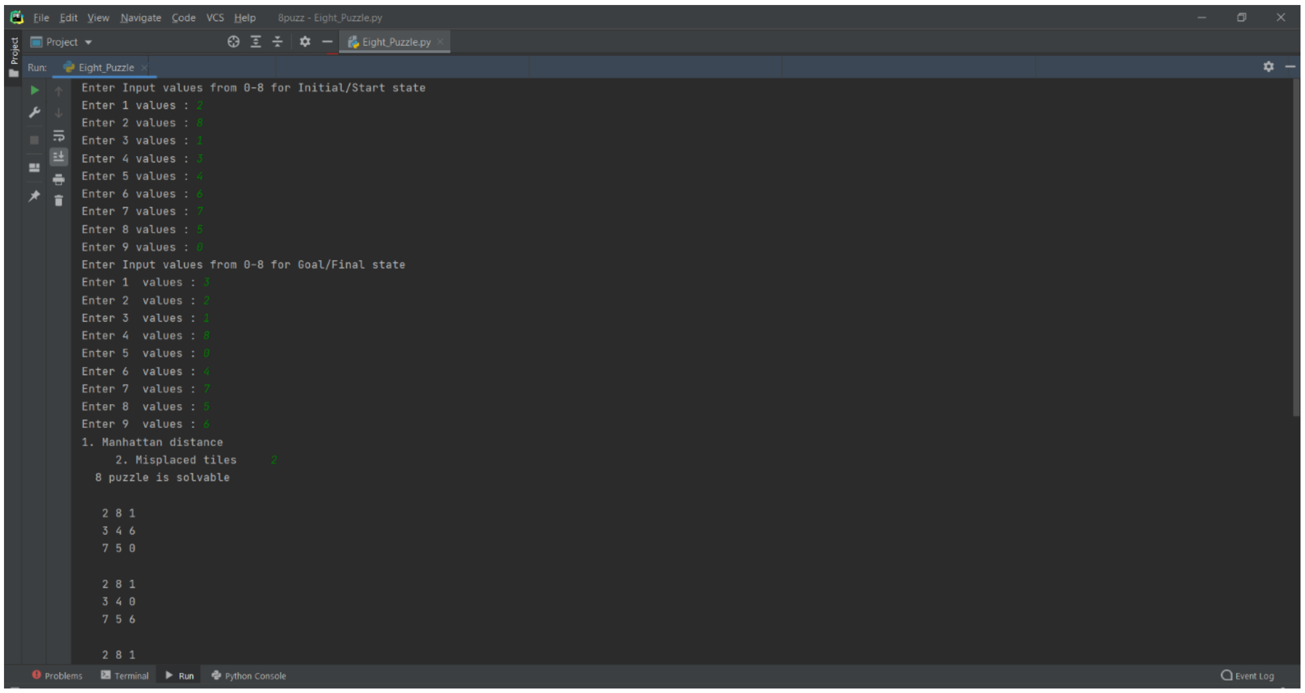
```
Enter Input values from 0-8 for Initial/Start state
Enter 1 values : 1
Enter 2 values : 2
Enter 3 values : 3
Enter 4 values : 7
Enter 5 values : 4
Enter 6 values : 5
Enter 7 values : 6
Enter 8 values : 8
Enter 9 values : 0
Enter Input values from 0-8 for Goal/Final state
Enter 1  values : 1
Enter 2  values : 2
Enter 3  values : 3
Enter 4  values : 8
Enter 5  values : 6
Enter 6  values : 4
Enter 7  values : 7
Enter 8  values : 5
Enter 9  values : 0
1. Manhattan distance
     2. Misplaced tiles     1
 The 8 puzzle is solvable !

   1 2 3
   7 4 5
   6 8 0

   1 2 3
   7 4 0
   6 8 5

   1 2 3
```



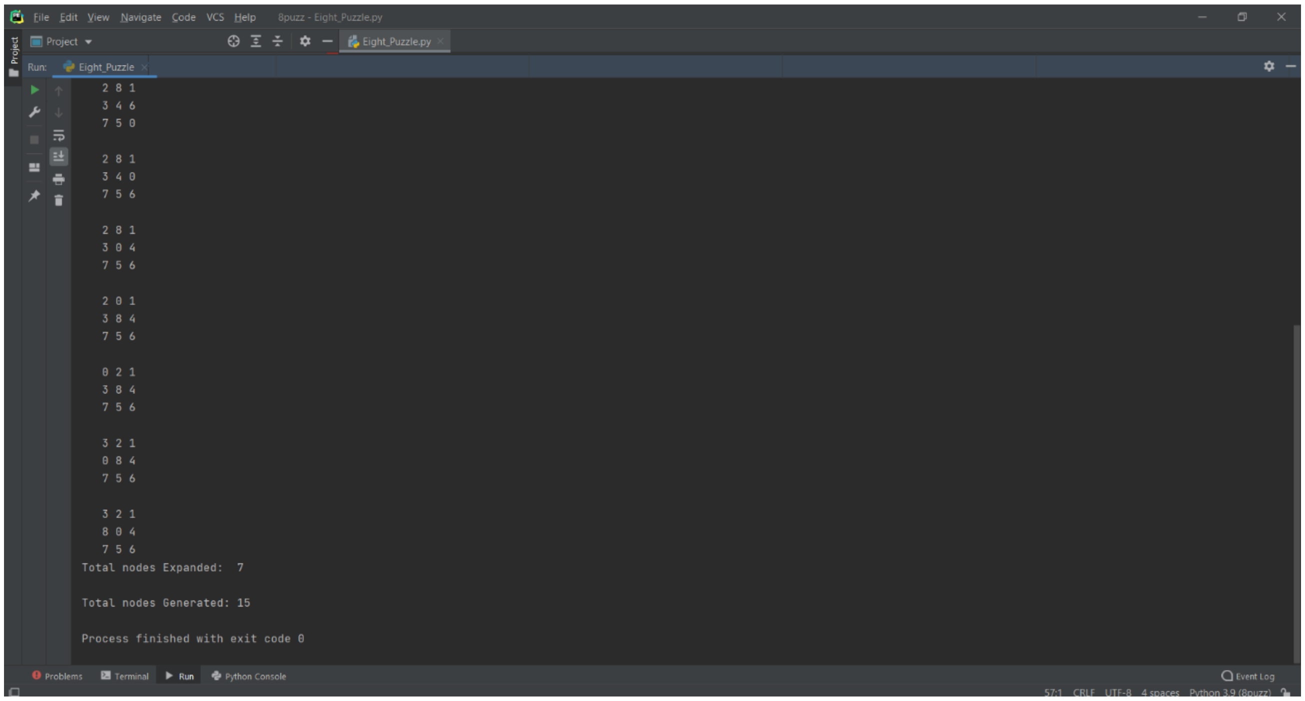
```
   1 2 3
   7 0 4
   6 8 5

   1 2 3
   7 8 4
   6 0 5

   1 2 3
   7 8 4
   0 6 5

   1 2 3
   0 8 4
   7 6 5

   1 2 3
   8 0 4
   7 6 5

   1 2 3
   8 6 4
   7 0 5

   1 2 3
   8 6 4
   7 5 0
Total nodes Expanded:  9

Total nodes Generated: 19

Process finished with exit code 0
```

Example 2
Initial state: 2 8 1 3 4 6 7 5 0
Goal State: 3 2 1 8 0 4 7 5 6

Misplaced Tiles:

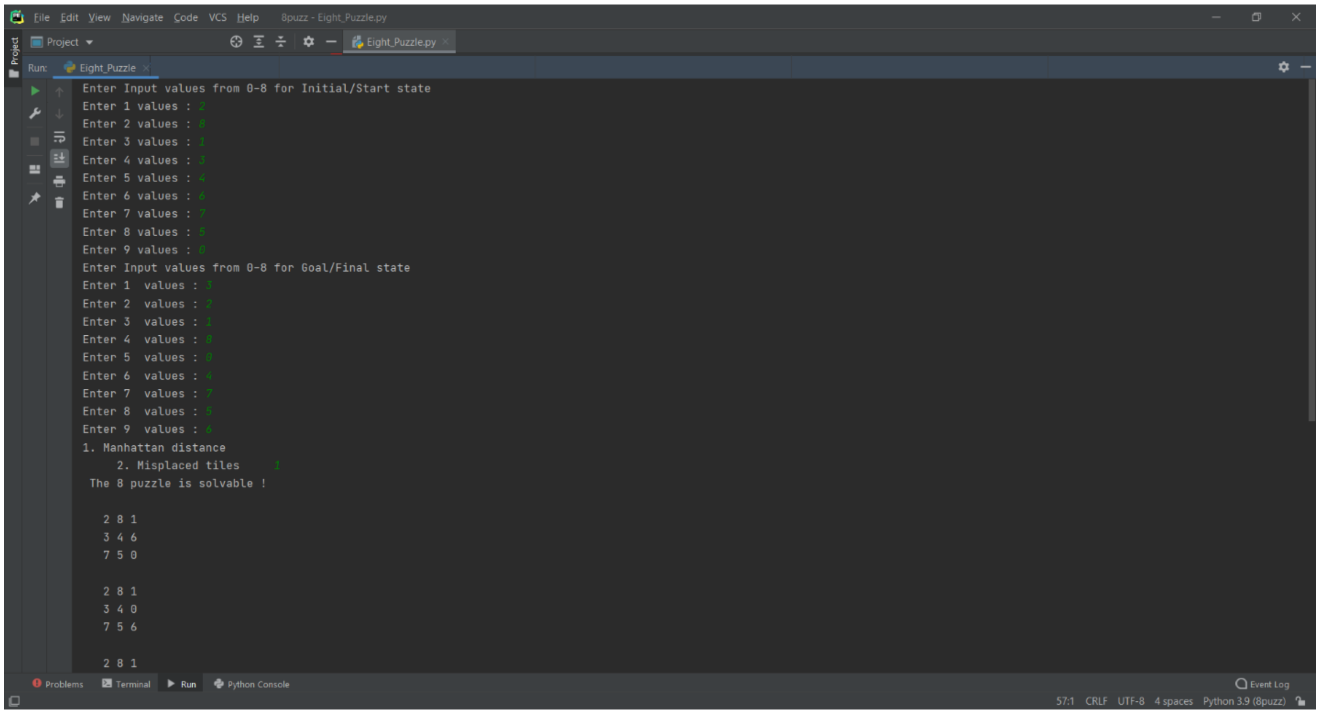


```
Enter Input values from 0-8 for Initial/Start state
Enter 1 values : 2
Enter 2 values : 8
Enter 3 values : 1
Enter 4 values : 3
Enter 5 values : 4
Enter 6 values : 6
Enter 7 values : 7
Enter 8 values : 5
Enter 9 values : 0
Enter Input values from 0-8 for Goal/Final state
Enter 1  values : 3
Enter 2  values : 2
Enter 3  values : 1
Enter 4  values : 8
Enter 5  values : 0
Enter 6  values : 4
Enter 7  values : 7
Enter 8  values : 5
Enter 9  values : 6
1. Manhattan distance
     2. Misplaced tiles      2
  8 puzzle is solvable

   2 8 1
   3 4 6
   7 5 0

   2 8 1
   3 4 0
   7 5 6

   2 8 1
```

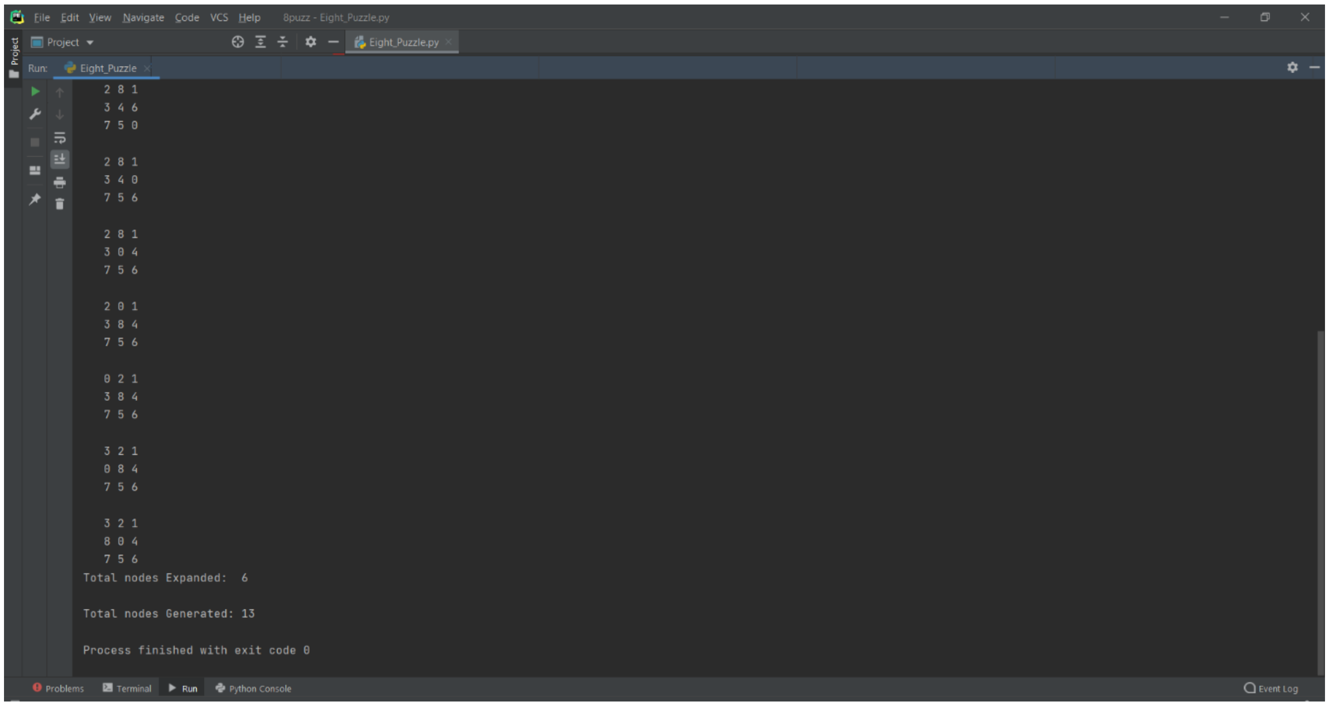


```
   2 8 1
   3 4 6
   7 5 0

   2 8 1
   3 4 0
   7 5 6

   2 8 1
   3 0 4
   7 5 6

   2 0 1
   3 8 4
   7 5 6

   0 2 1
   3 8 4
   7 5 6

   3 2 1
   0 8 4
   7 5 6

   3 2 1
   8 0 4
   7 5 6
Total nodes Expanded:  7

Total nodes Generated: 15

Process finished with exit code 0
```

Manhattan Distance:



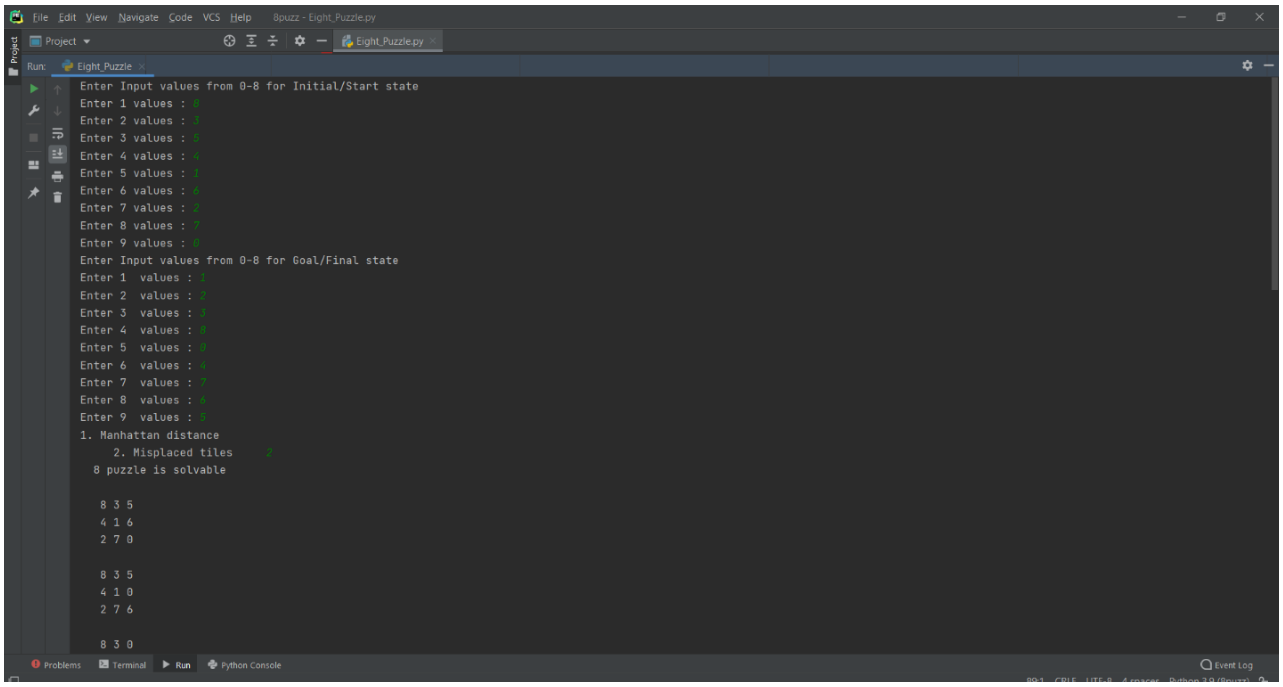
```
Enter Input values from 0-8 for Initial/Start state
Enter 1 values : 2
Enter 2 values : 8
Enter 3 values : 1
Enter 4 values : 3
Enter 5 values : 4
Enter 6 values : 6
Enter 7 values : 7
Enter 8 values : 5
Enter 9 values : 0
Enter Input values from 0-8 for Goal/Final state
Enter 1  values : 3
Enter 2  values : 2
Enter 3  values : 1
Enter 4  values : 8
Enter 5  values : 0
Enter 6  values : 4
Enter 7  values : 7
Enter 8  values : 5
Enter 9  values : 6
1. Manhattan distance
     2. Misplaced tiles      1
 The 8 puzzle is solvable !

   2 8 1
   3 4 6
   7 5 0

   2 8 1
   3 4 0
   7 5 6

   2 8 1
```



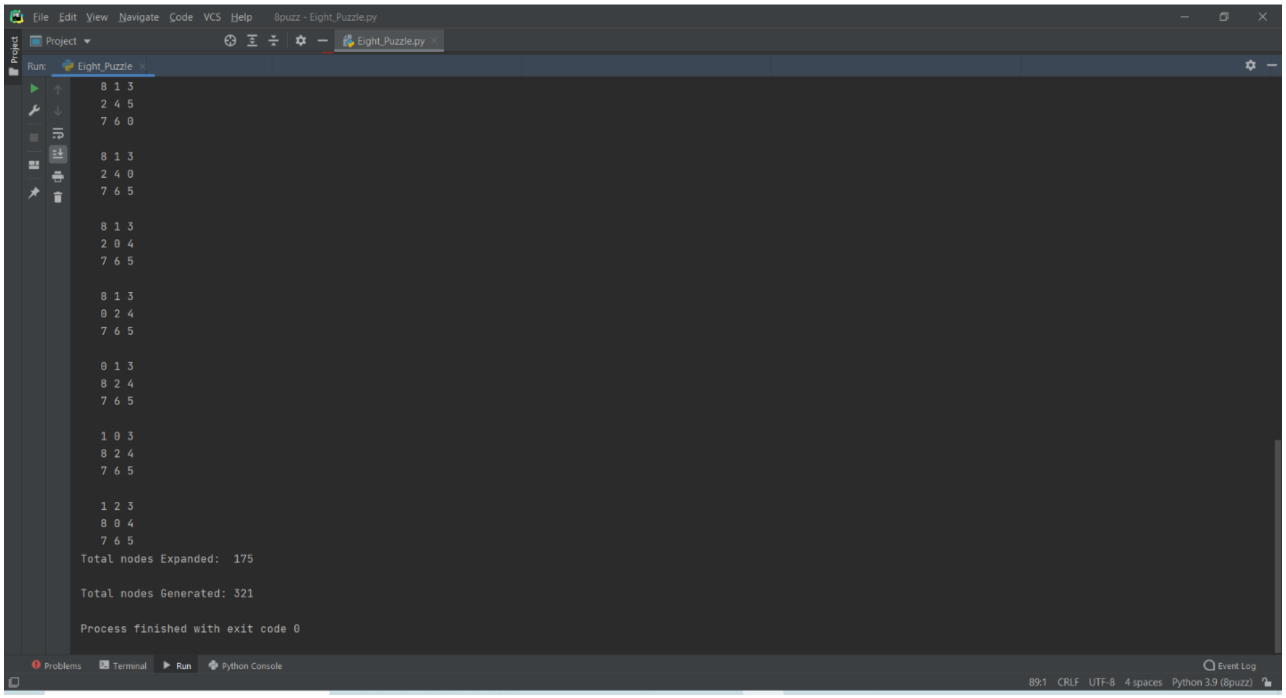
```
   2 8 1
   3 4 6
   7 5 0

   2 8 1
   3 4 0
   7 5 6

   2 8 1
   3 0 4
   7 5 6

   2 0 1
   3 8 4
   7 5 6

   0 2 1
   3 8 4
   7 5 6

   3 2 1
   0 8 4
   7 5 6

   3 2 1
   8 0 4
   7 5 6
Total nodes Expanded:  6

Total nodes Generated: 13

Process finished with exit code 0
```

Example 3:
Initial State: 8 3 5 4 1 6 2 7 0
Final State: 1 2 3 8 0 4 7 6 5

Misplaced Tiles:

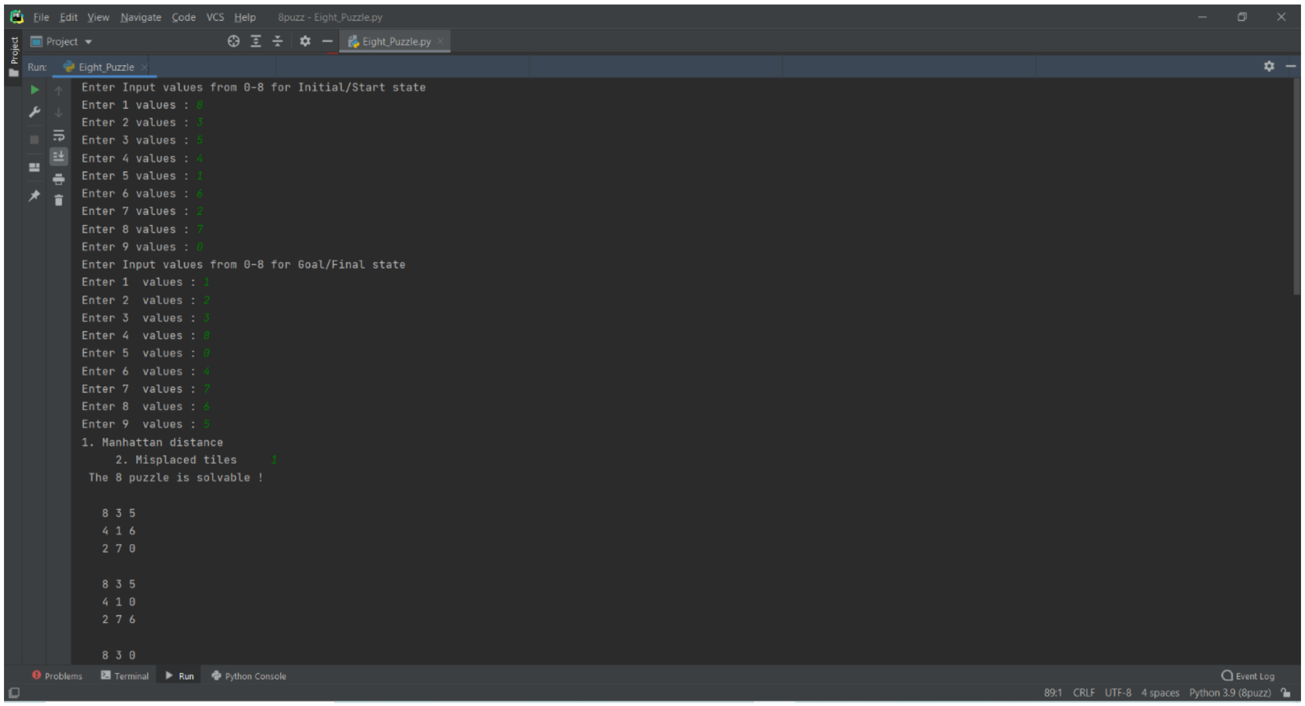


```
Enter Input values from 0-8 for Initial/Start state
Enter 1 values : 8
Enter 2 values : 3
Enter 3 values : 5
Enter 4 values : 4
Enter 5 values : 1
Enter 6 values : 6
Enter 7 values : 2
Enter 8 values : 7
Enter 9 values : 0
Enter Input values from 0-8 for Goal/Final state
Enter 1  values : 1
Enter 2  values : 2
Enter 3  values : 3
Enter 4  values : 8
Enter 5  values : 0
Enter 6  values : 4
Enter 7  values : 7
Enter 8  values : 6
Enter 9  values : 5
1. Manhattan distance
     2. Misplaced tiles     2
  8 puzzle is solvable


   8 3 5
   4 1 6
   2 7 0

   8 3 5
   4 1 0
   2 7 6

   8 3 0
```

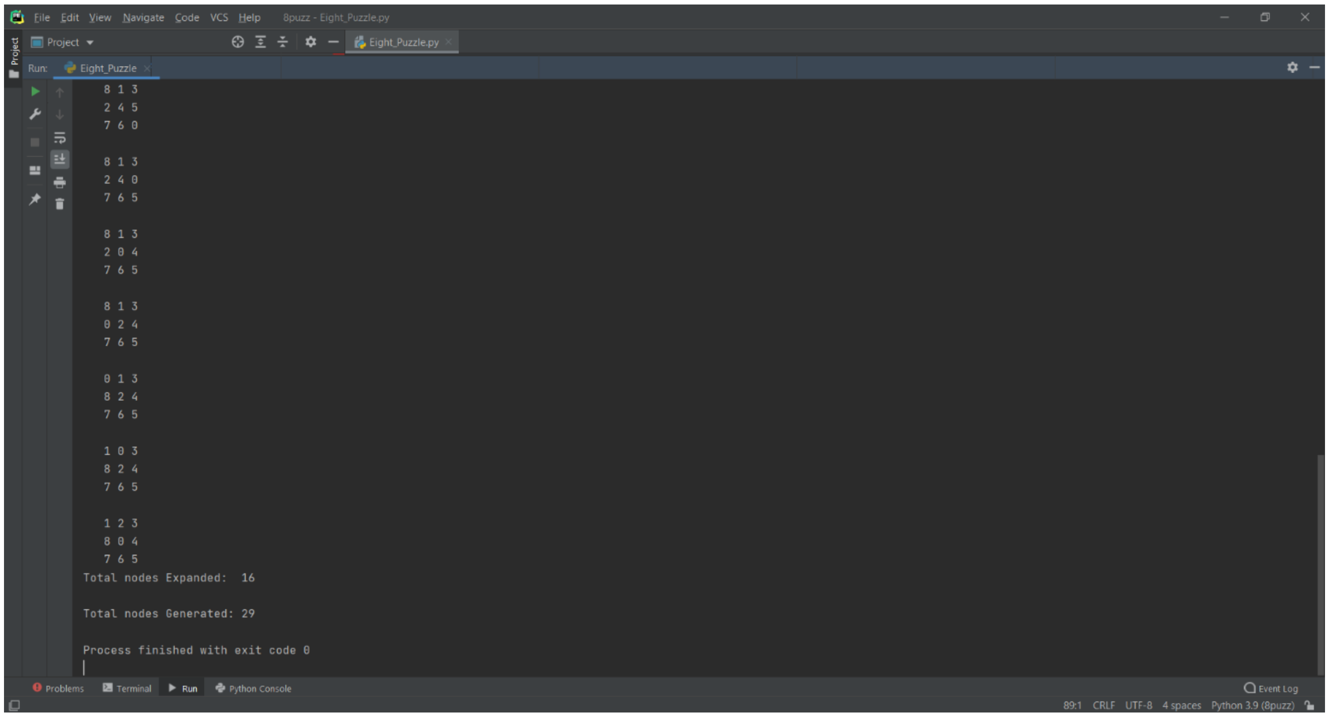


```
   8 1 3
   2 4 5
   7 6 0

   8 1 3
   2 4 0
   7 6 5

   8 1 3
   2 0 4
   7 6 5

   8 1 3
   0 2 4
   7 6 5

   0 1 3
   8 2 4
   7 6 5

   1 0 3
   8 2 4
   7 6 5

   1 2 3
   8 0 4
   7 6 5
Total nodes Expanded:  175

Total nodes Generated: 321

Process finished with exit code 0
```

Manhattan Distance:



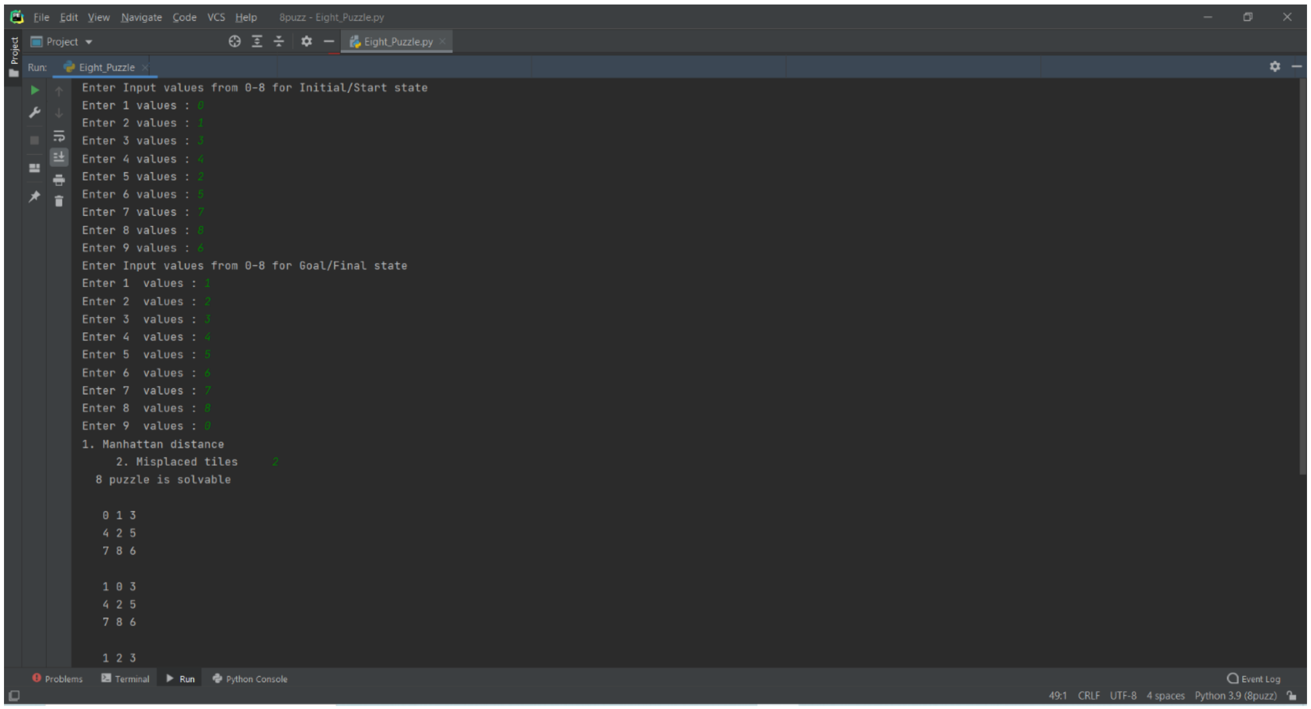
```
Enter Input values from 0-8 for Initial/Start state
Enter 1 values : 8
Enter 2 values : 3
Enter 3 values : 5
Enter 4 values : 4
Enter 5 values : 1
Enter 6 values : 6
Enter 7 values : 2
Enter 8 values : 7
Enter 9 values : 0
Enter Input values from 0-8 for Goal/Final state
Enter 1  values : 1
Enter 2  values : 2
Enter 3  values : 3
Enter 4  values : 8
Enter 5  values : 0
Enter 6  values : 4
Enter 7  values : 7
Enter 8  values : 6
Enter 9  values : 5
1. Manhattan distance
     2. Misplaced tiles     1
 The 8 puzzle is solvable !


   8 3 5
   4 1 6
   2 7 0

   8 3 5
   4 1 0
   2 7 6

   8 3 0
```



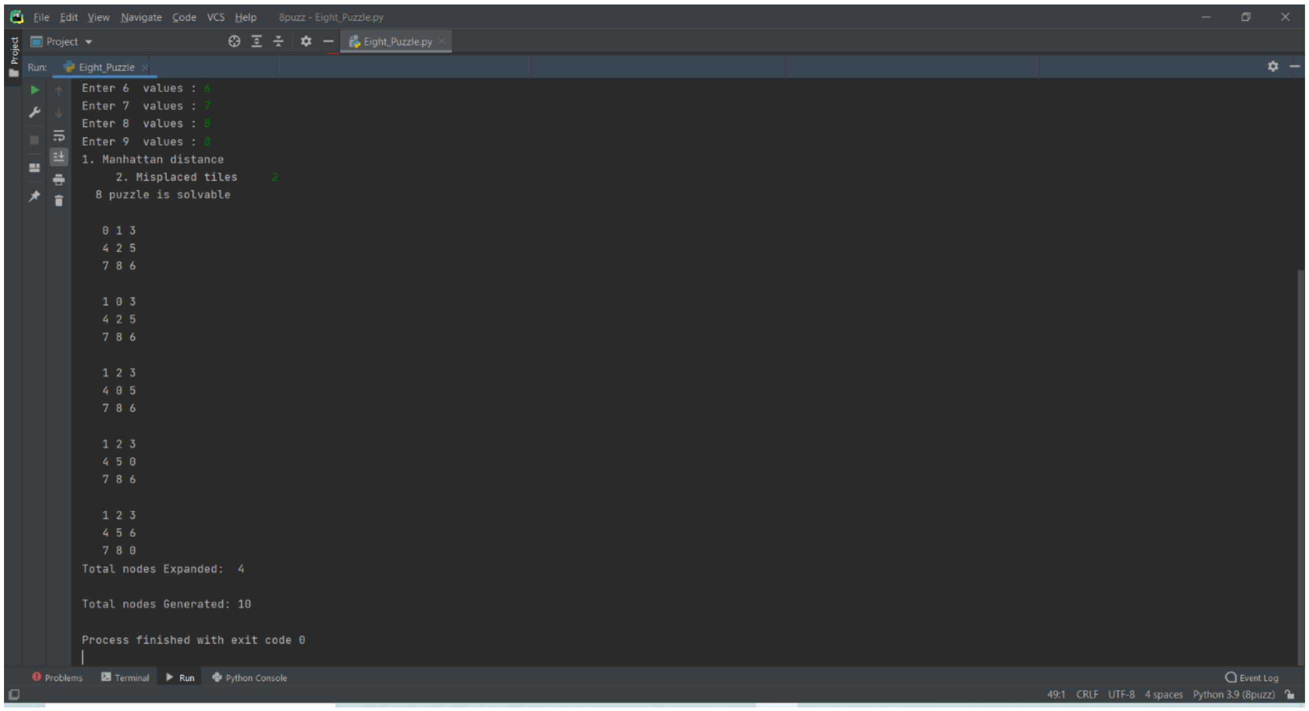
```
   8 1 3
   2 4 5
   7 6 0

   8 1 3
   2 4 0
   7 6 5

   8 1 3
   2 0 4
   7 6 5

   8 1 3
   0 2 4
   7 6 5

   0 1 3
   8 2 4
   7 6 5

   1 0 3
   8 2 4
   7 6 5

   1 2 3
   8 0 4
   7 6 5
Total nodes Expanded:  16

Total nodes Generated: 29

Process finished with exit code 0
```

Example 4:
Initial State: 0 1 3 4 2 5 7 8 6
Final State: 1 2 3 4 5 6 7 8 0

Misplaced Tiles:

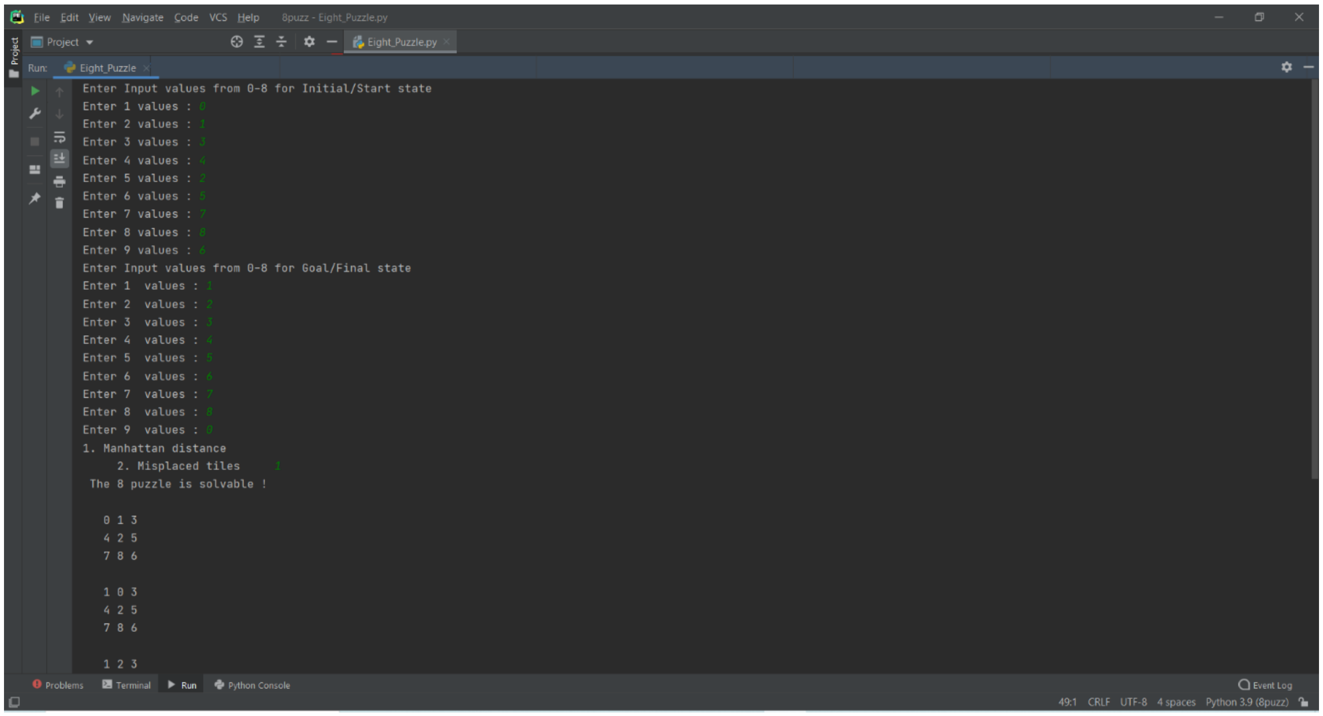


```
Enter Input values from 0-8 for Initial/Start state
Enter 1 values : 0
Enter 2 values : 1
Enter 3 values : 3
Enter 4 values : 4
Enter 5 values : 2
Enter 6 values : 5
Enter 7 values : 7
Enter 8 values : 8
Enter 9 values : 6
Enter Input values from 0-8 for Goal/Final state
Enter 1  values : 1
Enter 2  values : 2
Enter 3  values : 3
Enter 4  values : 4
Enter 5  values : 5
Enter 6  values : 6
Enter 7  values : 7
Enter 8  values : 8
Enter 9  values : 0
1. Manhattan distance
     2. Misplaced tiles     2
  8 puzzle is solvable

   0 1 3
   4 2 5
   7 8 6

   1 0 3
   4 2 5
   7 8 6

   1 2 3
```

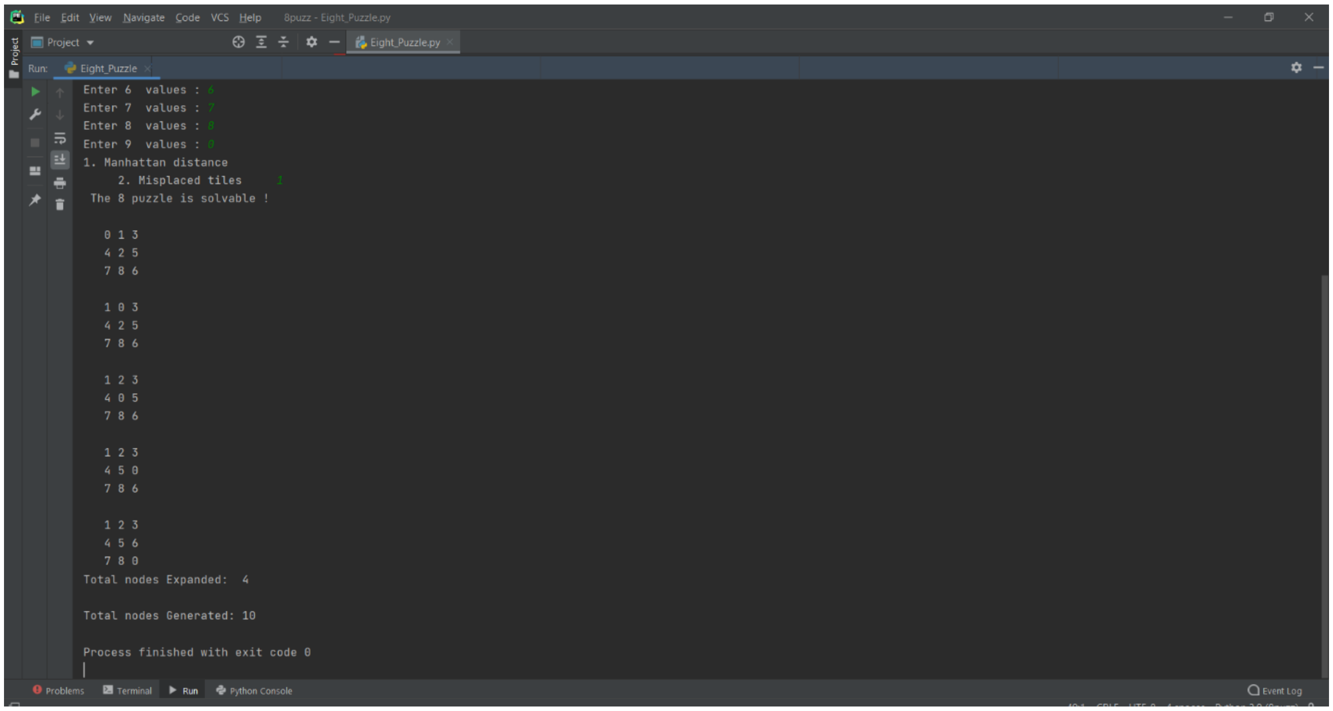


```
Enter 6  values : 6
Enter 7  values : 7
Enter 8  values : 8
Enter 9  values : 0
1. Manhattan distance
     2. Misplaced tiles     2
  8 puzzle is solvable

   0 1 3
   4 2 5
   7 8 6

   1 0 3
   4 2 5
   7 8 6

   1 2 3
   4 0 5
   7 8 6

   1 2 3
   4 5 0
   7 8 6

   1 2 3
   4 5 6
   7 8 0
Total nodes Expanded:  4

Total nodes Generated: 10

Process finished with exit code 0
```

Manhattan Distance:



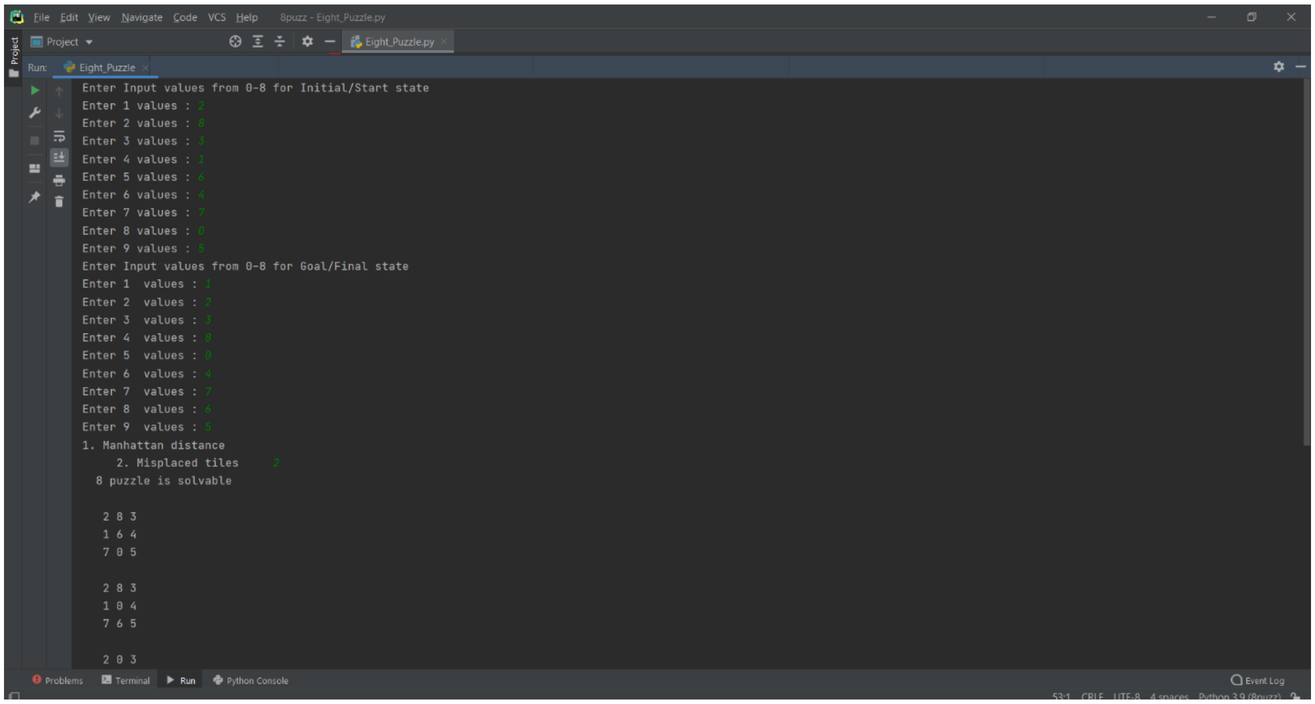
```
Enter Input values from 0-8 for Initial/Start state
Enter 1 values : 0
Enter 2 values : 1
Enter 3 values : 3
Enter 4 values : 4
Enter 5 values : 2
Enter 6 values : 5
Enter 7 values : 7
Enter 8 values : 8
Enter 9 values : 6
Enter Input values from 0-8 for Goal/Final state
Enter 1  values : 1
Enter 2  values : 2
Enter 3  values : 3
Enter 4  values : 4
Enter 5  values : 5
Enter 6  values : 6
Enter 7  values : 7
Enter 8  values : 8
Enter 9  values : 0
1. Manhattan distance
     2. Misplaced tiles     1
 The 8 puzzle is solvable !

   0 1 3
   4 2 5
   7 8 6

   1 0 3
   4 2 5
   7 8 6

   1 2 3
```



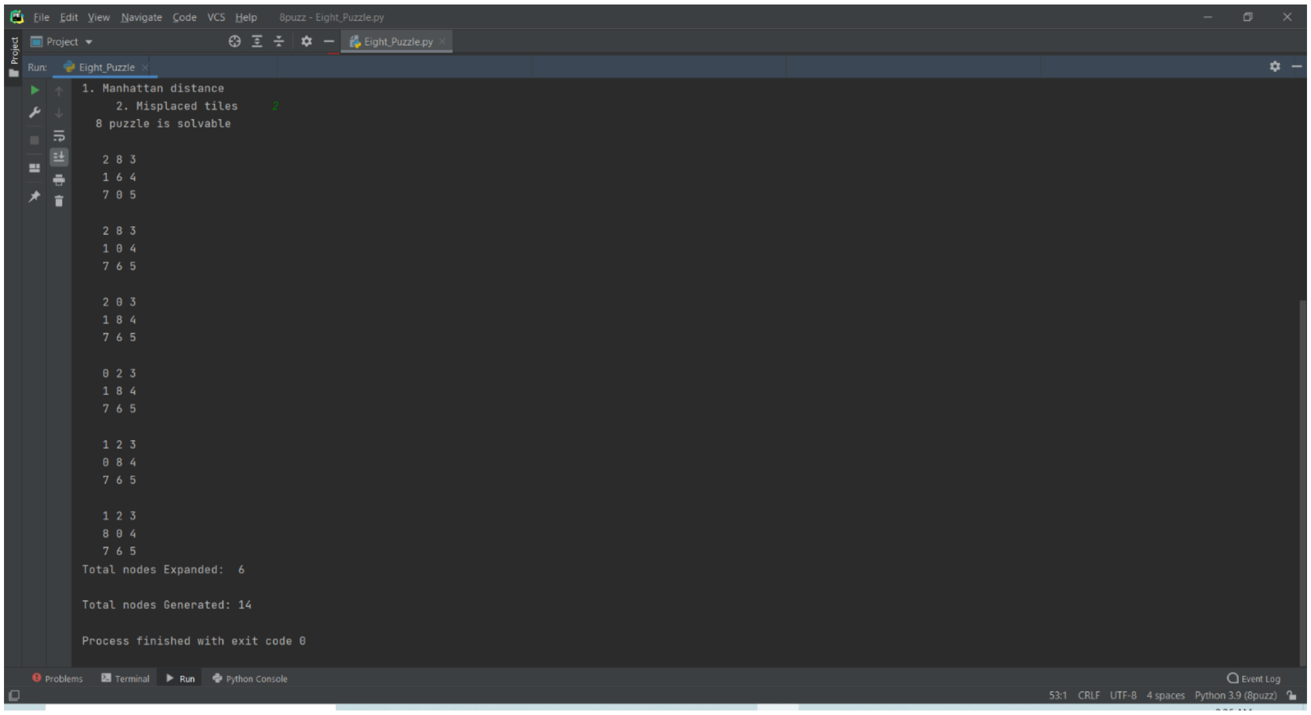
```
Enter 6  values : 6
Enter 7  values : 7
Enter 8  values : 8
Enter 9  values : 0
1. Manhattan distance
     2. Misplaced tiles     1
 The 8 puzzle is solvable !

   0 1 3
   4 2 5
   7 8 6

   1 0 3
   4 2 5
   7 8 6

   1 2 3
   4 0 5
   7 8 6

   1 2 3
   4 5 0
   7 8 6

   1 2 3
   4 5 6
   7 8 0
Total nodes Expanded:  4

Total nodes Generated: 10

Process finished with exit code 0
```

Example 5:
Initial State: 2 8 3 1 6 4 7 0 5
Final State: 1 2 3 8 0 4 7 6 5

Misplaced Tiles:

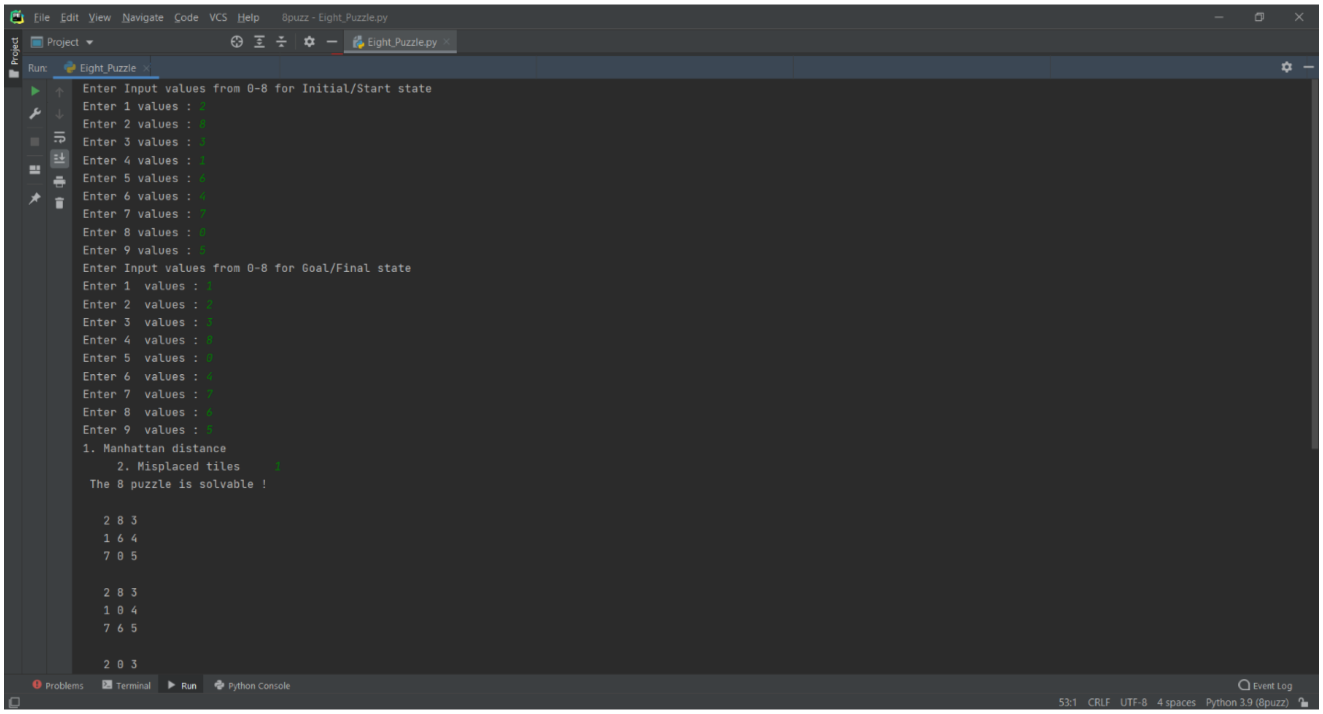

File Edit View Navigate Code VCS Help 8puzz - Eight_Puzzle.py
Project
Eight_Puzzle.py
Run: Eight_Puzzle
Problems Terminal Run Python Console
Event Log

```
Enter Input values from 0-8 for Initial/Start state
Enter 1 values : 2
Enter 2 values : 8
Enter 3 values : 3
Enter 4 values : 1
Enter 5 values : 6
Enter 6 values : 4
Enter 7 values : 7
Enter 8 values : 0
Enter 9 values : 5
Enter Input values from 0-8 for Goal/Final state
Enter 1  values : 1
Enter 2  values : 2
Enter 3  values : 3
Enter 4  values : 8
Enter 5  values : 0
Enter 6  values : 4
Enter 7  values : 7
Enter 8  values : 6
Enter 9  values : 5
1. Manhattan distance
     2. Misplaced tiles     2
  8 puzzle is solvable


   2 8 3
   1 6 4
   7 0 5

   2 8 3
   1 0 4
   7 6 5

   2 0 3
```

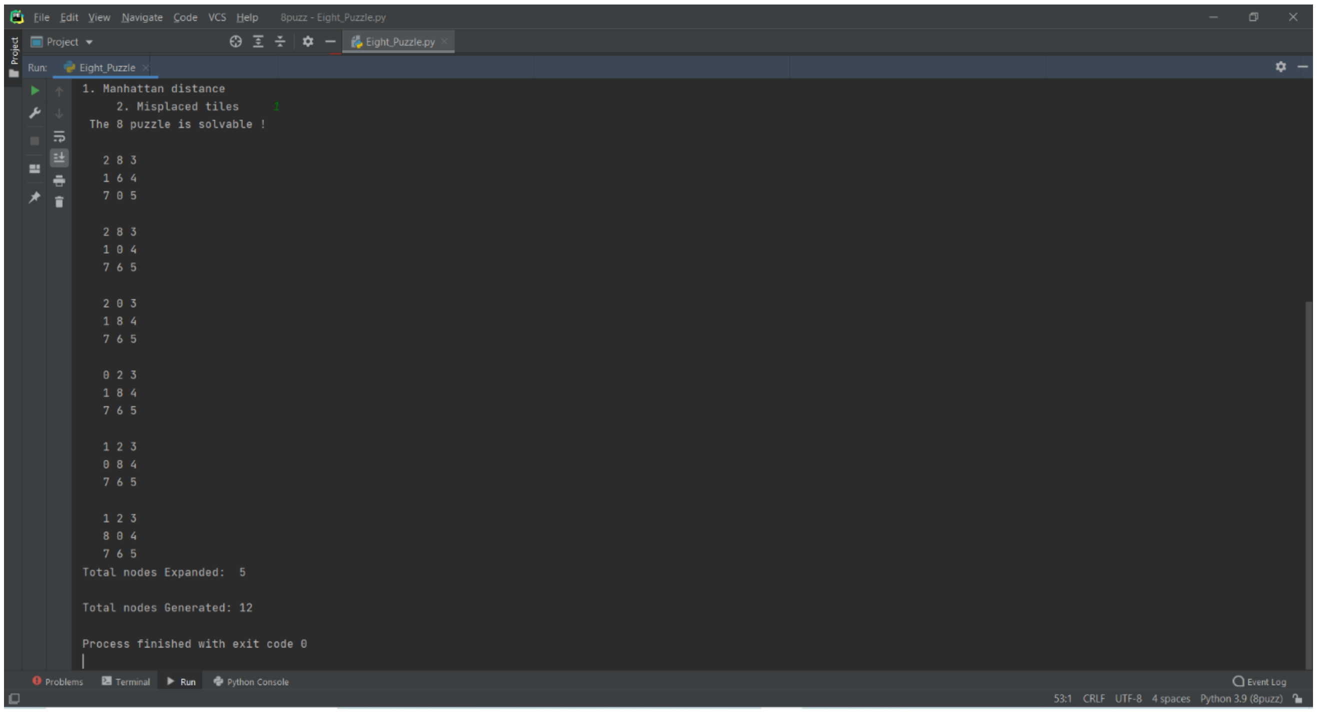

File Edit View Navigate Code VCS Help 8puzz - Eight_Puzzle.py
Project
Eight_Puzzle.py
Run: Eight_Puzzle
Problems Terminal Run Python Console
Event Log
53:1 CRLF UTF-8 4 spaces Python 3.9 (8puzz)

```
1. Manhattan distance
     2. Misplaced tiles     2
  8 puzzle is solvable

   2 8 3
   1 6 4
   7 0 5

   2 8 3
   1 0 4
   7 6 5

   2 0 3
   1 8 4
   7 6 5

   0 2 3
   1 8 4
   7 6 5

   1 2 3
   0 8 4
   7 6 5

   1 2 3
   8 0 4
   7 6 5
Total nodes Expanded:  6

Total nodes Generated: 14

Process finished with exit code 0
```

Manhattan Distance:



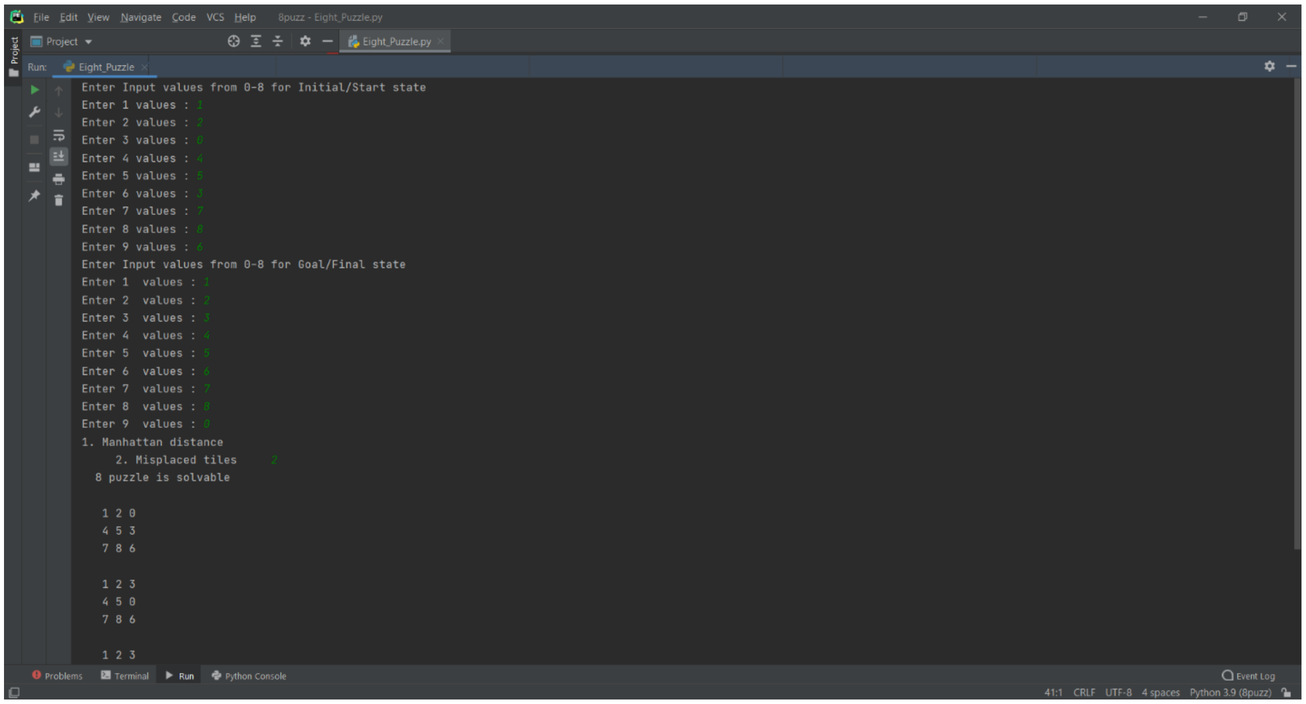
```
Enter Input values from 0-8 for Initial/Start state
Enter 1 values : 2
Enter 2 values : 8
Enter 3 values : 3
Enter 4 values : 1
Enter 5 values : 6
Enter 6 values : 4
Enter 7 values : 7
Enter 8 values : 0
Enter 9 values : 5
Enter Input values from 0-8 for Goal/Final state
Enter 1  values : 1
Enter 2  values : 2
Enter 3  values : 3
Enter 4  values : 8
Enter 5  values : 0
Enter 6  values : 4
Enter 7  values : 7
Enter 8  values : 6
Enter 9  values : 5
1. Manhattan distance
     2. Misplaced tiles     1
 The 8 puzzle is solvable !

   2 8 3
   1 6 4
   7 0 5

   2 8 3
   1 0 4
   7 6 5

   2 0 3
```



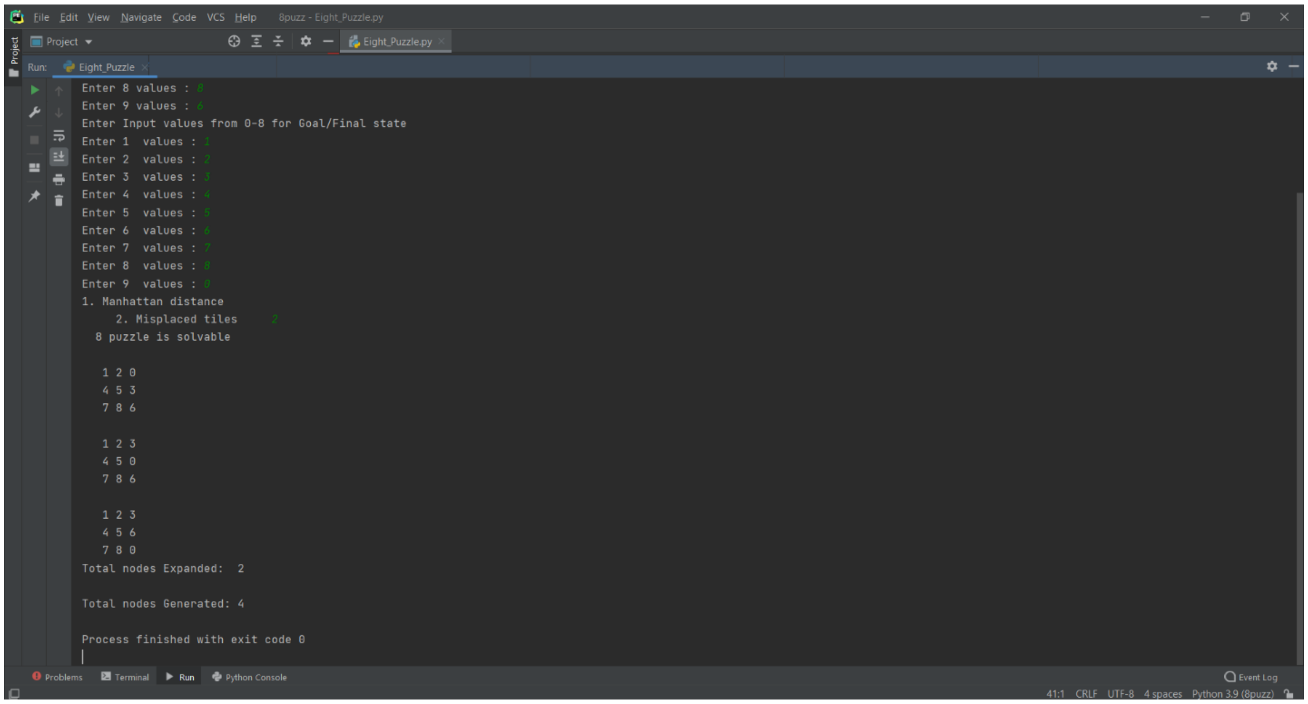
```
1. Manhattan distance
     2. Misplaced tiles     1
 The 8 puzzle is solvable !

   2 8 3
   1 6 4
   7 0 5

   2 8 3
   1 0 4
   7 6 5

   2 0 3
   1 8 4
   7 6 5

   0 2 3
   1 8 4
   7 6 5

   1 2 3
   0 8 4
   7 6 5

   1 2 3
   8 0 4
   7 6 5
Total nodes Expanded:  5

Total nodes Generated: 12

Process finished with exit code 0
```

Example 6:
Initial State: 1 2 0 4 5 3 7 8 6
Final State: 1 2 3 4 5 6 7 8 0

Misplaced Tiles:

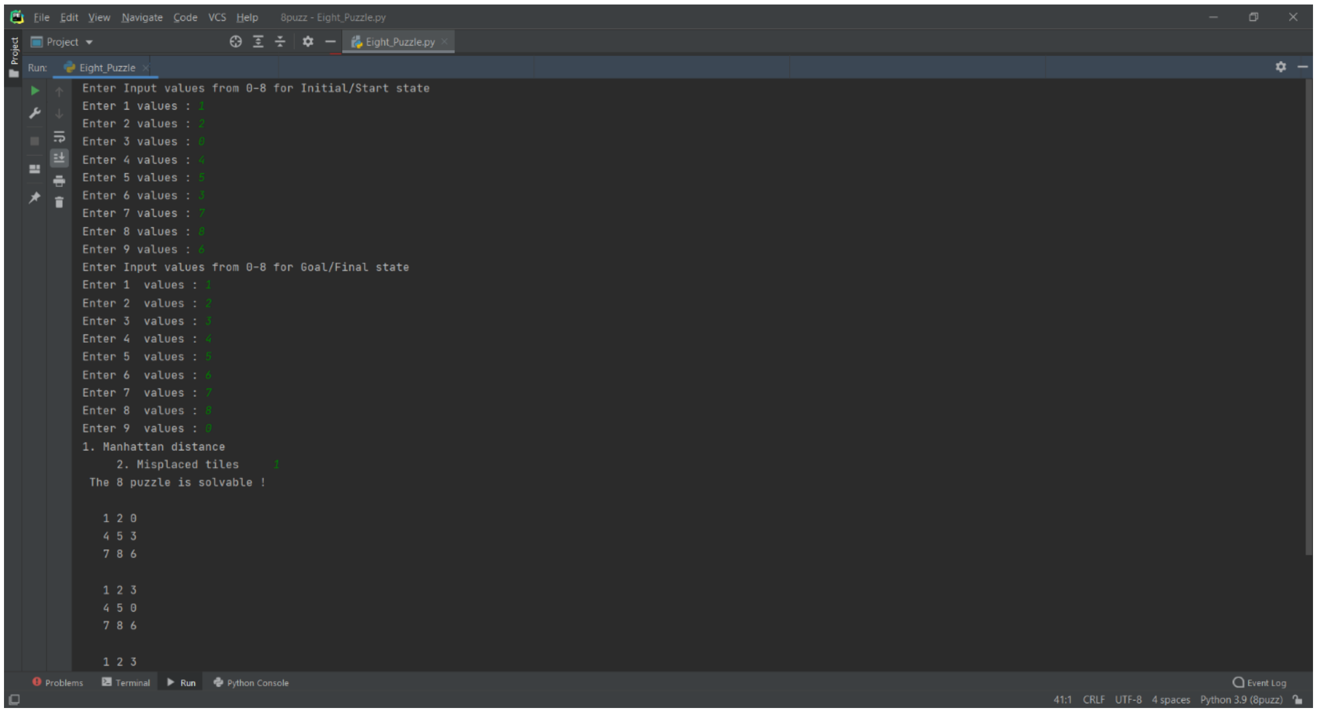


```
Enter Input values from 0-8 for Initial/Start state
Enter 1 values : 1
Enter 2 values : 2
Enter 3 values : 0
Enter 4 values : 4
Enter 5 values : 5
Enter 6 values : 3
Enter 7 values : 7
Enter 8 values : 8
Enter 9 values : 6
Enter Input values from 0-8 for Goal/Final state
Enter 1  values : 1
Enter 2  values : 2
Enter 3  values : 3
Enter 4  values : 4
Enter 5  values : 5
Enter 6  values : 6
Enter 7  values : 7
Enter 8  values : 8
Enter 9  values : 0
1. Manhattan distance
     2. Misplaced tiles     2
  8 puzzle is solvable


   1 2 0
   4 5 3
   7 8 6


   1 2 3
   4 5 0
   7 8 6


   1 2 3
```

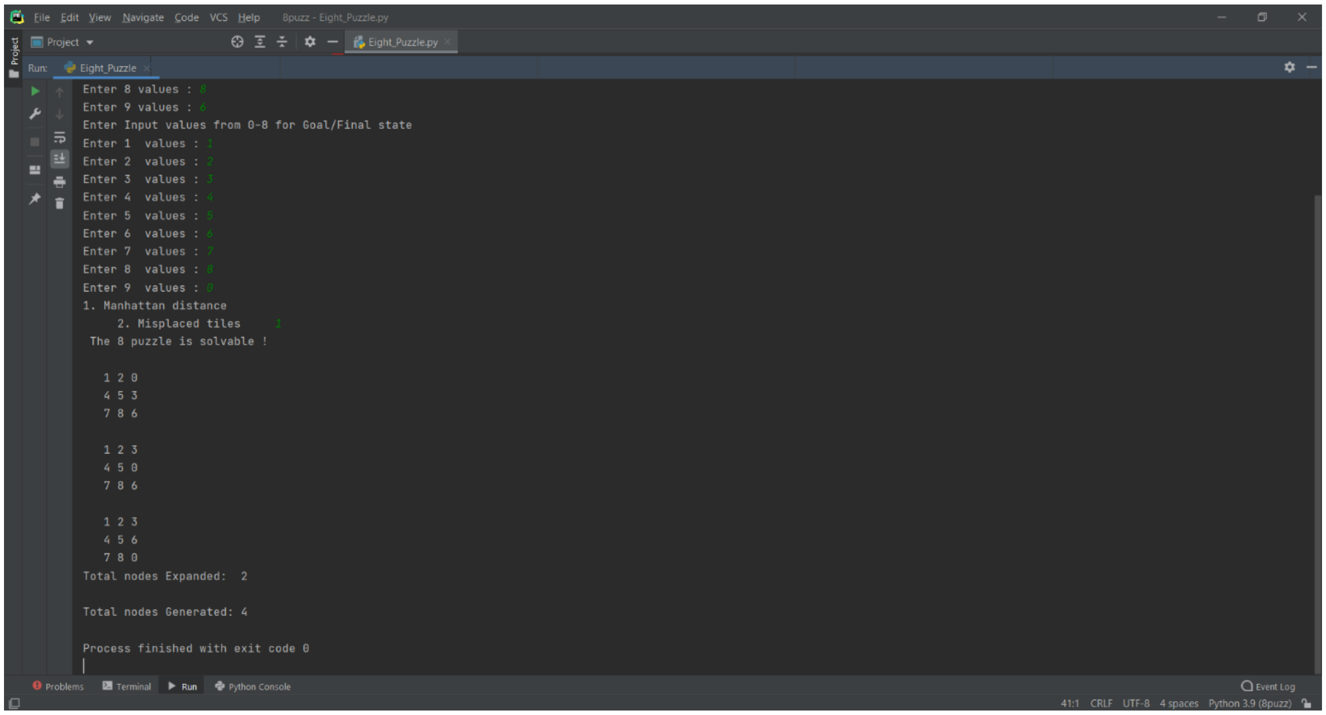


```
Enter 8 values : 8
Enter 9 values : 6
Enter Input values from 0-8 for Goal/Final state
Enter 1  values : 1
Enter 2  values : 2
Enter 3  values : 3
Enter 4  values : 4
Enter 5  values : 5
Enter 6  values : 6
Enter 7  values : 7
Enter 8  values : 8
Enter 9  values : 0
1. Manhattan distance
     2. Misplaced tiles     2
  8 puzzle is solvable


   1 2 0
   4 5 3
   7 8 6


   1 2 3
   4 5 0
   7 8 6


   1 2 3
   4 5 6
   7 8 0
Total nodes Expanded:  2

Total nodes Generated: 4

Process finished with exit code 0
```

Manhattan Distance:



```
Enter Input values from 0-8 for Initial/Start state
Enter 1 values : 1
Enter 2 values : 2
Enter 3 values : 0
Enter 4 values : 4
Enter 5 values : 5
Enter 6 values : 3
Enter 7 values : 7
Enter 8 values : 8
Enter 9 values : 6
Enter Input values from 0-8 for Goal/Final state
Enter 1  values : 1
Enter 2  values : 2
Enter 3  values : 3
Enter 4  values : 4
Enter 5  values : 5
Enter 6  values : 6
Enter 7  values : 7
Enter 8  values : 8
Enter 9  values : 0
1. Manhattan distance
     2. Misplaced tiles     1
 The 8 puzzle is solvable !

   1 2 0
   4 5 3
   7 8 6

   1 2 3
   4 5 0
   7 8 6

   1 2 3
```



```
Enter 8 values : 8
Enter 9 values : 6
Enter Input values from 0-8 for Goal/Final state
Enter 1  values : 1
Enter 2  values : 2
Enter 3  values : 3
Enter 4  values : 4
Enter 5  values : 5
Enter 6  values : 6
Enter 7  values : 7
Enter 8  values : 8
Enter 9  values : 0
1. Manhattan distance
     2. Misplaced tiles     1
 The 8 puzzle is solvable !

   1 2 0
   4 5 3
   7 8 6

   1 2 3
   4 5 0
   7 8 6

   1 2 3
   4 5 6
   7 8 0
Total nodes Expanded:  2

Total nodes Generated: 4

Process finished with exit code 0
```

Summary Table

Initial State	Goal State	Misplaced Tiles	Manhattan Distance
1 2 3 7 4 5 6 8 0	1 2 3 8 6 4 7 5 0	Nodes Expanded: 23 Nodes Generated: 44	Nodes Expanded: 9 Nodes Generated: 19
2 8 1 3 4 6 7 5 0	3 2 1 8 0 4 7 5 6	Nodes Expanded: 7 Nodes Generated: 15	Nodes Expanded: 4 Nodes Generated: 13
8 3 5 4 1 6 2 7 0	1 2 3 8 0 4 7 6 5	Nodes Expanded: 175 Nodes Generated: 321	Nodes Expanded: 16 Nodes Generated: 29
0 1 3 4 2 5 7 8 6	1 2 3 4 5 6 7 8 0	Nodes Expanded: 4 Nodes Generated: 10	Nodes Expanded: 4 Nodes Generated: 10
2 8 3 1 6 4 7 0 5	1 2 3 8 0 4 7 6 5	Nodes Expanded: 6 Nodes Generated: 14	Nodes Expanded: 5 Nodes Generated: 12
1 2 0 4 5 3 7 8 6	1 2 3 4 5 6 7 8 0	Nodes Expanded: 2 Nodes Generated: 4	Nodes Expanded: 2 Nodes Generated: 4

Conclusion

The 8 puzzle algorithm is solved using A* algorithm which uses heuristics such as manhattan distance and misplaced tile.

Reference

1. http://www.sfu.ca/~tjd/310summer2019/a1.html
2. https://stackoverflow.com/
3. https://medium.com/@faramira.sg/solving-the-8-puzzle-problem-using-a-star-algorithm-5cf1db4cdb0f
4. http://www.github.com
5. https://numpy.org/doc/stable/user/basics.io.html