Name: Kenil Chovatiya Student id: 105163073

Out put of A-Star for graph A and for Graph B is:

Text

Description automatically generated

Text

Description automatically generated

Time (in microseconds) difference or cost as compared to bfs, dfs and ids is:

Table

Description automatically generated

Que 2

Part -1

Here is the code for part 1 and it generates the random puzzle and in place of \* I have 0:

import random

class Randompuzzle:

    def \_\_init\_\_(self, number):

        self.CreatePuzzle(number)

    def CreatePuzzle(self,number):

        MAX = 8

        MIN = 0

        USABLENUMBERS   = list(range(0, 9))

        USEDNUMBERS = []

        board = [-1] \* 9

        while(self.checksolvable(board)):

            i = 0

            USEDNUMBERS = []

            board = [-1] \* 9

            while(len(USEDNUMBERS) < len(USABLENUMBERS)):

                rand = random.randint(MIN,MAX)

                if(rand not in USEDNUMBERS):

                    USEDNUMBERS.append(rand)

                    board[i] = rand

                    i = i + 1

        self.printboard(board)

    def checksolvable (self, board):

        inversion = 0

        for i in (range(0, 9)):

            for j in range(i+1, 9):

                # make sure not to count 0 tile

                if (board [i] != 0 and board[j] != 0 and board[i] >= board[j]):

                    inversion = inversion+1

        return inversion%2 == 0

    def printboard (self, board):

        for i in range(9):

            print(board[i], end=' ')

            if (i+1)%3==0:

                print()

test1= Randompuzzle(0)

output:

Text

Description automatically generated

Part: 2

code for that is in puzzel.py file.

Output will be shown as:

Text

Description automatically generated

Output for 3rd choise:

Type 1 to bigin puzzle,or -1 to quit.

1

Choice of algorithms to use for A\*:

1. The misplaced tile heuristic

2: The Manhattan distance heuristic

3. The max of the misplaced tile heuristic and the Manhattan distance heurist

Enter: 3

The best node to expand with g(n) = 0 and h(n) = 1 is...

1 2 3

4 6

7 5 8

Expanding this node...

The best node to expand with g(n) = 1 and h(n) = 1 is...

1 2 3

4 6

7 5 8

Expanding this node...

The best node to expand with g(n) = 1 and h(n) = 1 is...

2 3

1 4 6

7 5 8

Expanding this node...

The best node to expand with g(n) = 1 and h(n) = 1 is...

1 2 3

7 4 6

5 8

Expanding this node...

The best node to expand with g(n) = 2 and h(n) = 1 is...

1 2 3

4 6

7 5 8

Expanding this node...

The best node to expand with g(n) = 2 and h(n) = 1 is...

1 2 3

4 6

7 5 8

Expanding this node...

The best node to expand with g(n) = 2 and h(n) = 1 is...

1 3

4 2 6

7 5 8

Expanding this node...

The best node to expand with g(n) = 2 and h(n) = 1 is...

1 2 3

4 5 6

7 8

Expanding this node...

The best node to expand with g(n) = 2 and h(n) = 1 is...

2 3

1 4 6

7 5 8

Expanding this node...

The best node to expand with g(n) = 2 and h(n) = 1 is...

1 2 3

4 6

7 5 8

Expanding this node...

The best node to expand with g(n) = 2 and h(n) = 1 is...

1 2 3

7 4 6

5 8

Expanding this node...

The best node to expand with g(n) = 2 and h(n) = 1 is...

1 2 3

4 6

7 5 8

Expanding this node...

The best node to expand with g(n) = 3 and h(n) = 1 is...

1 2 3

4 6

7 5 8

Expanding this node...

The best node to expand with g(n) = 3 and h(n) = 1 is...

2 3

1 4 6

7 5 8

Expanding this node...

The best node to expand with g(n) = 3 and h(n) = 1 is...

1 2 3

7 4 6

5 8

Expanding this node...

The best node to expand with g(n) = 3 and h(n) = 1 is...

1 2 3

4 6

7 5 8

Expanding this node...

The best node to expand with g(n) = 3 and h(n) = 1 is...

1 2

4 6 3

7 5 8

Expanding this node...

The best node to expand with g(n) = 3 and h(n) = 1 is...

1 2 3

4 6 8

7 5

Expanding this node...

The best node to expand with g(n) = 3 and h(n) = 1 is...

1 3

4 2 6

7 5 8

Expanding this node...

The best node to expand with g(n) = 3 and h(n) = 1 is...

1 3

4 2 6

7 5 8

Expanding this node...

The best node to expand with g(n) = 3 and h(n) = 1 is...

1 2 3

4 6

7 5 8

Expanding this node...

The best node to expand with g(n) = 3 and h(n) = 1 is...

1 2 3

4 5 6

7 8

Expanding this node...

The best node to expand with g(n) = 3 and h(n) = 1 is...

1 2 3

4 5 6

7 8

Expanding this node...

Solution found!!

1 2 3

4 5 6

7 8

Expanded a total of 61 nodes

Maximum number of nodes in the queue was 40

The depth of the goal node was 3

Time taken is: 226848

Part 3:

Time difference and cost for puzzle for 3 different algorithms:

Table

Description automatically generated

Best suitable algorithm is Manhattan distance heuristic taking less time and less cost to solve the puzzle.