

8 puzzle A\*  
Code:-

Date: / /  
Page:

*open*  
*closed*

### 4 8 puzzle problem using A\* algorithm:

- (1) Node (data, level): initialize node with puzzle state & level
- (2) def puzzle():
  - accept():  $\leftarrow$  accept start state & goal state
  - f(start, goal)  $\leftarrow$  calculate  $f = h + g$
  - process()  $\leftarrow$  process A\* algorithm
- (3) Heuristic
 

$h(\text{start}, \text{goal}) =$  No of Misplaced tiles.
- (4) A\* Algorithm:
  - 1) initialize start & goal state & add to open list
  - 2) loop until goal state comes
    - $\rightarrow$  select node have lowest from open list
    - $\rightarrow$  Display state & check goal
    - $\rightarrow$  generate child nodes
    - $\rightarrow$  add current node which is done with expanding to closed list. add the ones we don't want to continue.
    - $\rightarrow$  Explore list
    - $\rightarrow$  Display nodes.

1	2	3
4	5	6
7	8	

goal state

1	2	4
3	6	
7	8	5

1	2	4
3	6	6
7	8	5

$h =$

$h = 4$

$h =$

*Pro 8/12*

class Node:

def \_\_init\_\_(self, data, level, fval):

self.data = data

self.level = level

self.fval = fval

def generate\_child(self):

x, y = self.find(self.data, '=')

val-list = [(x+1, y, -1), (x, y+1), (x-1, y), (x, y-1)]

children = []

for i in val-list:

child = self.shuffle(self.data, x, y, i[0], i[1])

if child is not None:

child\_node = Node(child, self.level+1,

children.append(child\_node)

return children

def shuffle(self, orig, x1, y1, x2, y2):

if x

output:-

```
main.py
88     print("Enter the goal state matrix \n")
89     goal = self.accept()
90
91     start = Node(start,0,0)
92     start.fval = self.f(start,goal)
93     """ Put the start node in the open list"""

Enter the start state matrix

1 2 3
4 5 6
_ 7 8
Enter the goal state matrix

1 2 3
4 5 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 _

...Program finished with exit code 0
Press ENTER to exit console.
```