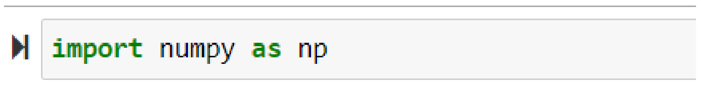
8-puzzle problem
Report

Name: Muhammad Bilal

ID: F2018266166

Section: V3

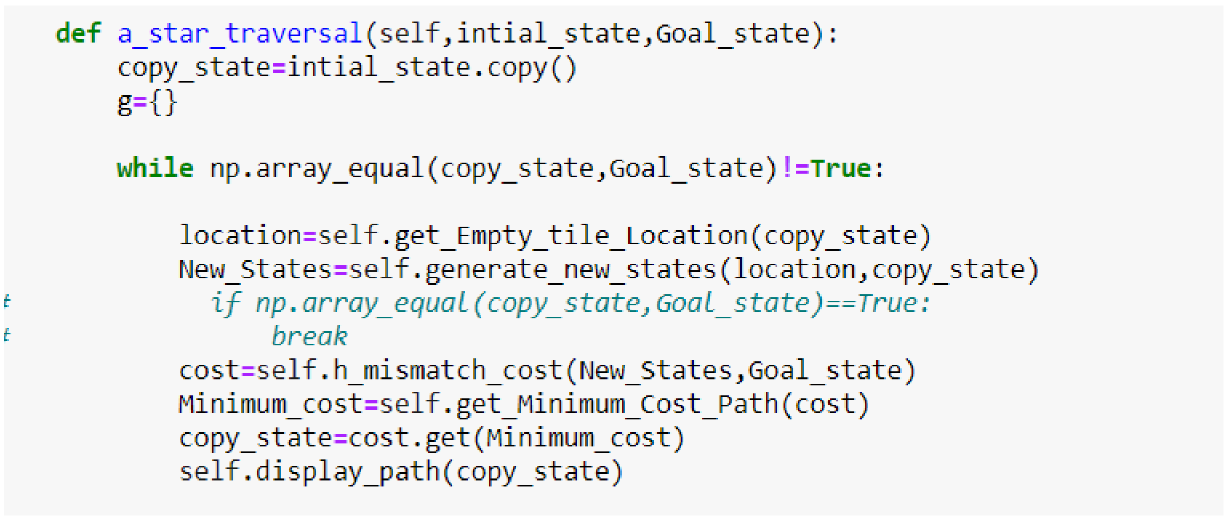


```python
import numpy as np
```

First of we are importing numpy to create 8-Puzzle .

After that we create Puzzle class.

After that we are define some function on puzzle class.



```python
def a_star_traversal(self,intial_state,Goal_state):
    copy_state=intial_state.copy()
    g={}

    while np.array_equal(copy_state,Goal_state)!=True:

        location=self.get_Empty_tile_Location(copy_state)
        New_States=self.generate_new_states(location,copy_state)
          if np.array_equal(copy_state,Goal_state)==True:
              break
        cost=self.h_mismatch_cost(New_States,Goal_state)
        Minimum_cost=self.get_Minimum_Cost_Path(cost)
        copy_state=cost.get(Minimum_cost)
        self.display_path(copy_state)
```

Now to see a_star_traversal(self,initial,Goal_state) method after class puzzle object initialization we call this method to start execution puzzle problem this method will give us solution of 8 puzzle problem. We initial state of our problem and goal state pass in parameter after we

copy_state=intial_state.copy()   we create copy of our state.

g={}   and create empty dictionary

you see that we start loop that will goes to till we will come to goal state.

location=self.get_Empty_tile_Location(copy_state)

This will get empty tile location

New_States=self.generate_new_states(location,copy_state)

This method will generate new states of copy state means that empty tile kaha kaha move kr skti hai we have get all possible senerios it get one or two or more np arrays

cost=self.h_mismatch_cost(New_States,Goal_state)

Then will pass new state and goal state to the h_mismatch_cost method to find cost of new states it return all np array cost inform of the dictinary.

Minimum_cost=self.get_Minimum_Cost_Path(cost)

After that we will find minimum cost of newstates by passing cost parameter into get_Minimum_Cost_Path method it will return minimum cost

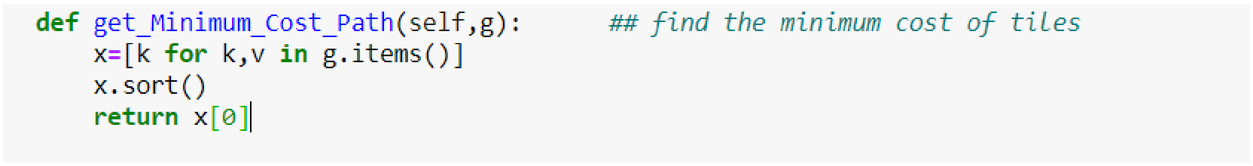
copy_state=cost.get(Minimum_cost)

we know we store cost in the form of dictionary so after finding the minimum cost we pass into the cost.get method to get that minimum cost array from new states array

self.display_path(copy_state)

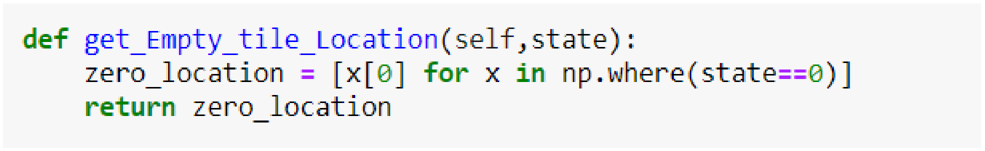
so after that we print that minimum state this loop continues running till we reach our goal.

Now I’m explain that method which used in a_star_traversal.



```
def get_Minimum_Cost_Path(self,g):      ## find the minimum cost of tiles
    x=[k for k,v in g.items()]
    x.sort()
    return x[0]
```

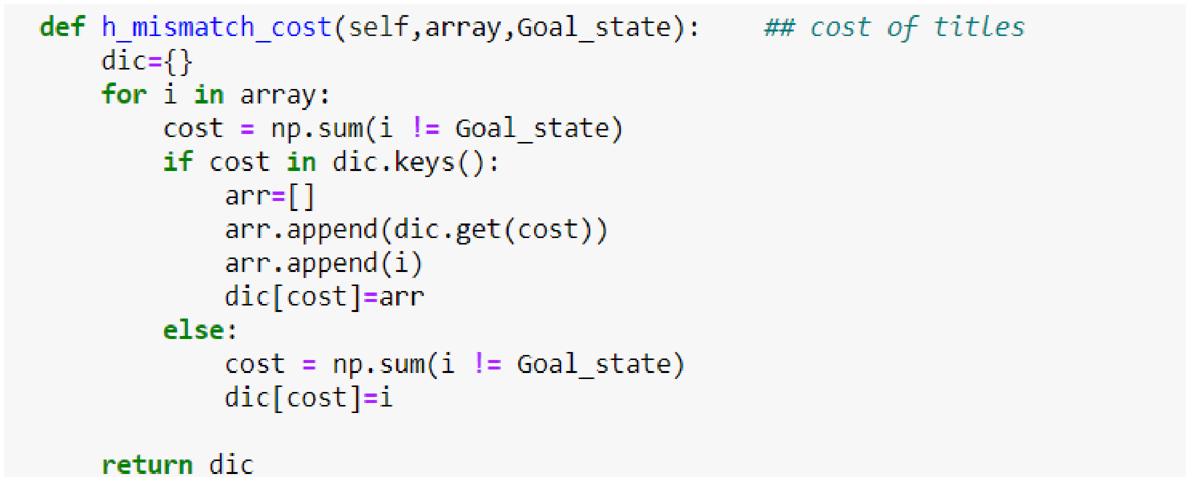
Simply you see in the screenshot that this get_Minimum_Cost_Path method will return minimum cost only of dictionary first we sort it in ascending order and then return 0 index value.



```
def get_Empty_tile_Location(self,state):
    zero_location = [x[0] for x in np.where(state==0)]
    return zero_location
```

This Method will return empty tile location we use comprehension and that where states ==0 and then we return it.

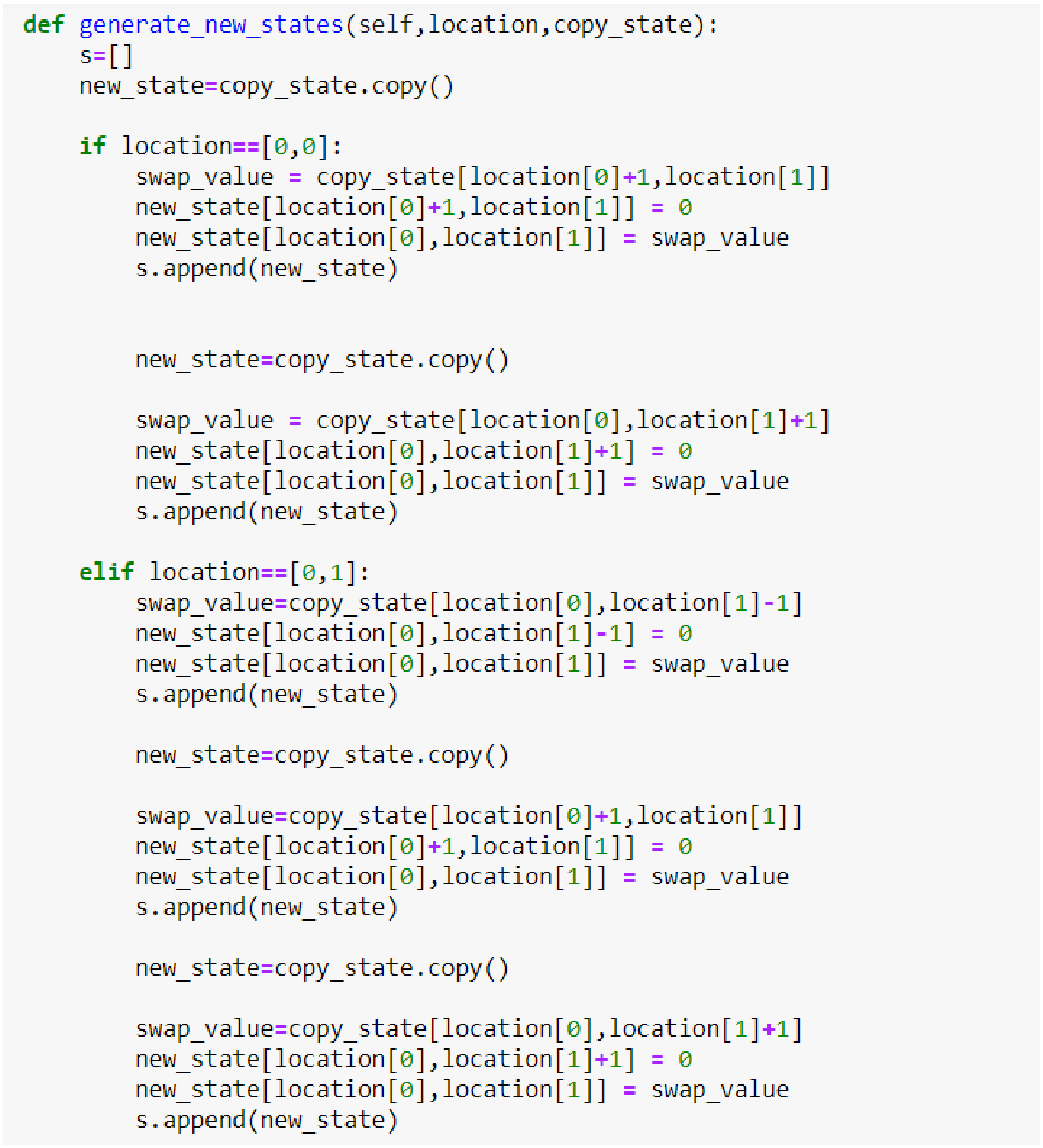
Note in np array we use 0 for empty tile



```python
def h_mismatch_cost(self,array,Goal_state):     ## cost of titles
    dic={}
    for i in array:
        cost = np.sum(i != Goal_state)
        if cost in dic.keys():
            arr=[]
            arr.append(dic.get(cost))
            arr.append(i)
            dic[cost]=arr
        else:
            cost = np.sum(i != Goal_state)
            dic[cost]=i

    return dic
```

This will return cost of tiles that how many away from the goal



```python
def generate_new_states(self,location,copy_state):
    s=[]
    new_state=copy_state.copy()

    if location==[0,0]:
        swap_value = copy_state[location[0]+1,location[1]]
        new_state[location[0]+1,location[1]] = 0
        new_state[location[0],location[1]] = swap_value
        s.append(new_state)


        new_state=copy_state.copy()

        swap_value = copy_state[location[0],location[1]+1]
        new_state[location[0],location[1]+1] = 0
        new_state[location[0],location[1]] = swap_value
        s.append(new_state)

    elif location==[0,1]:
        swap_value=copy_state[location[0],location[1]-1]
        new_state[location[0],location[1]-1] = 0
        new_state[location[0],location[1]] = swap_value
        s.append(new_state)

        new_state=copy_state.copy()

        swap_value=copy_state[location[0]+1,location[1]]
        new_state[location[0]+1,location[1]] = 0
        new_state[location[0],location[1]] = swap_value
        s.append(new_state)

        new_state=copy_state.copy()

        swap_value=copy_state[location[0],location[1]+1]
        new_state[location[0],location[1]+1] = 0
        new_state[location[0],location[1]] = swap_value
        s.append(new_state)
```

This method will generate new states and return that new states

In this method we pass location of empty tile and current state of puzzle

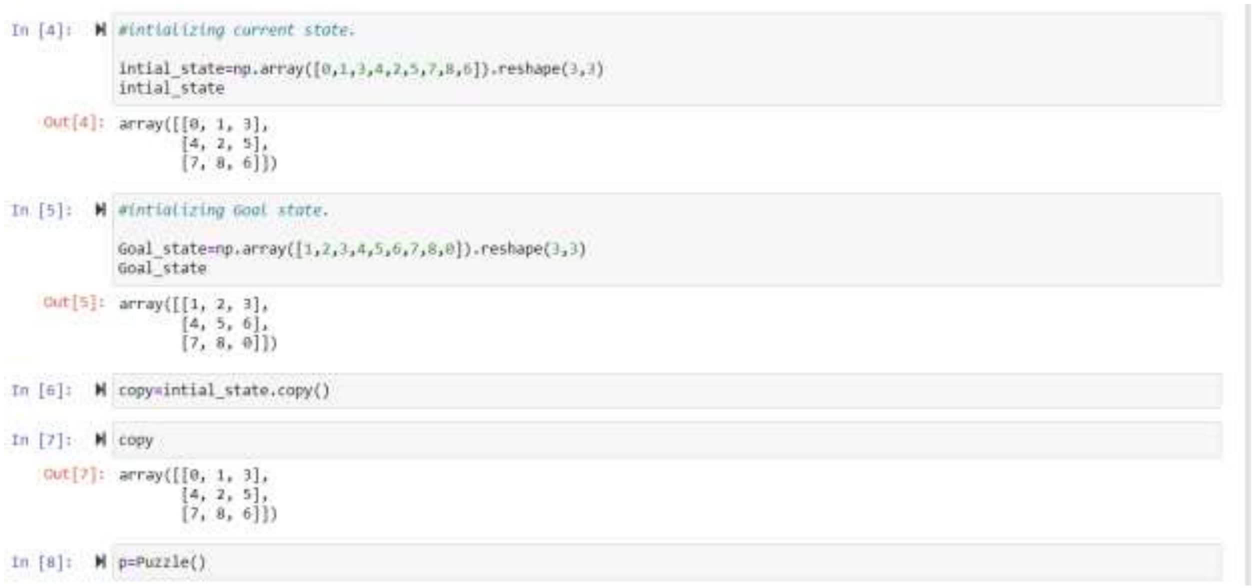
For example:

If location==[0,0] mean that if the empty tile is in 0 row and 0 column then we find it all possible scenario that empty tile goes right and down and generate that possible state and return that generated states.

```
def display_path(self,state):
    print(state)
```

This will simply display the state.

Now I'm explain main code



```
In [4]:  #intializing current state.

         intial_state=np.array([0,1,3,4,2,5,7,8,6]).reshape(3,3)
         intial_state

Out[4]:  array([[0, 1, 3],
                [4, 2, 5],
                [7, 8, 6]])

In [5]:  #intializing Goal state.

         Goal_state=np.array([1,2,3,4,5,6,7,8,0]).reshape(3,3)
         Goal_state

Out[5]:  array([[1, 2, 3],
                [4, 5, 6],
                [7, 8, 0]])

In [6]:  copy=intial_state.copy()

In [7]:  copy

Out[7]:  array([[0, 1, 3],
                [4, 2, 5],
                [7, 8, 6]])

In [8]:  p=Puzzle()
```

You see the screen we first define intial state and reshape into 3 by 3 matrix

And afterward we intial goal state and create object of puzzle

```
p.a_star_traversal(intial_state,Goal_state)

[[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]]
```

You see that we initial state and Goal state to the a_star_traversal method you see output also.