Lab 2

8 puzzle Bfs

① Initialize Node:
with state
with parent
with action

② Class puzzle:

* Initialize self, start, goal
* def neighbors(self, state):
* initialize mat, (row, col) = state
* result = []

if row > 0 then do
mat1 = copy(mat)
mat1[row][col], mat1[row-1][col] = mat1[row-1][col], mat1[row][col]
move the tailes up
~~print~~ append it to results

if col > 0
mat1 = copy(mat)
move the empty tiles to left
append it to results

if row < 2:
mat1 = copy(mat)
move the empty tiles right
append it to results.

def bfs(self):
initialize the start_node such that parent = None
action = None
initialize frontier which is deque
initialize visited = set()

While frontier:
* Get the node from the front of the queue
* if current state in goal state then do:
return solution path
* Mark the current state as visited

* for action, state in self.neighbors(node.state):

If neighbour_state not in visited:
Create child node
Add the child node to frontier

return None if no solution is found

def get_solution(self, node):
Create a list to store the action
Create a list to store the states

While parent is not None:
* Record the action that led to this state
* Store the current state
* node = node.parent

return actions, cells

print solution

State space tree

Initial state

1	2	3
4	8	0
7	6	5

Goal state

1	2	3
4	5	6
7	8	0