

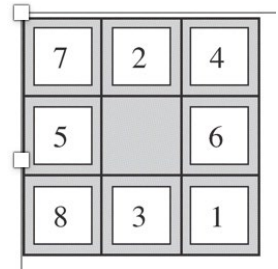
CPSC 4420/6420: ARTIFICIAL INTELLIGENCE

ASSIGNMENT 1- DUE: SEP 16, 2022 @11:59 PM

NAME: CHARANJIT SINGH

For the above puzzle shown here, develop a Python program that:

- (A) [10 pts] Lists all states [No need to submit the output (the list of states) for this part, and just submit the code and the first [or randomly selected] 10 states, since the list will be very long !!!]



7	2	4
5		6
8	3	1

Ans) Randomly selected first 10

states:

```
(3, 0, 8, 1, 2, 6, 4, 5, 7),  
(1, 6, 7, 2, 3, 0, 8, 5, 4),  
(8, 6, 7, 0, 4, 2, 5, 3, 1),  
(4, 0, 6, 2, 3, 5, 8, 1, 7),  
(3, 2, 8, 1, 7, 6, 0, 5, 4),  
(3, 5, 4, 1, 8, 6, 2, 0, 7),  
(2, 6, 3, 4, 5, 0, 1, 7, 8),  
(7, 5, 3, 2, 1, 0, 4, 6, 8),  
(7, 3, 2, 5, 0, 6, 4, 1, 8),  
(4, 8, 7, 2, 6, 0, 1, 3, 5)
```

Code:

```
#List of all states  
from itertools import permutations  
import random  
all_states =  
list(permutations([0,1,2,3,4,5,6,7,8]))  
print(f'Total no. of states:  
{len(all_states)}')  
print(random.sample(all_states,10))
```

- (B) [10 pts] Gets the current state and the action (moving up:1, down:2, left:3, right:4) as input, and returns the resulting state. Represent the blank spot with "0" and use one of the following naming formats for states
- Represent each state with a sequence of numbers from left to right and top to bottom. Ex. Use 7-2-4-5-0-6-8-3-1 for the state shown above
 - Represent each state by a 9-digit integer number. Like show the above state by 724506831

Ans) Code submitted in python HW1_B.py file

- (C) [10 pts] Suppose that the goal is to arrange the numbers so that the resulting 3-digit numbers created by each row are divisible by 3. For instance, 7-2-0-5-4-6-8-3-1 is a goal state because 720, 546, and 831 are divisible by 3. Write a program that prompts the user to receive an arbitrary initial state, and then performs random actions to reach the goal state. Show the sequence of actions and the sequence of states.

7	2	
5	4	6
8	3	1

Sample goal state

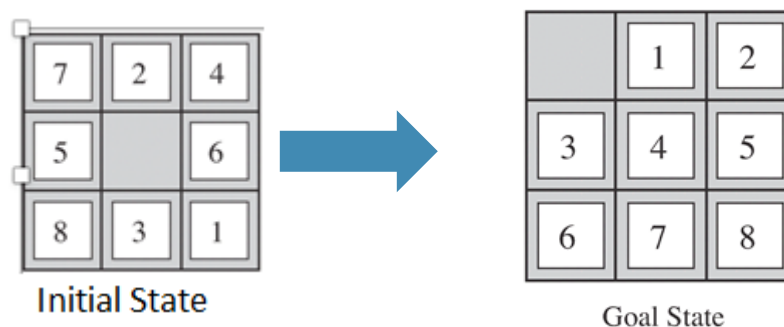
Ans) Code submitted in python HW1_C.py file

- (D) [20 pts] Suppose that the goal is arranging the blocks in numerical order as shown below. Develop a Breadth First Search (BFS) algorithm and show the results. Present the sequence of states and moves, starting from the initial state. How many moves (actions) did it take to reach the goal state?

Ans) No. of moves performed by BFS: 168883

Path Followed: ['724506831', '724056831', '024756831', '204756831', '254706831', '254736801', '254736081', '254036781', '254306781', '254360781', '250364781', '205364781', '025364781', '325064781', '325604781', '325640781', '325641780', '325641708', '325601748', '325610748', '320615748', '302615748', '312605748', '312645708', '312645078', '312045678', '012345678']

Code submitted in python HW1_D.py file



(E) [10 pts] Repeat part (D) using a Depth-First Search (DFS). How many moves (actions) did it take to reach the goal state?

Which algorithm found the solution with fewer moves? Explain your observation.

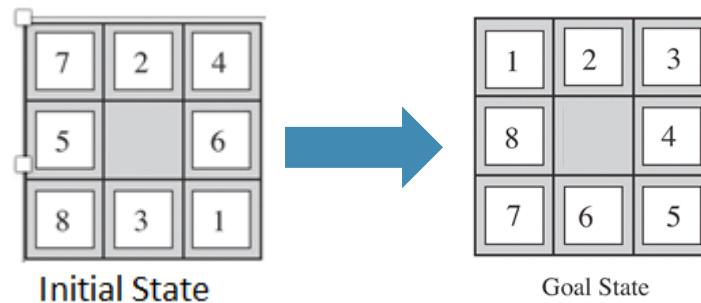
Ans) No. of moves performed by DFS: 111768

No. of states in final path: 59627

DFS found the goal state in fewer steps, but it was observed that the path followed to reach goal state in DFS (59627 steps) was much greater than the path in BFS (27 steps). So, although DFS provided a solution efficiently, BFS proved to be more optimal in this case.

Code submitted in python HW1_E.py file

(F) [10 pts] Repeat Part (D), if the goal is ordering the numbers clockwise around the blank space, with the given initial state, as shown below.



Ans) The given state is not solvable to this particular goal state. The code runs till the end of the loop, until it has explored all the possible child nodes and returns no path which can lead to the goal state.

(G) Implement a Uniform Cost Search (UCS), if the goal is achieving the final state in part F from an arbitrary initial state, if we have the following costs for different moves

[15 pts] G1) All moves have a unit cost

[15 pts] G2) Up (Cost=1), Down (Cost=1) Left (Cost=2) Right (Cost=0.5) Present the sequences of moves and states for each option. How many actions are used to achieve the solution for each option? Explain your observation.

Ans) Code submitted in python HW1_G1.py and HW1_G2.py files

Code Output for input state: (213084675)

Actions, cost in G1 = 31388, 19

Actions, cost in G2 = 32770, 20.5

(sequence of moves can be found in code output)

Observations:

- On differentiating the costs for different moves, the number of steps increased in order to find the path with lowest cost.
- The lowest cost path found by code in G2 had more cost (20.5) as compared to code in part G1.