Shri Vile Parle Kelavani Mandal's
DWARKADAS J. SANGHVI COLLEGE OF ENGINEERING
(Autonomous College Affiliated to the University of Mumbai)
NAAC Accredited with "A" Grade (CGPA : 3.18)

Department of Computer Science and Engineering (Data Science)

Subject: Artificial Intelligence (DJ19DSC502)

AY: 2023-24

Experiment 1

60009210033

Jhanvi Parekh

D11

(Problem Solving)

Aim: Implement domain specific functions for given problems required for problem solving.

Theory:

There are two domain specific functions required in all problem solving methods.
1. GoalTest Function:

goalTest(State) Returns *true* if the input state is the goal state and *false* otherwise.

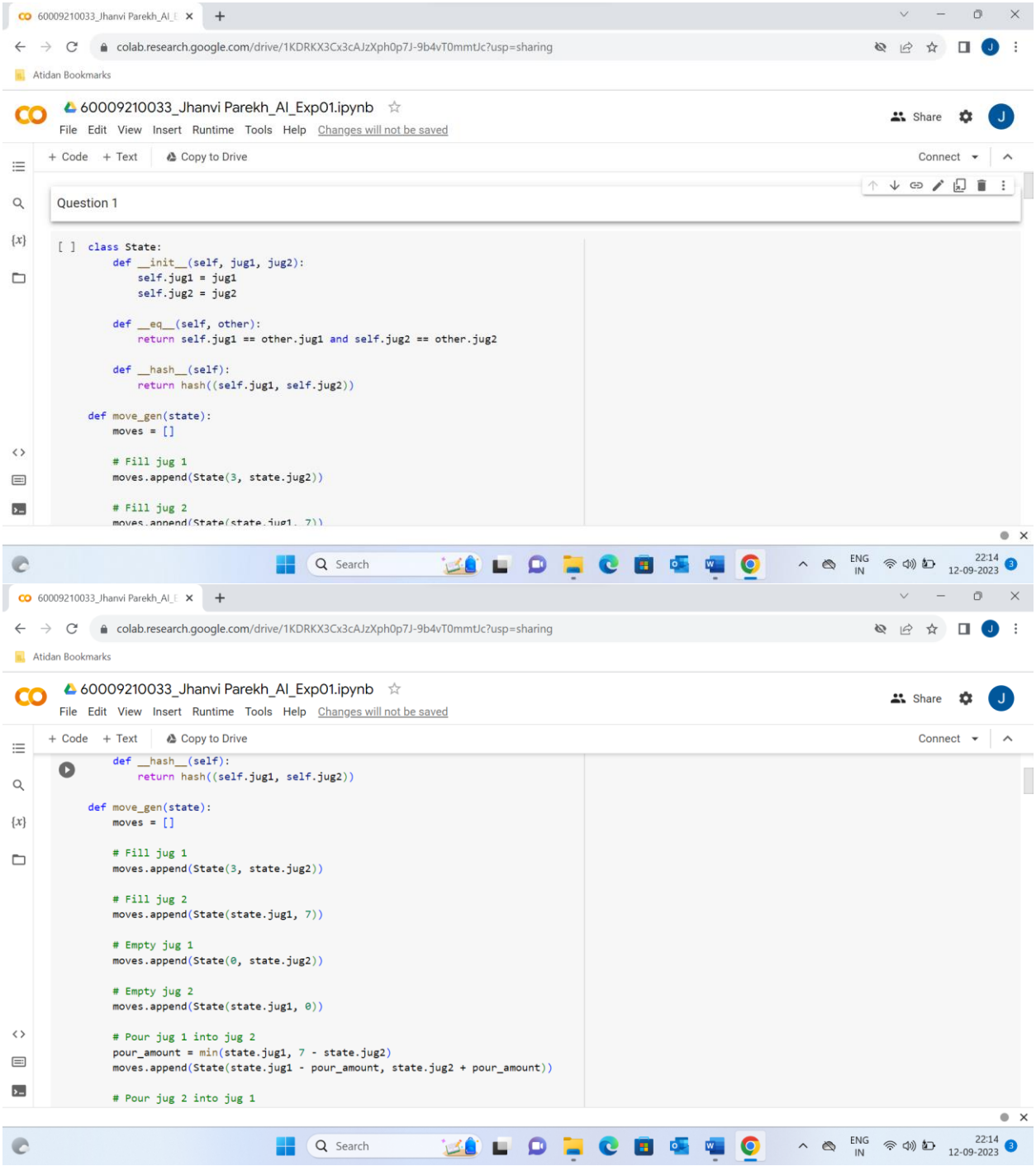
goalTest(State, Goal) Returns *true* if *State* matches *Goal,* and *false* otherwise.

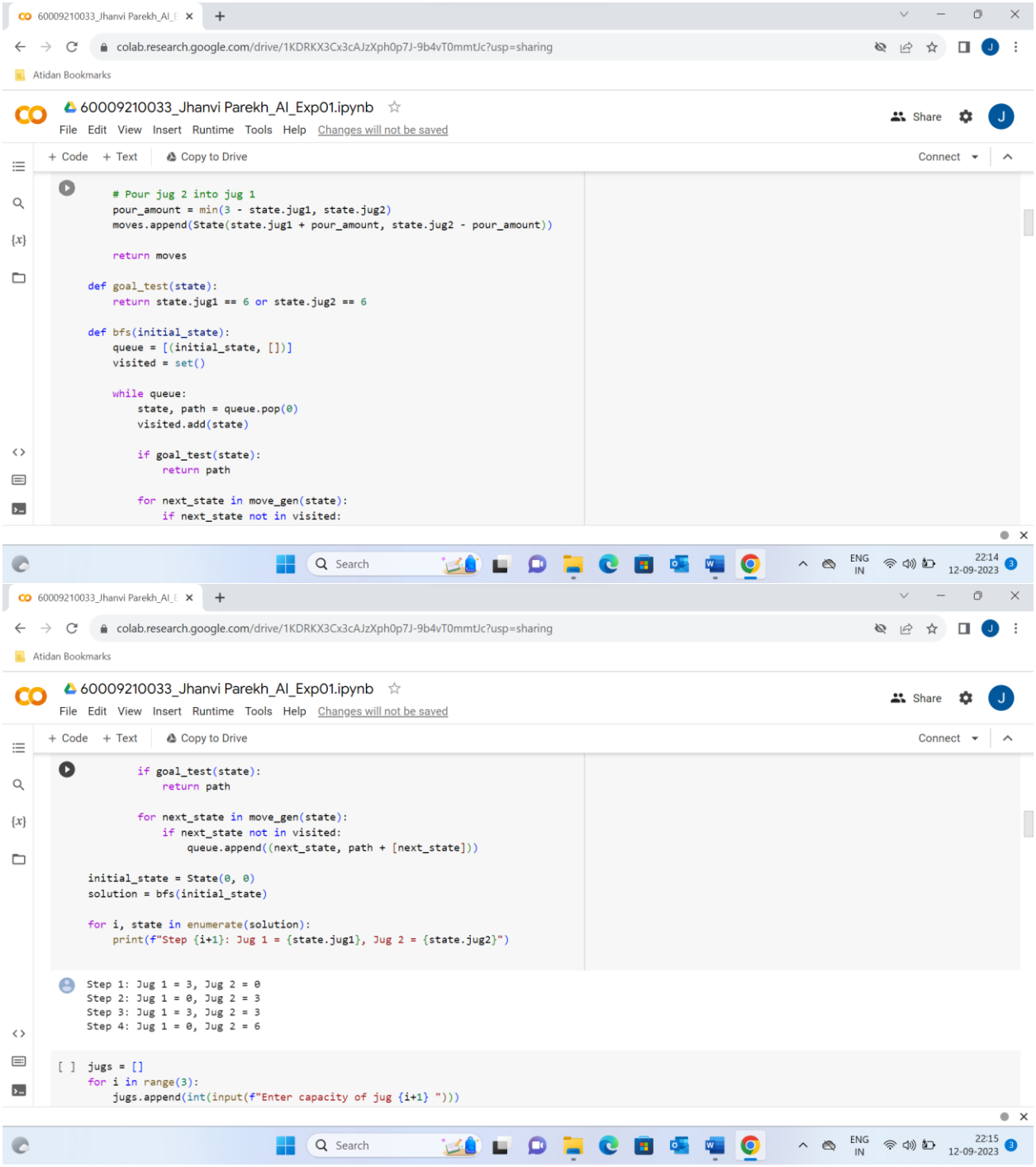
2. MoveGen function:

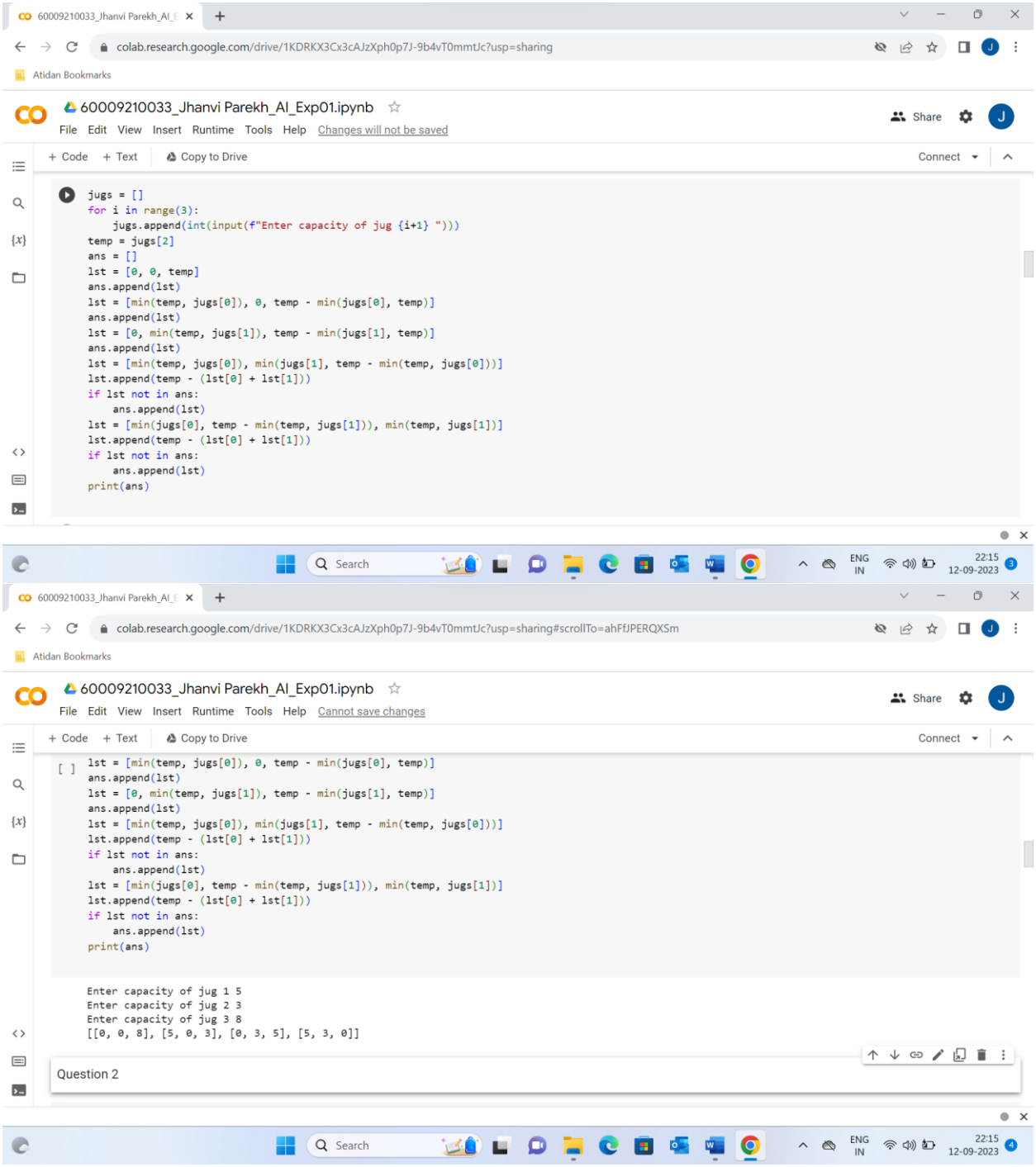
```
Initialize set of successors C to empty set.
Add M to the complement of given state N to get new state S.
If given state has Left, then add Right to S, else add Left.
If legal(S) then add S to set of successors C.
For each other-entity E in N
    make a copy S' of S,
    add E to S',
    If legal (S'), then add S' to C.
Return (C).
```

Lab Assignment to do:
Create MoveGen and GoalTest Functions for the given problems
1. **Water Jug Problem**

There are two jugs available of different volumes such as a 3 litres and a 7 litres and you have to measure a different volume such as 6 litre.

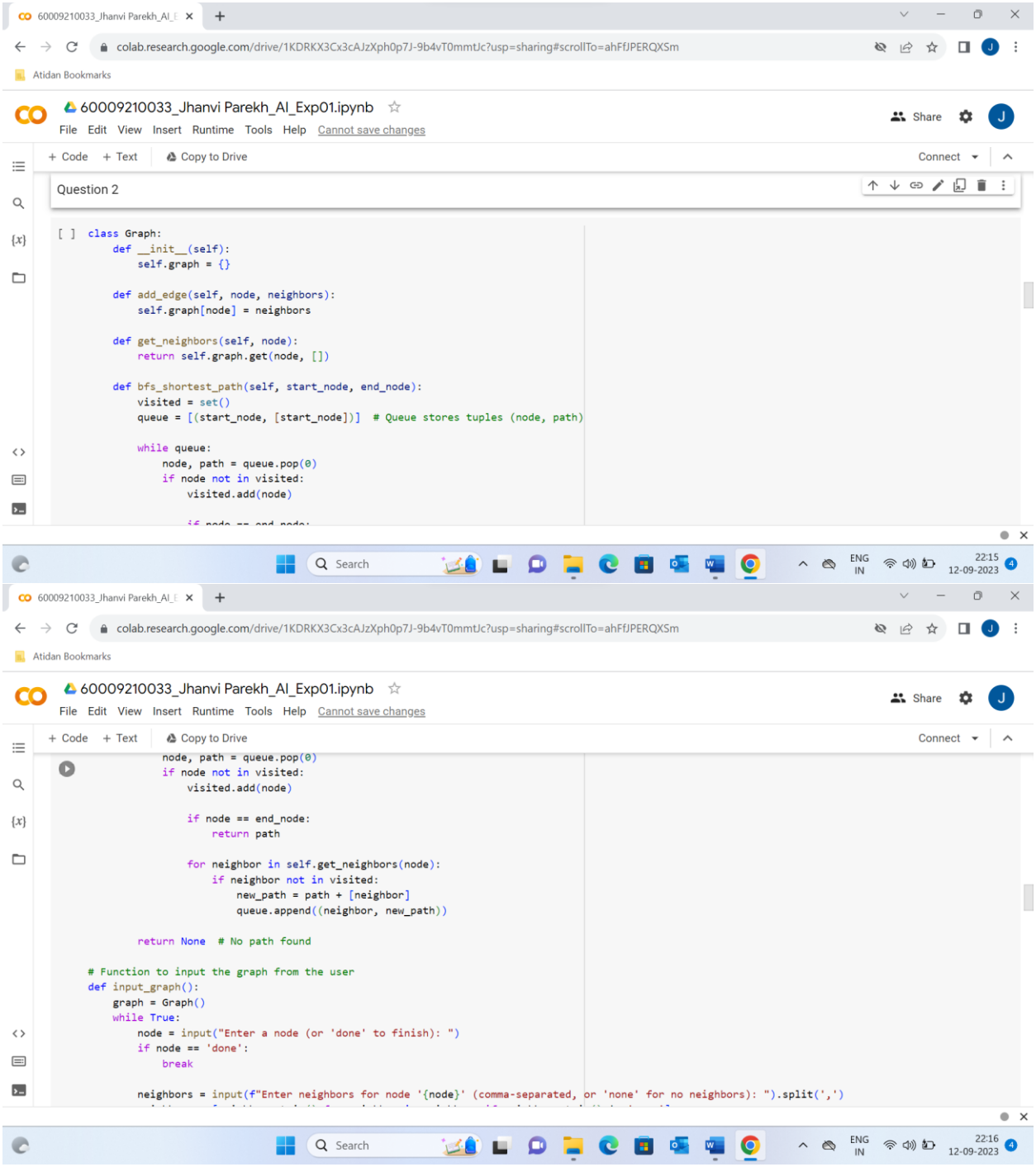


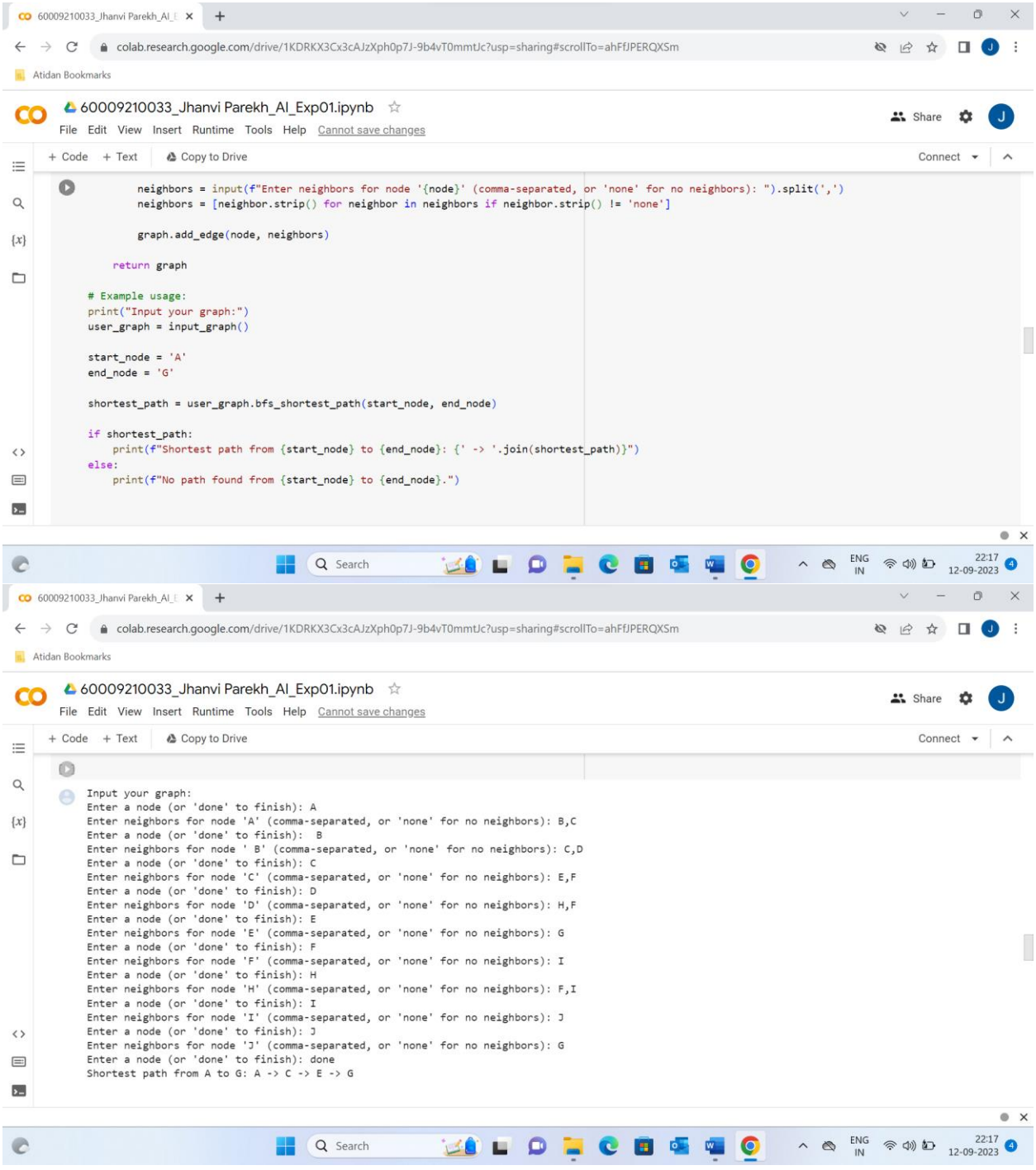

60009210033_Jhanvi Parekh_AI_E
colab.research.google.com/drive/1KDRKX3Cx3cAJzXph0p7J-9b4vT0mmtJc?usp=sharing
Atidan Bookmarks
60009210033_Jhanvi Parekh_AI_Exp01.ipynb
File Edit View Insert Runtime Tools Help Changes will not be saved
Share
+ Code + Text Copy to Drive
Connect
# Pour jug 2 into jug 1
pour_amount = min(3 - state.jug1, state.jug2)
moves.append(State(state.jug1 + pour_amount, state.jug2 - pour_amount))
return moves
def goal_test(state):
return state.jug1 == 6 or state.jug2 == 6
def bfs(initial_state):
queue = [(initial_state, [])]
visited = set()
while queue:
state, path = queue.pop(0)
visited.add(state)
if goal_test(state):
return path
for next_state in move_gen(state):
if next_state not in visited:
Search
ENG IN
22:14 12-09-2023
60009210033_Jhanvi Parekh_AI_E
colab.research.google.com/drive/1KDRKX3Cx3cAJzXph0p7J-9b4vT0mmtJc?usp=sharing
Atidan Bookmarks
60009210033_Jhanvi Parekh_AI_Exp01.ipynb
File Edit View Insert Runtime Tools Help Changes will not be saved
Share
+ Code + Text Copy to Drive
Connect
if goal_test(state):
return path
for next_state in move_gen(state):
if next_state not in visited:
queue.append((next_state, path + [next_state]))
initial_state = State(0, 0)
solution = bfs(initial_state)
for i, state in enumerate(solution):
print(f"Step {i+1}: Jug 1 = {state.jug1}, Jug 2 = {state.jug2}")
Step 1: Jug 1 = 3, Jug 2 = 0
Step 2: Jug 1 = 0, Jug 2 = 3
Step 3: Jug 1 = 3, Jug 2 = 3
Step 4: Jug 1 = 0, Jug 2 = 6
[ ] jugs = []
for i in range(3):
jugs.append(int(input(f"Enter capacity of jug {i+1} ")))
Search
ENG IN
22:15 12-09-2023



2. **Travelling Salesman Problem**

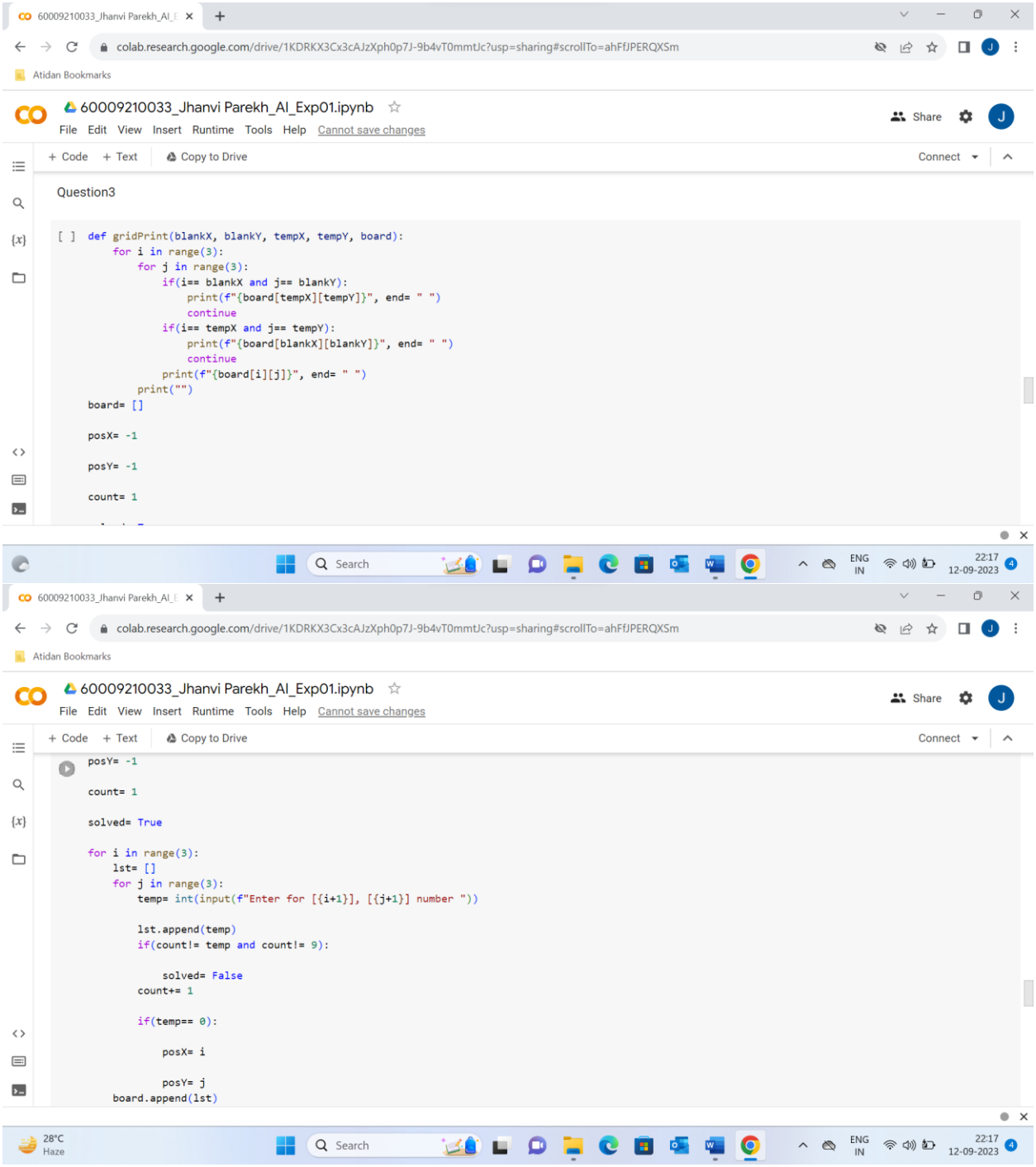
A salesman is travelling and selling his/her product to in different cities. The condition is that it has to travel each city just once.

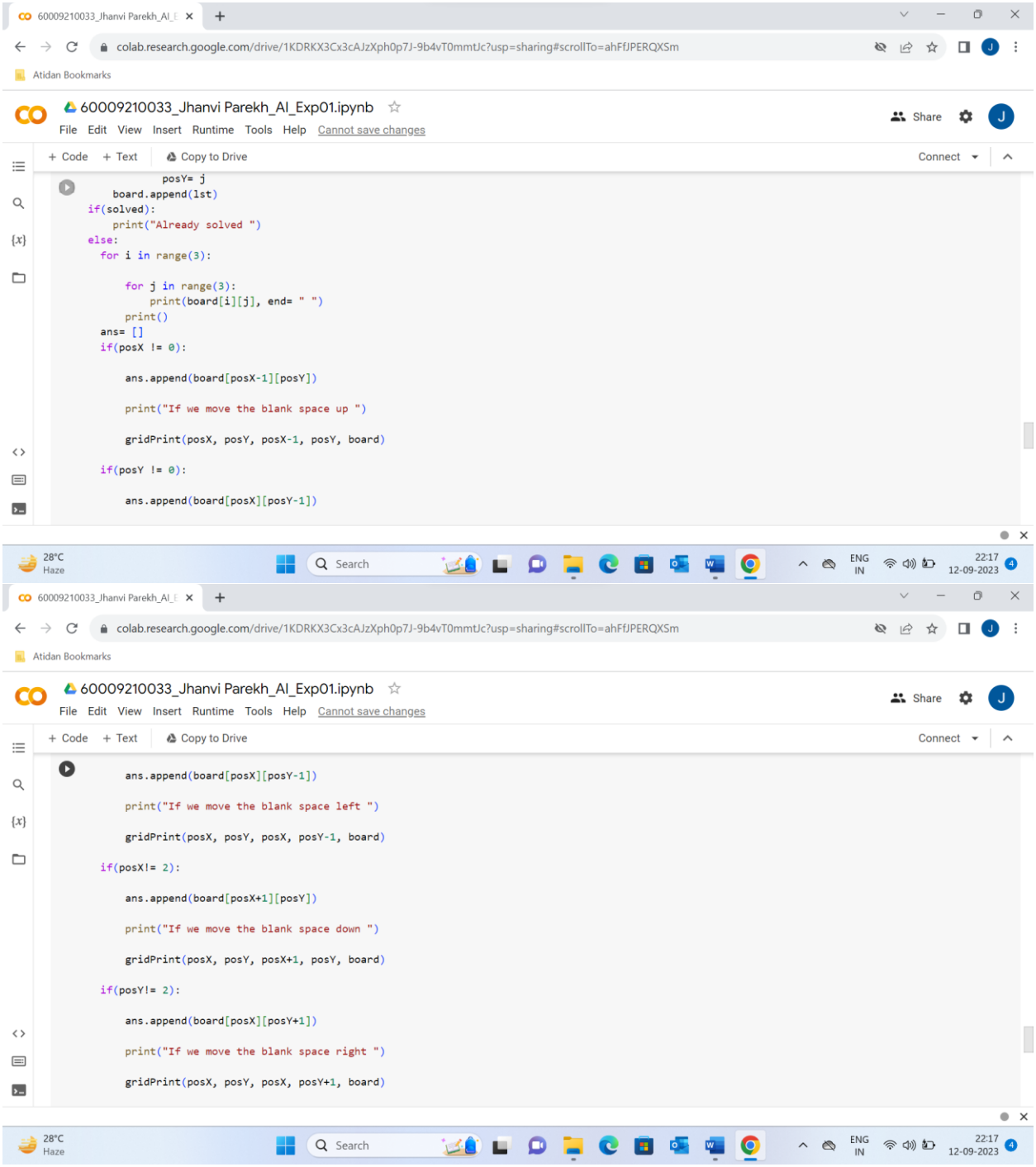

60009210033_Jhanvi Parekh_AI_E
colab.research.google.com/drive/1KDRKX3Cx3cAJzXph0p7J-9b4vT0mmtJc?usp=sharing#scrollTo=ahFfJPERQXSm
Atidan Bookmarks
60009210033_Jhanvi Parekh_AI_Exp01.ipynb
File Edit View Insert Runtime Tools Help Cannot save changes
Share
+ Code + Text Copy to Drive
Connect
Question 2
[ ] class Graph:
def __init__(self):
self.graph = {}
def add_edge(self, node, neighbors):
self.graph[node] = neighbors
def get_neighbors(self, node):
return self.graph.get(node, [])
def bfs_shortest_path(self, start_node, end_node):
visited = set()
queue = [(start_node, [start_node])] # Queue stores tuples (node, path)
while queue:
node, path = queue.pop(0)
if node not in visited:
visited.add(node)
Search
ENG IN
22:15 12-09-2023
60009210033_Jhanvi Parekh_AI_E
colab.research.google.com/drive/1KDRKX3Cx3cAJzXph0p7J-9b4vT0mmtJc?usp=sharing#scrollTo=ahFfJPERQXSm
Atidan Bookmarks
60009210033_Jhanvi Parekh_AI_Exp01.ipynb
File Edit View Insert Runtime Tools Help Cannot save changes
Share
+ Code + Text Copy to Drive
Connect
node, path = queue.pop(0)
if node not in visited:
visited.add(node)
if node == end_node:
return path
for neighbor in self.get_neighbors(node):
if neighbor not in visited:
new_path = path + [neighbor]
queue.append((neighbor, new_path))
return None # No path found
# Function to input the graph from the user
def input_graph():
graph = Graph()
while True:
node = input("Enter a node (or 'done' to finish): ")
if node == 'done':
break
neighbors = input(f"Enter neighbors for node '{node}' (comma-separated, or 'none' for no neighbors): ").split(',')
Search
ENG IN
22:16 12-09-2023

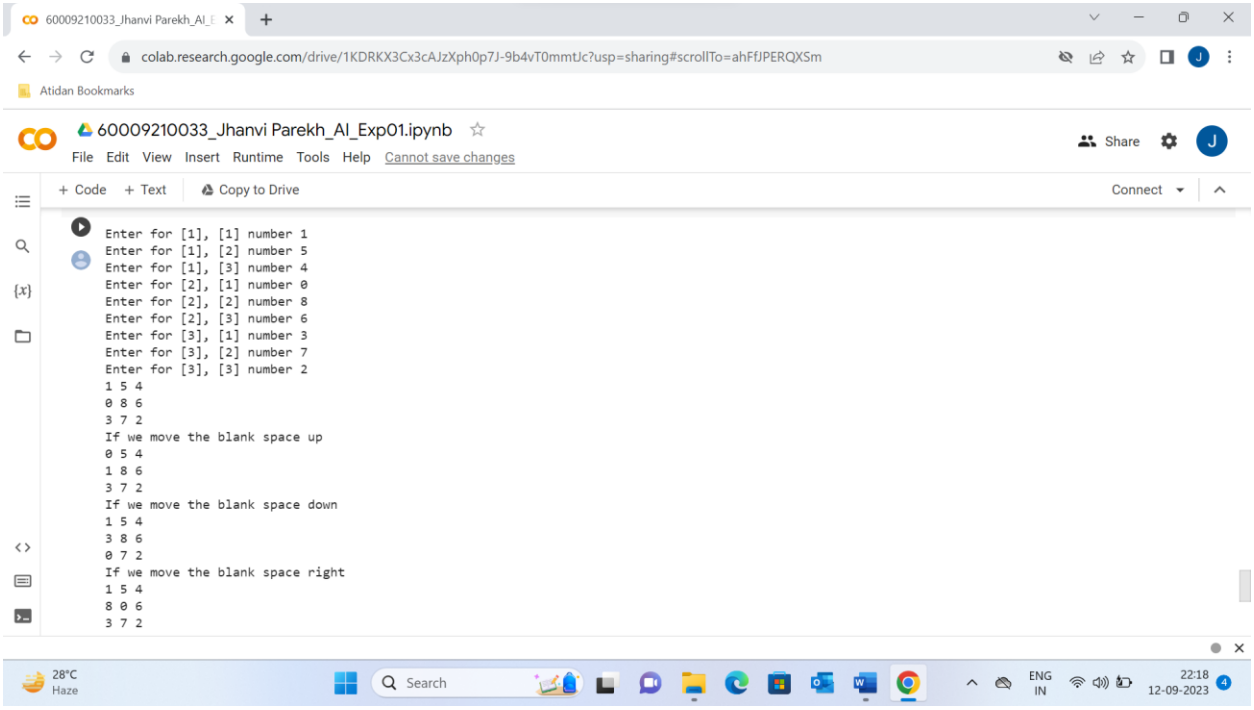


3**. 8 Puzzle Problem**

An initial state is given in a 8 puzzle where one place is blank out of 9 places. You can shift this blank space and get a different state to reach to a given goal state.


60009210033_Jhanvi Parekh_AI_Exp01.ipynb
File Edit View Insert Runtime Tools Help Cannot save changes
Share
+ Code + Text Copy to Drive
Connect
Question3
[ ] def gridPrint(blankX, blankY, tempX, tempY, board):
for i in range(3):
for j in range(3):
if(i== blankX and j== blankY):
print(f"{board[tempX][tempY]}", end= " ")
continue
if(i== tempX and j== tempY):
print(f"{board[blankX][blankY]}", end= " ")
continue
print(f"{board[i][j]}", end= " ")
print("")
board= []
posX= -1
posY= -1
count= 1
colab.research.google.com/drive/1KDRKX3Cx3cAJzXph0p7J-9b4vT0mmtJc?usp=sharing#scrollTo=ahFfJPERQXSm
Atidan Bookmarks
Search
ENG IN
22:17 12-09-2023
28°C Haze


60009210033_Jhanvi Parekh_AI_E
colab.research.google.com/drive/1KDRKX3Cx3cAJzXph0p7J-9b4vT0mmtJc?usp=sharing#scrollTo=ahFfJPERQXSm
Atidan Bookmarks
60009210033_Jhanvi Parekh_AI_Exp01.ipynb
File Edit View Insert Runtime Tools Help Cannot save changes
Share
+ Code + Text Copy to Drive
Connect
posY= j
board.append(lst)
if(solved):
print("Already solved ")
else:
for i in range(3):
for j in range(3):
print(board[i][j], end= " ")
print()
ans= []
if(posX != 0):
ans.append(board[posX-1][posY])
print("If we move the blank space up ")
gridPrint(posX, posY, posX-1, posY, board)
if(posY != 0):
ans.append(board[posX][posY-1])
28°C Haze
Search
ENG IN
22:17 12-09-2023
60009210033_Jhanvi Parekh_AI_E
colab.research.google.com/drive/1KDRKX3Cx3cAJzXph0p7J-9b4vT0mmtJc?usp=sharing#scrollTo=ahFfJPERQXSm
Atidan Bookmarks
60009210033_Jhanvi Parekh_AI_Exp01.ipynb
File Edit View Insert Runtime Tools Help Cannot save changes
Share
+ Code + Text Copy to Drive
Connect
ans.append(board[posX][posY-1])
print("If we move the blank space left ")
gridPrint(posX, posY, posX, posY-1, board)
if(posX!= 2):
ans.append(board[posX+1][posY])
print("If we move the blank space down ")
gridPrint(posX, posY, posX+1, posY, board)
if(posY!= 2):
ans.append(board[posX][posY+1])
print("If we move the blank space right ")
gridPrint(posX, posY, posX, posY+1, board)
28°C Haze
Search
ENG IN
22:17 12-09-2023



Link:
https://colab.research.google.com/drive/1KDRKX3Cx3cAJzXph0p7J-9b4vT0mmtJc?usp=sharing