5. Write the python program for Missionaries Cannibal problem

**AIM :** program for Missionaries Cannibal problem

**ALGORITHM :**

1. Define the initial and goal states. Each state consists of the number of missionaries and cannibals on each side of the river and the boat's position.
2. Implement a function to check the validity of a state. Ensure that the number of cannibals does not exceed the number of missionaries on either side of the river.
3. Implement a function to generate valid next states. This involves moving either 1 or 2 people from one side of the river to the other.
4. Implement a depth-first search or breadth-first search algorithm to explore possible state transitions until the goal state is reached.

**PROGRAM :**

def is\_valid\_state(state):

missionaries\_left, cannibals\_left, boat\_left = state

missionaries\_right = 3 - missionaries\_left

cannibals\_right = 3 - cannibals\_left

if missionaries\_left < 0 or missionaries\_left > 3:

return False

if cannibals\_left < 0 or cannibals\_left > 3:

return False

if missionaries\_right < 0 or missionaries\_right > 3:

return False

if cannibals\_right < 0 or cannibals\_right > 3:

return False

if (missionaries\_left < cannibals\_left) and missionaries\_left > 0:

return False

if (missionaries\_right < cannibals\_right) and missionaries\_right > 0:

return False

return True

def generate\_next\_states(current\_state):

states = []

missionaries, cannibals, boat = current\_state

for m in range(3):

for c in range(3):

if 1 <= m + c <= 2:

new\_state = (missionaries - m, cannibals - c, not boat)

if is\_valid\_state(new\_state):

states.append(new\_state)

return states

def depth\_first\_search(current\_state, visited, path):

visited.add(current\_state)

if current\_state == (0, 0, 0):

return path

for next\_state in generate\_next\_states(current\_state):

if next\_state not in visited:

result = depth\_first\_search(next\_state, visited, path + [next\_state])

if result:

return result

return None

initial\_state = (3, 3, 1)

visited\_states = set()

path = depth\_first\_search(initial\_state, visited\_states, [initial\_state])

if path:

print("Solution found:")

for state in path:

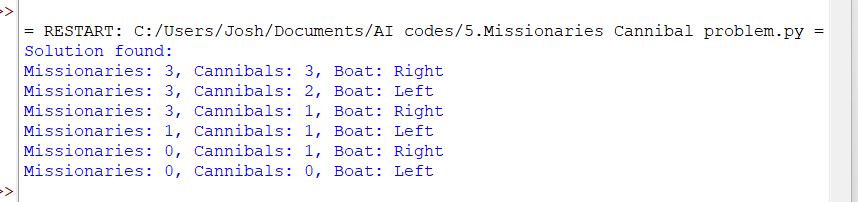
missionaries, cannibals, boat = state

print(f"Missionaries: {missionaries}, Cannibals: {cannibals}, Boat: {'Left' if boat == 0 else 'Right'}")

else:

print("No solution found.")

**OUT PUT :**

****