EXPERIMENT-1

Name: Adwait Purao
UID: 2021300101
Div: B
Batch: B2

Problem Definition: Implement an Intelligent agent (problem formulation and implementation) -Water Jug Problem, Missionary Cannibal problem.

Theory:

1) **Missionaries and Cannibal Problem:** In the missionaries and cannibals' problem, three missionaries and three cannibals must cross a river using a boat which can carry at most two people, under the constraint that, for both banks, if there are missionaries present on the bank, they cannot be outnumbered by cannibals (if they were, the cannibals would eat the missionaries). The boat cannot cross the river by itself with no people on board.

   Menu-Driven Algorithm for Missionary Cannibal Problem:

   - **Initialize the State:** Start with the initial state of the problem, including the positions of the missionaries, cannibals, and the boat. Initialize a list or data structure to store the state.

   - **Display the Menu:**
     - Display a menu to the user with options such as:
       - Show current state.
       - Choose and perform an action.
       - Exit the program.

   - **Show Current State:**
     - Provide an option to display the current state of the problem, showing the positions of the missionaries, cannibals, and the boat.

   - **Choose and Perform an Action:**
     - Allow the user to select an action from the menu. Actions can include moving one or two people from one side of the river to the other.
     - Implement logic to ensure that the selected action is valid (i.e., it doesn't violate the constraints of the problem).
     - Update the state based on the chosen action.

- **Check for Goal State:**
    - After each action, check whether the current state is the goal state where all missionaries and cannibals have safely crossed to the other side without any rule violations.

- **Repeat Steps 3-5:**
    - Keep looping through the menu options until the user chooses to exit or until the goal state is reached.

- **Exit the Program:**
    - Provide an option for the user to exit the program when they've either solved the problem or chosen to quit.

2) Water Jug Problem Definition

"You are given two jugs, a 4-liter one and a 3-liter one. Neither has any measuring markers on it. There is a pump that can be used to fill the jugs with water. How can you get exactly 2 liters of water in either of jug."

Menu-Driven Algorithm:

- **Initialize the State:** Start with both jugs empty. Create variables to represent the water levels in the 4-liter jug and the 3-liter jug.

- **Display the Menu:**
    - Display a menu to the user with options such as:
        - Show current jug levels.
        - Fill the 4-liter jug.
        - Fill the 3-liter jug.
        - Empty the 4-liter jug.
        - Empty the 3-liter jug.
        - Pour from the 4-liter jug to the 3-liter jug.
        - Pour from the 3-liter jug to the 4-liter jug.
        - Check if the goal is achieved.
        - Exit the program.

- **Show Current Jug Levels:**
    - Provide an option to display the current water levels in both jugs.

- **Perform Actions:**
    - Allow the user to select an action from the menu. Implement logic to update the water levels in the jugs according to the selected action.

- **Check for Goal:**
  - After each action, check if the goal state is reached where either of the jugs contains 2 liters of water.

- **Repeat Steps 3-5:**
  - Keep looping through the menu options until the user chooses to exit or until the goal state is achieved.

- **Exit the Program:**
  - Provide an option for the user to exit the program when they have successfully measured 2 liters or if they choose to quit.

This menu-driven approach allows the user to interact with the Water Jug problem by selecting actions, observing the current jug levels, and determining when they have successfully achieved the goal of measuring exactly 2 liters of water in either jug.

Code:

1. **Missionaries and Cannibal Problem:**

```
import java.util.Scanner;

class Exp_1_1 {
    // ml -> Missionaries on left
    // mr -> Missionaries on left
    // cl -> Cannibals on left
    // cr -> Cannibals on left
    // bp -> Boat Position
    private int ml;
    private int cl;
    private int mr;
    private int cr;
    private int bp;

    public Exp_1_1() {
        ml = 3;
        cl = 3;
        mr = 0;
        cr = 0;
        bp = 0;
    }

    public void currState() {
        System.out.println("\nCurrent State:");
```

```
        System.out.println("Left  = " + " Missionaries " + ml + "  |  " + "
Cannibals " + cl);
        System.out.println("Right = " + " Missionaries " + mr + "  |  " + "
Cannibals " + cr);
        System.out.println("Boat = " + (bp == 0 ? "Left" : "Right") + " side");
    }

    public void move(int missionaries, int cannibals) {
        if (bp == 0) {
            ml -= missionaries;
            cl -= cannibals;
            mr += missionaries;
            cr += cannibals;
        } else {
            mr -= missionaries;
            cr -= cannibals;
            ml += missionaries;
            cl += cannibals;
        }
        bp = 1 - bp;
    }

    public boolean validate(int missionaries, int cannibals) {
        if (missionaries < 0 || cannibals < 0) {
            return false;
        }
        if (bp == 0) {
            if (missionaries > 2 || cannibals > 2) {
                return false;
            }
            return (missionaries + cannibals >= 1) && (ml -
                    missionaries >= 0)
                    && (cl - cannibals >= 0);
        } else {
            if (missionaries > 2 || cannibals > 2) {
                return false;
            }
            return (missionaries + cannibals >= 1) && (mr -
                    missionaries >= 0)
                    && (cr - cannibals >= 0);
        }
    }

    public boolean isResult() {
```

```
        return ml == 0 && cl == 0 && bp == 1;
    }

    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        Exp_1_1 problem = new Exp_1_1();
        System.out.println("\n*************** Missionary and Cannibal Game
***************");
        while (true) {
            problem.currState();
            if (problem.isResult()) {
                System.out.println("Congratulations! Problem Solved.");
                break;
            }
            System.out.print("\nEnter the number of missionaries and cannibals to
move (Enter -1 to exit): ");
            int missionaries = sc.nextInt();
            int cannibals = sc.nextInt();
            if (missionaries == -1 || cannibals == -1) {
                System.out.println("Exiting the program.");
                break;
            }
            if (problem.validate(missionaries, cannibals)) {
                problem.move(missionaries, cannibals);
            }
            else {
                System.out.println("Invalid move. Please try again.");
            }
        }
        sc.close();
    }
}
```

2. Water Jug Problem

```
import java.util.Scanner;

public class Exp_1_2 {
    private int jug4Capacity = 4;
    private int jug3Capacity = 3;
    private int jug4Level = 0;
    private int jug3Level = 0;

    public void currState() {
        System.out.println("\nCurrent Jug Levels:");
        System.out.println("4-Liter Jug: " + jug4Level + " liters");
        System.out.println("3-Liter Jug: " + jug3Level + " liters");
    }

    public void fill4LiterJug() {
        jug4Level = jug4Capacity;
    }

    public void fill3LiterJug() {
        jug3Level = jug3Capacity;
    }

    public void empty4LiterJug() {
        jug4Level = 0;
    }

    public void empty3LiterJug() {
        jug3Level = 0;
    }

    public void pourFrom4To3() {
        int spaceIn3Jug = jug3Capacity - jug3Level;
        if (jug4Level <= spaceIn3Jug) {
            jug3Level += jug4Level;
            jug4Level = 0;
        } else {
            jug4Level -= spaceIn3Jug;
            jug3Level = jug3Capacity;
        }
    }
```

```
    public void pourFrom3To4() {
        int spaceIn4Jug = jug4Capacity - jug4Level;
        if (jug3Level <= spaceIn4Jug) {
            jug4Level += jug3Level;
            jug3Level = 0;
        } else {
            jug3Level -= spaceIn4Jug;
            jug4Level = jug4Capacity;
        }
    }

    public boolean isResult() {
        return jug4Level == 2 || jug3Level == 2;
    }

    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        Exp_1_2 problem = new Exp_1_2();
        System.out.println("*************** Water Jug Problem ***************");
        while (true) {
            problem.currState();
            if (problem.isResult()) {
                System.out.println("Congratulations! 2 liters measured
successfully.");
                break;
            }
            System.out.println("\nChoose an action:");
            System.out.println("1. Fill the 4-Liter Jug");
            System.out.println("2. Fill the 3-Liter Jug");
            System.out.println("3. Empty the 4-Liter Jug");
            System.out.println("4. Empty the 3-Liter Jug");
            System.out.println("5. Pour from 4-Liter Jug to 3-Liter Jug");
            System.out.println("6. Pour from 3-Liter Jug to 4-Liter Jug");
            System.out.println("7. Quit");
            System.out.print("Enter your choice: ");
            int choice = scanner.nextInt();
            switch (choice) {
                case 1:
                    problem.fill4LiterJug();
                    break;
                case 2:
                    problem.fill3LiterJug();
                    break;
                case 3:
```

```
                    problem.empty4LiterJug();
                    break;
                case 4:
                    problem.empty3LiterJug();
                    break;
                case 5:
                    problem.pourFrom4To3();
                    break;
                case 6:
                    problem.pourFrom3To4();
                    break;
                case 7:
                    System.out.println("Exiting the program.");
                    scanner.close();
                    return;
                default:
                    System.out.println("Invalid choice. Please try again.");
            }
        }
        scanner.close();
    }
}
```

Output:

1. **Missionaries and Cannibal Problem:**

```
PS D:\Rohit College\5th sem\AIML\Exp 1> cd "d:\Rohit College\5th sem\AIML\Exp 1\" ; if ($?) { javac Exp_1_1.java } ; if ($?) { java Exp_1_1 }

 ********** Missionary and Cannibal Game **********

Current State:
Left  =  Missionaries 3  |   Cannibals 3
Right =  Missionaries 0  |   Cannibals 0
Boat = Left side

Enter the number of missionaries and cannibals to move (Enter -1 to exit): 0 2

Current State:
Left  =  Missionaries 3  |   Cannibals 1
Right =  Missionaries 0  |   Cannibals 2
Boat = Right side

Enter the number of missionaries and cannibals to move (Enter -1 to exit): 0 1

Current State:
Left  =  Missionaries 3  |   Cannibals 2
Right =  Missionaries 0  |   Cannibals 1
Boat = Left side

Enter the number of missionaries and cannibals to move (Enter -1 to exit): 0 2

Current State:
Left  =  Missionaries 3  |   Cannibals 0
Right =  Missionaries 0  |   Cannibals 3
Boat = Right side

Enter the number of missionaries and cannibals to move (Enter -1 to exit): 0 1

Current State:
Left  =  Missionaries 3  |   Cannibals 1
Right =  Missionaries 0  |   Cannibals 2
Boat = Left side

Enter the number of missionaries and cannibals to move (Enter -1 to exit): 1 1

Current State:
Left  =  Missionaries 2  |   Cannibals 0
Right =  Missionaries 1  |   Cannibals 3
Boat = Right side
```

```
Enter the number of missionaries and cannibals to move (Enter -1 to exit): 0 2

Current State:
Left  =  Missionaries 2  |   Cannibals 2
Right =  Missionaries 1  |   Cannibals 1
Boat = Left side

Enter the number of missionaries and cannibals to move (Enter -1 to exit): 2 0

Current State:
Left  =  Missionaries 0  |   Cannibals 2
Right =  Missionaries 3  |   Cannibals 1
Boat = Right side

Enter the number of missionaries and cannibals to move (Enter -1 to exit): 0 1

Current State:
Left  =  Missionaries 0  |   Cannibals 3
Right =  Missionaries 3  |   Cannibals 0
Boat = Left side

Enter the number of missionaries and cannibals to move (Enter -1 to exit): 0 2

Current State:
Left  =  Missionaries 0  |   Cannibals 1
Right =  Missionaries 3  |   Cannibals 2
Boat = Right side

Enter the number of missionaries and cannibals to move (Enter -1 to exit): 0 1

Current State:
Left  =  Missionaries 0  |   Cannibals 2
Right =  Missionaries 3  |   Cannibals 1
Boat = Left side

Enter the number of missionaries and cannibals to move (Enter -1 to exit): 0 2

Current State:
Left  =  Missionaries 0  |   Cannibals 0
Right =  Missionaries 3  |   Cannibals 3
Boat = Right side
Congratulations! Problem Solved.
PS D:\Rohit College\5th sem\AIML\Exp 1>
```

2. Water Jug Problem

```
PS D:\Rohit College\5th sem\AIML\Exp 1> cd "d:\Rohit College\5th sem\AIML\Exp 1\" ; if ($?) { javac Exp_1_2.java } ; if ($?) { java Exp_1_2 }
************** Water Jug Problem **************

Current Jug Levels:
4-Liter Jug: 0 liters
3-Liter Jug: 0 liters

Choose an action:
1. Fill the 4-Liter Jug
2. Fill the 3-Liter Jug
3. Empty the 4-Liter Jug
4. Empty the 3-Liter Jug
5. Pour from 4-Liter Jug to 3-Liter Jug
6. Pour from 3-Liter Jug to 4-Liter Jug
7. Quit
Enter your choice: 2

Current Jug Levels:
4-Liter Jug: 0 liters
3-Liter Jug: 3 liters

Choose an action:
1. Fill the 4-Liter Jug
2. Fill the 3-Liter Jug
3. Empty the 4-Liter Jug
4. Empty the 3-Liter Jug
5. Pour from 4-Liter Jug to 3-Liter Jug
6. Pour from 3-Liter Jug to 4-Liter Jug
7. Quit
Enter your choice: 6

Current Jug Levels:
4-Liter Jug: 3 liters
3-Liter Jug: 0 liters

Choose an action:
1. Fill the 4-Liter Jug
2. Fill the 3-Liter Jug
3. Empty the 4-Liter Jug
4. Empty the 3-Liter Jug
5. Pour from 4-Liter Jug to 3-Liter Jug
6. Pour from 3-Liter Jug to 4-Liter Jug
7. Quit
Enter your choice: 2

Current Jug Levels:
4-Liter Jug: 3 liters
3-Liter Jug: 3 liters

Choose an action:
1. Fill the 4-Liter Jug
2. Fill the 3-Liter Jug
3. Empty the 4-Liter Jug
4. Empty the 3-Liter Jug
5. Pour from 4-Liter Jug to 3-Liter Jug
6. Pour from 3-Liter Jug to 4-Liter Jug
7. Quit
Enter your choice: 6

Current Jug Levels:
4-Liter Jug: 4 liters
3-Liter Jug: 2 liters
Congratulations! 2 liters measured successfully.
PS D:\Rohit College\5th sem\AIML\Exp 1>
```