**Overview of the Problem -**

The problem at hand is a **nurse scheduling problem** where the goal is to assign nurses to shifts over a 56-day period while satisfying various constraints. The problem involves:

1. **Nurses:** A list of nurses who belong to different teams (e.g., Team A).
2. **Shifts:** Each day is divided into 3 shifts.
3. **Days:** The scheduling is for 56 days.
4. **Constraints:** Various conditions must be met regarding how shifts are assigned, ensuring fairness, continuity, and other team-specific rules.

**Approach**

To solve this problem, we use **Constraint Programming (CP)**, which is well-suited for scheduling problems with complex constraints. The model is created using the **OR-Tools library** from Google, specifically the CpModel class.

**Detailed Explanation**

1. **Variables:**
   * The primary decision variables are shifts[(n, d, s)], where n represents a nurse, d a day, and s a shift. These variables are Boolean (0 or 1), indicating whether nurse n is working shift s on day d.
2. **Constraints:**
   * **One Nurse Per Shift Per Day:** Ensures that each shift is assigned to exactly one nurse each day. This is enforced using the AddExactlyOne constraint.
   * **One Shift Per Day Per Nurse:** Ensures that each nurse works at most one shift per day, achieved using the AddAtMostOne constraint.
   * **Even Distribution of Shifts:** The total number of shifts worked by each nurse should be approximately equal. The model calculates the minimum and maximum number of shifts each nurse should work and enforces these bounds using the Add constraint.
   * **Fair Weekend Shifts Distribution:** Ensures that the weekend shifts are distributed evenly among the nurses.
   * **Avoid Consecutive Shift 0 and 1 Assignments:** Prevents nurses from being assigned to shifts 0 or 1 on consecutive days, ensuring better work-life balance.
   * **Ensure Shift 2 Continuity:** If a nurse works the night shift (Shift 2) on Monday, they should continue working it for the next two days.
   * **Limit to 4 Shifts Per Week:** No nurse should work more than 4 shifts in any given week.
   * **Team A Constraints:** Nurses in Team A are not allowed to work Shift 0.
   * **One Nurse per Team per Shift:** No more than one nurse from the same team should work on the same shift on any given day.
3. **Solution Search:**
   * A custom solution callback class, NursesPartialSolutionPrinter, is defined to print out solutions as they are found, up to a specified limit (5 solutions in this case).
   * The solver explores the solution space and uses the constraints to prune invalid assignments, searching for valid schedules that satisfy all conditions.
4. **Solver Execution:**
   * The CpSolver is instantiated and the model is solved while printing solutions using the custom solution printer. After the search, statistics such as the number of conflicts, branches, and wall time are reported.

**Output**

The output includes several feasible solutions (up to 5), showing how shifts are assigned to nurses across the 56 days. Each solution respects all the defined constraints. Additionally, the statistics provide insight into the complexity of the problem and the efficiency of the solver.

**Summary**

The approach leverages the power of constraint programming to solve a complex nurse scheduling problem by defining and enforcing various constraints related to shift assignments. The solution provides multiple feasible schedules, ensuring fairness, continuity, and adherence to team-specific rules.