**Create fitness Tracker System**

**Objective:**

Design and implement a Python-based Fitness Tracker System that helps users monitor their daily activities, workouts, and health metrics using OOP principles.

**Task Description:**

1. **Create a Workout class** to represent different types of workouts. Include the following attributes:
   * workout\_type (string): Represents the type of workout (e.g., running, cycling, weightlifting).
   * duration (float): Duration of the workout in minutes.
   * calories\_burned (float): The number of calories burned during the workout.

**Required Methods**:

* + calculate\_calories(): A method to calculate calories burned based on workout type and duration.
  + get\_workout\_summary(): Returns a summary of the workout including type, duration, and calories burned.

1. **Create a HealthMetric class** to represent health data tracked by the user:
   * steps (integer): The number of steps taken in a day.
   * heart\_rate (float): The average heart rate during the day.
   * sleep\_hours (float): The number of hours slept.
   * hydration (float): Amount of water consumed (in liters).

**Required Methods**:

* + track\_steps(step\_count): Updates the step count.
  + track\_sleep(hours): Updates the sleep hours.
  + track\_hydration(amount): Updates the hydration level.
  + get\_health\_summary(): Returns a summary of health metrics for the day.

1. **Create a User class** that stores personal information and manages workouts and health metrics:
   * **Attributes**:
     + name (string): The name of the user.
     + age (integer): The age of the user.
     + weight (float): The weight of the user.
     + height (float): The height of the user.
     + workouts (list of Workout objects): A list of the user's completed workouts.
     + health\_metrics (list of HealthMetric objects): A list of the user’s daily health metrics.

**Required Methods**:

* + add\_workout(workout): Adds a new workout to the user’s workout log.
  + add\_health\_metrics(health\_metric): Logs daily health metrics for the user.
  + calculate\_bmi(): Calculates and returns the user’s BMI using weight and height.
  + get\_fitness\_summary(): Provides an overview of all workouts and health metrics, including total calories burned, average heart rate, and step count for the week.

1. **Create a FitnessApp class** to manage the overall system:
   * **Attributes**:
     + users (list of User objects): A list of all users registered in the system.

**Required Methods**:

* + register\_user(name, age, weight, height): Registers a new user by creating a User object.
  + log\_workout(user\_name, workout): Logs a workout for a specific user.
  + log\_health\_metrics(user\_name, health\_metric): Logs health metrics for a specific user.
  + generate\_user\_report(user\_name): Generates a detailed report for a user, showing workouts, health metrics, and BMI.

1. **Additional Requirements**:
   * **Calories Burned Calculation**: Customize the calorie calculation method to use different formulas based on workout type and intensity.
   * **Weekly Summary**: Implement a method that generates a weekly summary of all workouts, health metrics, and fitness progress for a user.
   * **Progress Tracking**: Track progress towards fitness goals (e.g., a daily step target, a calorie burn goal).

**Advanced Features (Optional):**

* **Goal Setting**: Allow users to set personal fitness goals (e.g., “Run 10 miles this week” or “Burn 3000 calories”).
* **Notifications**: Add a feature to notify users when they meet or fail to meet their goals.
* **Workout Plans**: Allow users to create predefined workout plans and track their progress over time.