Digital Step Counter

- **Scenario**: You're creating a simple fitness step counter for a smartwatch. Every hour, the smartwatch records the number of steps taken. At the end of a day, the device should analyze the step data.
- **Problem**: Write a program that takes the number of steps for each hour (24 inputs) and calculates:
    - Total steps taken in the day.
    - Average steps per hour.
    - The hour with the maximum and minimum steps.
- **Requirements**:
    - Use a `for` loop to input and analyze data.
    - Use variables to track the total, average, and peak hours.
- **Challenge**: Display a message if there's any hour with no steps recorded (indicating inactivity).

2. Budget Tracker for Daily Expenses

- **Scenario**: A student is tracking their daily expenses over a week to manage their budget. They want to understand which days they're overspending.
- **Problem**: Write a program that:
    - Takes daily spending amounts as input for 7 days.
    - Calculates the total weekly spending and average daily spending.
    - Checks if any day's spending exceeds a user-defined daily budget limit and flags that day as "Overspent."
- **Requirements**:
    - Use a `for` loop to input expenses and analyze each day.
- **Challenge**: Calculate how much over the budget the user was for any "Overspent" day and provide an overall spending summary.

3. Password Strength Checker

- **Scenario**: A company wants to ensure employees are creating strong passwords. A password is considered "strong" if it's at least 8 characters long, contains both uppercase and lowercase letters, and includes at least one numeric digit.
- **Problem**: Write a program that repeatedly prompts the user to enter a password until a strong password is provided.
- **Requirements**:
    - Use a `while` loop to repeatedly check if the password meets the criteria.
    - Use conditional statements to validate each rule.
- **Challenge**: After each failed attempt, provide feedback on what the password is missing (length, case, or digits).

4. Temperature Analysis for Weather App

- **Scenario**: A weather app collects temperature data for each day of a month and needs to analyze trends to give users insights into temperature changes.
- **Problem**: Write a program that takes 30 temperature readings (one for each day of the month) and calculates:
    - The average monthly temperature.
    - The highest and lowest temperature recorded.
    - Days where the temperature was above the average.
- **Requirements**:
    - Use a `for` loop to input temperature data and calculate the average, max, and min.
- **Challenge**: List the days with temperatures above average and indicate "Hot Days" if the temperature was above 35°C.

5. Online Study Hours Tracker

- **Scenario**: A student wants to track their study hours daily to see if they're meeting their weekly study goals.
- **Problem**: Write a program that:
    - Asks the student to enter their study hours for each day of the week (7 inputs).
    - Calculates the total study hours for the week.
    - Checks if they met the minimum weekly goal of 35 hours and prints a motivational message if they fall short.
- **Requirements**:
    - Use a `for` loop to input daily hours and calculate the weekly total.
- **Challenge**: If they study over 10 hours on any single day, print "Warning: Burnout Alert!"