Case Study 1: Employee Salary Management

Scenario:

A company wants to store and analyze employee salary data. You need to create a system where:

1. Employees are stored in a dictionary with **employee ID as key** and **(name, department, salary)** as values.
2. Implement functions to:
    - **Increase the salary** of all employees in a particular department by a given percentage.
    - **Find the employee with the highest salary**.
    - **Filter employees** earning more than a given amount.

Hints:

- Use a **dictionary** to store employee data.
- Use **loops** to iterate and modify salaries.
- Use **tuple unpacking** when retrieving data.

Case Study 2: Student Performance Analysis

Scenario:

A school maintains student records with their scores in different subjects. You need to:

1. Store students in a **list of tuples** where each tuple contains (name, math_score, science_score, english_score).
2. Implement functions to:
    - **Find the student with the highest average marks**.
    - **Sort students based on total marks**.
    - **Find students who scored below 40 in any subject (failed students)**.

Hints:

- Use a **list of tuples** to store data.
- Use **loops** and **conditional statements** to process the data.
- Use the sorted() function to sort based on total marks.

Case Study 3: Unique Word Counter

Scenario:

You are given a paragraph of text. You need to:

1. **Extract all unique words** from the paragraph.
2. **Count the frequency of each word**.
3. Display the **top 5 most common words**.

Hints:

- Use a **set** to store unique words.
- Use a **dictionary** to maintain word counts.
- Use **loops** to iterate and count words.
- Use sorted() with lambda to get the top 5 words.

Case Study 4: E-commerce Product Catalog

Scenario:

You are developing a **product catalog system** for an e-commerce store. You need to:

1. Store products as a **dictionary**, where keys are **product IDs** and values are **(name, category, price)**.
2. Implement functions to:
    - Get all products in a specific category.
    - Find the **most expensive product**.
    - Identify and remove **duplicate product names** using a set.

Hints:

- Use a **dictionary** for product data.
- Use a **set** to remove duplicate names.
- Use **loops** and **conditional statements** to find the most expensive product.

Case Study 5: Movie Recommendation System

Scenario:

A streaming service wants to recommend movies based on user interests. You need to:

1. Store **users and their watched movies** as a dictionary:

```
users = {
   "Sam": {"Inception", "Titanic", "Avengers"},
   "Mohit": {"Inception", "Avatar", "Jurassic Park"},
```

```
    "Raj": {"Titanic", "Avatar", "Harry Potter"}
}
```

2. Implement functions to:
    - Find **common movies** between two users (movies they both watched).
    - Suggest **new movies** for a user based on what their friends watched.
    - Find **the most-watched movie** among all users.

Hints:

- Use **sets** to find common and suggested movies.
- Use **dictionaries** for user-movie mapping.
- Use **loops** to analyze data.