Problem Statements: Tuple

1. Employee Records Management

Given a tuple of employee records, where each record is a tuple with (employee_id, name, age, department), write a program to find all employees in a specific department and print their details as a list of tuples.

Sample Input:

```
employees = (
    (1, 'Alice', 30, 'HR'),
    (2, 'Bob', 25, 'Engineering'),
    (3, 'Charlie', 28, 'Engineering'),
    (4, 'Diana', 32, 'Marketing')
)
department_to_find = 'Engineering'
```

Sample Output:

```
[(2, 'Bob', 25, 'Engineering'), (3, 'Charlie', 28, 'Engineering')]
```

2. Store Inventory System

Create a tuple to store inventory items, where each item is a tuple with (item_id, name, quantity, price). Write a program to update the quantity of a given item and print the updated inventory as a tuple.

Sample Input:

```
inventory = (
    (101, 'Apples', 50, 0.5),
    (102, 'Bananas', 30, 0.2),
    (103, 'Cherries', 20, 1.5)
)
item_id_to_update = 102
new_quantity = 45
```

Sample Output:

```
(
    (101, 'Apples', 50, 0.5),
    (102, 'Bananas', 45, 0.2),
    (103, 'Cherries', 20, 1.5)
)
```

3. Student Grades Analysis

Given a tuple of student records, where each record is a tuple with (student_id, name, grades) and grades is a list of integers, write a program to calculate the average grade for each student and print a list of tuples with (student_id, name, average_grade).

Sample Input:

```
students = (
    (1, 'Alice', [90, 85, 88]),
    (2, 'Bob', [78, 82, 84]),
    (3, 'Charlie', [92, 91, 89])
)
```

Sample Output:

```
[(1, 'Alice', 87.67), (2, 'Bob', 81.33), (3, 'Charlie', 90.67)]
```

4. Library Catalog

Create a tuple representing a library catalog, where each book is a tuple with (book_id, title, author, genre). Write a program to search for books by a specific author and print the list of matching books.

Sample Input:

```
catalog = (
    (1001, 'The Great Gatsby', 'F. Scott Fitzgerald', 'Fiction'),
    (1002, '1984', 'George Orwell', 'Dystopian'),
    (1003, 'To Kill a Mockingbird', 'Harper Lee', 'Fiction'),
    (1004, 'Animal Farm', 'George Orwell', 'Satire')
)
author_to_find = 'George Orwell'
```

Sample Output:

```
[
    (1002, '1984', 'George Orwell', 'Dystopian'),
    (1004, 'Animal Farm', 'George Orwell', 'Satire')
]
```

5. Movie Database

Given a tuple of movies, where each movie is a tuple with (movie_id, title, director, rating), write a program to find the highest-rated movie and print its details.

Sample Input:

```
movies = (
    (201, 'Inception', 'Christopher Nolan', 8.8),
    (202, 'The Godfather', 'Francis Ford Coppola', 9.2),
    (203, 'Pulp Fiction', 'Quentin Tarantino', 8.9)
)
```

Sample Output:

```
(202, 'The Godfather', 'Francis Ford Coppola', 9.2)
```

6. Flight Schedule Management

Create a tuple to store flight schedules, where each flight is a tuple with (flight_id, origin, destination, departure_time). Write a program to find all flights between two cities and print the list of matching flights.

Sample Input:

```
flights = (
    (301, 'New York', 'Los Angeles', '08:00'),
    (302, 'Chicago', 'New York', '09:00'),
    (303, 'New York', 'Chicago', '10:00'),
    (304, 'Los Angeles', 'Chicago', '11:00')
)
origin_city = 'New York'
destination_city = 'Chicago'
```

Sample Output:

```
[(303, 'New York', 'Chicago', '10:00')]
```

7. Shopping Cart System

Given a tuple of items in a shopping cart, where each item is a tuple with (item_id, name, price, quantity), write a program to calculate the total cost of the cart and print it.

Sample Input:

```
cart = (
    (401, 'Laptop', 999.99, 1),
    (402, 'Mouse', 49.99, 2),
    (403, 'Keyboard', 79.99, 1)
)
```

Sample Output:

```
1179.96
```

8. Weather Data Analysis

Create a tuple representing weather data for a week, where each day's data is a tuple with (day, temperature, humidity). Write a program to find the day with the highest temperature and print its details.

Sample Input:

```
weather = (
    ('Monday', 75, 65),
    ('Tuesday', 80, 70),
    ('Wednesday', 78, 68),
    ('Thursday', 85, 72),
    ('Friday', 82, 67)
)
```

Sample Output:

```
('Thursday', 85, 72)
```

9. Restaurant Menu Management

Given a tuple of menu items, where each item is a tuple with (item_id, name, price, category), write a program to update the price of a specific item and print the updated menu as a tuple.

Sample Input:

```
menu = (
    (501, 'Burger', 5.99, 'Main Course'),
    (502, 'Fries', 2.99, 'Side Dish'),
    (503, 'Soda', 1.49, 'Beverage')
)
item_id_to_update = 502
new_price = 3.49
```

Sample Output:

```
(
    (501, 'Burger', 5.99, 'Main Course'),
    (502, 'Fries', 3.49, 'Side Dish'),
    (503, 'Soda', 1.49, 'Beverage')
)
```

10. Sports Team Roster

Create a tuple representing a sports team roster, where each player is a tuple with (player_id, name, position, goals_scored). Write a program to find the top scorer and print their details.

Sample Input:

```
team = (
    (601, 'John', 'Forward', 10),
    (602, 'Mike', 'Midfielder', 7),
    (603, 'Dave', 'Defender', 3)
)
```

Sample Output:

```
(601, 'John', 'Forward', 10)
```

11. Bank Account Transactions

Given a tuple of transactions, where each transaction is a tuple with (transaction_id, account_id, amount, transaction_type), write a program to calculate the balance for a specific account and print it.

Sample Input:

```
transactions = (
    (701, 1001, 200.0, 'deposit'),
    (702, 1002, 150.0, 'withdrawal'),
    (703, 1001, 300.0, 'deposit'),
    (704, 1001, 100.0, 'withdrawal')
)
account_id_to_check = 1001
```

Sample Output:

```
400.0
```

12. Online Course Enrollment

Create a tuple to store course enrollment, where each enrollment is a tuple with (course_id, student_id, enrollment_date). Write a program to find all students enrolled in a specific course and print their details.

Sample Input:

```
enrollments = (
    (801, 2001, '2024-01-15'),
    (802, 2002, '2024-01-16'),
    (801, 2003, '2024-01-17'),
    (803, 2001, '2024-01-18')
)
course_id_to_find = 801
```

Sample Output:

```
[(801, 2001, '2024-01-15'), (801, 2003, '2024-01-17')]
```

13. Bookstore Sales Analysis

Given a tuple of sales records, where each record is a tuple with (sale_id, book_id, quantity, sale_date), write a program to calculate the total sales for a specific book and print it.

Sample Input:

```
sales = (
    (901, 3001, 2, '2024-02-01'),
    (902, 3002, 1, '2024-02-02'),
    (903,

 3001, 3, '2024-02-03')
)
book_id_to_check = 3001
```

Sample Output:

```
5
```

14. Conference Schedule Management

Create a tuple representing a conference schedule, where each session is a tuple with (session_id, title, speaker, time). Write a program to find all sessions by a specific speaker and print the list of matching sessions.

Sample Input:

```
schedule = (
    (1001, 'AI in 2024', 'Dr. Smith', '10:00'),
    (1002, 'Blockchain Basics', 'Dr. Jones', '11:00'),
    (1003, 'Future of AI', 'Dr. Smith', '14:00')
)
speaker_to_find = 'Dr. Smith'
```

Sample Output:

```
[
    (1001, 'AI in 2024', 'Dr. Smith', '10:00'),
    (1003, 'Future of AI', 'Dr. Smith', '14:00')
]
```

15. E-commerce Order Management

Given a tuple of orders, where each order is a tuple with (order_id, customer_id, items) and items is a list of tuples with (item_id, quantity), write a program to calculate the total items ordered by a specific customer and print it.

Sample Input:

```
orders = (
    (1101, 4001, [(501, 2), (502, 1)]),
    (1102, 4002, [(503, 5), (504, 2)]),
    (1103, 4001, [(501, 1), (505, 3)])
)
customer_id_to_check = 4001
```

Sample Output:

```
7
```

16. Patient Records Management

Create a tuple to store patient records, where each record is a tuple with (patient_id, name, age, diagnosis). Write a program to find all patients with a specific diagnosis and print their details.

Sample Input:

```
patients = (
    (1201, 'Alice', 45, 'Flu'),
    (1202, 'Bob', 50, 'Diabetes'),
    (1203, 'Charlie', 37, 'Flu'),
    (1204, 'Diana', 29, 'Allergy')
)
diagnosis_to_find = 'Flu'
```

Sample Output:

```
[(1201, 'Alice', 45, 'Flu'), (1203, 'Charlie', 37, 'Flu')]
```

17. University Class Schedule

Given a tuple of class schedules, where each class is a tuple with (class_id, course, instructor, time), write a program to find all classes taught by a specific instructor and print the list of matching classes.

Sample Input:

```
classes = (
    (1301, 'Math 101', 'Prof. Lee', '09:00'),
    (1302, 'Physics 101', 'Prof. Smith', '10:00'),
    (1303, 'Chemistry 101', 'Prof. Lee', '11:00')
)
instructor_to_find = 'Prof. Lee'
```

Sample Output:

```
[
    (1301, 'Math 101', 'Prof. Lee', '09:00'),
    (1303, 'Chemistry 101', 'Prof. Lee', '11:00')
]
```

18. Real Estate Listings

Create a tuple representing real estate listings, where each listing is a tuple with (listing_id, property_type, location, price). Write a program to find the most expensive property and print its details.

Sample Input:

```
listings = (
    (1401, 'Apartment', 'New York', 1000000),
    (1402, 'House', 'Los Angeles', 1500000),
    (1403, 'Condo', 'Chicago', 750000)
)
```

Sample Output:

```
(1402, 'House', 'Los Angeles', 1500000)
```

19. Music Playlist Management

Given a tuple of songs in a playlist, where each song is a tuple with (song_id, title, artist, duration), write a program to find the total duration of the playlist and print it.

Sample Input:

```
playlist = (
    (1501, 'Song A', 'Artist 1', 3.5),
    (1502, 'Song B', 'Artist 2', 4.0),
    (1503, 'Song C', 'Artist 1', 5.2)
)
```

Sample Output:

```
12.7
```

20. Grocery Store Inventory

Create a tuple to store grocery items, where each item is a tuple with (item_id, name, category, stock_quantity). Write a program to find all items in a specific category and print the list of matching items.

Sample Input:

```
grocery_items = (
    (1601, 'Milk', 'Dairy', 20),
    (1602, 'Cheese', 'Dairy', 15),
    (1603, 'Bread', 'Bakery', 25),
    (1604, 'Butter', 'Dairy', 10)
)
category_to_find = 'Dairy'
```

Sample Output:

```
[
    (1601, 'Milk', 'Dairy', 20),
    (1602, 'Cheese', 'Dairy', 15),
    (1604, 'Butter', 'Dairy', 10)
]
```

Problem Statement: Company Hierarchy and Salary Analysis (Master Problem)

You are given a tuple representing a company's hierarchy and employee information. Each employee is represented by a tuple with the following structure:

```
(employee_id, name, position, manager_id, salary, projects)
```

- `employee_id`: Unique identifier for the employee (integer).
- `name`: Name of the employee (string).
- `position`: Job position of the employee (string).
- `manager_id`: Employee ID of the employee's manager (integer). If the employee is the CEO, the `manager_id` will be `None`.
- `salary`: Annual salary of the employee (float).
- `projects`: A list of project names the employee is currently working on (list of strings).

Tasks:

1. Find the total number of employees in the company.
2. Find the total salary expenditure of the company.
3. Find the employee with the highest salary and print their details.
4. Find all employees who are working on a specific project and print their details.
5. Create a list of tuples representing the hierarchy of the company, where each tuple is in the format `(manager_name, employee_name)`.
6. Create a list of tuples where each tuple contains a project name and a list of names of employees working on that project.

Sample Input:

```
employees = (
    (1, 'Alice', 'CEO', None, 250000.0, ['Project X', 'Project Y']),
    (2, 'Bob', 'CTO', 1, 200000.0, ['Project X']),
    (3, 'Charlie', 'CFO', 1, 190000.0, ['Project Y']),
    (4, 'Diana', 'Engineer', 2, 120000.0, ['Project X', 'Project Z']),
    (5, 'Eve', 'Engineer', 2, 110000.0, ['Project Z']),
    (6, 'Frank', 'Accountant', 3, 80000.0, ['Project Y'])
)
project_to_find = 'Project X'
```

Expected Output:

```
# 1. Total number of employees
6

# 2. Total salary expenditure
950000.0
```

```
# 3. Employee with the highest salary
(1, 'Alice', 'CEO', None, 250000.0, ['Project X', 'Project Y'])

# 4. Employees working on 'Project X'
[(1, 'Alice', 'CEO', None, 250000.0, ['Project X', 'Project Y']),
 (2, 'Bob', 'CTO', 1, 200000.0, ['Project X']),
 (4, 'Diana', 'Engineer', 2, 120000.0, ['Project X', 'Project Z'])]

# 5. Company hierarchy
[('Alice', 'Bob'),
 ('Alice', 'Charlie'),
 ('Bob', 'Diana'),
 ('Bob', 'Eve'),
 ('Charlie', 'Frank')]

# 6. List of projects with employees
[
    ('Project X', ['Alice', 'Bob', 'Diana']),
    ('Project Y', ['Alice', 'Charlie', 'Frank']),
    ('Project Z', ['Diana', 'Eve'])
]
```