**Problem 1: Social Network Analysis**

**Scenario:** You are working on a social networking platform. One of the features is to suggest new friends to users based on mutual friends. Given a graph where nodes represent users and edges represent friendships, you need to find the top k friend suggestions for a user based on the number of mutual friends.

**Description:** Write a script that takes a graph represented by an adjacency list, a user u, and a number k. The script should output the top k users who have the most mutual friends with user u.

**Sample Input:**

graph = {

'Alice': ['Bob', 'Charlie', 'David'],

'Bob': ['Alice', 'Charlie', 'Eve'],

'Charlie': ['Alice', 'Bob', 'Eve', 'David'],

'David': ['Alice', 'Charlie', 'Eve'],

'Eve': ['Bob', 'Charlie', 'David']

}

user = 'Alice'

k = 2

**Sample Output:**

['Eve', 'Charlie']

**Problem 2: Job Scheduler**

**Scenario:** You are developing a job scheduling system for a cloud computing platform. Each job has a start time, an end time, and a profit. You need to find the maximum profit that can be earned by scheduling non-overlapping jobs.

**Description:** Write a script that takes a list of jobs where each job is represented as a tuple (start, end, profit). The script should output the maximum profit that can be earned by scheduling non-overlapping jobs.

**Sample Input:**

jobs = [(1, 3, 50), (3, 5, 20), (0, 6, 60), (5, 7, 30), (5, 9, 50), (7, 8, 10)]

**Sample Output:**

80

**Problem 3: Inventory Management**

**Scenario:** You are working on an e-commerce platform's inventory management system. The system needs to track the stock of items and automatically reorder items when their stock falls below a certain threshold. You need to implement a function that simulates the reordering process.

**Description:** Write a script that takes a dictionary representing the inventory (item: stock), a reorder threshold, and a reorder quantity. The script should update the inventory by reordering items that are below the threshold and print the updated inventory.

**Sample Input:**

inventory = {'item1': 5, 'item2': 2, 'item3': 12}

threshold = 3

reorder\_quantity = 10

**Sample Output:**

{'item1': 5, 'item2': 12, 'item3': 12}

**Problem 4: Flight Itinerary Planner**

**Scenario:** You are building a flight itinerary planner. Given a list of flights, where each flight is represented as a tuple (start, end), you need to determine if there exists an itinerary that uses all the flights exactly once and starts and ends at specified airports.

**Description:** Write a script that takes a list of flights and two airports (start and end). The script should output True if there exists an itinerary that uses all the flights exactly once and starts and ends at the specified airports, otherwise False.

**Sample Input:**

flights = [('JFK', 'ATL'), ('ATL', 'SFO'), ('SFO', 'JFK'), ('JFK', 'LAX'), ('LAX', 'SFO')]

start = 'JFK'

end = 'SFO'

**Sample Output:**

True

**Sample Input:**

flights = [('JFK', 'ATL'), ('ATL', 'SFO'), ('SFO', 'JFK'), ('JFK', 'LAX'), ('LAX', 'ORD')]

start = 'JFK'

end = 'SFO'

**Sample Output:**

False

**Problem 5: Machine Learning Model Deployment**

**Scenario:** You are responsible for deploying a machine learning model in a production environment. The model needs to make real-time predictions based on incoming data and store the results in a database. You need to implement a part of this system that handles incoming data, makes predictions, and stores the results.

**Description:** Write a script that simulates receiving incoming data, making predictions using a pre-trained model (simulated with a simple function), and storing the results in a database (simulated with a dictionary). The script should handle multiple data points and print the database after processing all the data.

**Sample Input:**

data = [{'id': 1, 'features': [0.1, 0.2, 0.3]}, {'id': 2, 'features': [0.4, 0.5, 0.6]}]

**Pre-trained Model Simulation:**

def predict(features):

return sum(features) # Simple sum function as a placeholder for a real model

**Sample Output:**

{1: 0.6, 2: 1.5}