**Topic 1 – Review JS ES6**

**📦 Sample JSON Datasets**

**1. users.json**

[

{ "id": 1, "name": "Alice", "email": "alice@example.com" },

{ "id": 2, "name": "Bob", "email": "bob@example.com" },

{ "id": 3, "name": "Charlie", "email": "charlie@example.com" }

]

**2. products.json**

[

{ "id": 101, "name": "Laptop", "price": 1200 },

{ "id": 102, "name": "Phone", "price": 800 },

{ "id": 103, "name": "Tablet", "price": 600 }

]

**3. orders.json**

[

{ "id": 1001, "userId": 1, "productId": 101, "quantity": 1 },

{ "id": 1002, "userId": 2, "productId": 102, "quantity": 2 },

{ "id": 1003, "userId": 1, "productId": 103, "quantity": 1 },

{ "id": 1004, "userId": 3, "productId": 102, "quantity": 3 }

]

**🧠 10 Exercises with Increasing Difficulty**

**Exercise 1: List all user names**

**Goal:** Use map to extract all names from users.

// Output: ["Alice", "Bob", "Charlie"]

**Exercise 2: Find the product with the highest price**

**Goal:** Use reduce to find the most expensive product.

// Output: { id: 101, name: "Laptop", price: 1200 }

**Exercise 3: Get orders made by "Alice"**

**Goal:** Combine users and orders data to filter only orders by Alice.

// Output: Orders where userId === 1

**Exercise 4: Calculate total revenue**

**Goal:** For each order, compute price \* quantity and sum all values.

// Output: Total revenue from all orders

**Exercise 5: Create a summary for each order**

**Goal:** Create an array of objects like:

[

{

orderId: 1001,

userName: "Alice",

productName: "Laptop",

quantity: 1,

totalPrice: 1200

},

...

]

**Exercise 6: List users who bought more than one item**

**Goal:** Find users who have a total order quantity > 1.

**Exercise 7: Group orders by user**

**Goal:** Create an object with user names as keys and their orders as arrays of product names.

// Output:

// {

// Alice: ["Laptop", "Tablet"],

// Bob: ["Phone"],

// Charlie: ["Phone"]

// }

**Exercise 8: Add a new product and update orders accordingly**

**Goal:** Use the spread operator to add a new product, then create a new order with it (immutably).

**Exercise 9: Use destructuring to extract product info**

**Goal:** Given a product object, use destructuring to extract name and price.

const product = { id: 105, name: "Monitor", price: 300 };

**Exercise 10: Async/Await - Simulate fetching and linking data**

**Goal:** Simulate fetching data from 3 endpoints and link orders with user and product using async/await.

// Expected: A list of orders with userName and productName added