Here’s a slightly tougher assignment while still keeping it manageable for beginners.

Assignment: Working with Array Methods and Loops

1. For Loop & Reduce (Combination)

- You have an array of student scores:
- `const scores = [80, 95, 78, 88, 92, 67, 75, 89, 100, 55];`
    - Use a `for` loop to find the highest score in the array.
    - Use `reduce` to find the total sum of all scores and calculate the average.

2. Reduce & Map (Advanced Transformation)

- Given an array of products:
- `const products = [`
- `  { name: "Laptop", price: 1500 },`
- `  { name: "Phone", price: 700 },`
- `  { name: "Tablet", price: 300 },`
- `  { name: "Monitor", price: 400 },`
- `];`
    - Use `reduce` to calculate the total cost of all products.
    - Use `map` to create a new array that adds a `"discountedPrice"` field to each product, where the discount is 10% off the original price.

3. ForEach & Map (String Manipulation)

- Given an array of people's full names:
- `const names = ["john doe", "jane smith", "alice wonderland", "bob builder"];`
    - Use `map` to return an array where each name is properly capitalized (e.g., `"John Doe"`).
    - Use `forEach` to log each person's initials (e.g., `"JD"` for `"John Doe"`).

4. Filter & Sort (Complex Filtering & Sorting)

- Given an array of employees:
- `const employees = [`
- `  { name: "Michael", age: 45, salary: 5000 },`
- `  { name: "Sarah", age: 30, salary: 7000 },`
- `  { name: "David", age: 25, salary: 4500 },`
- `  { name: "Emily", age: 28, salary: 5500 },`
- `  { name: "John", age: 35, salary: 6000 },`
- `];`
    - Use `filter` to get employees who earn more than `5000`.
    - Use `filter` to get employees younger than `30`.

- Use `sort` to arrange employees by their salary in descending order.

5. Combination Challenge (Real-Life Example)

- You are given a list of transactions in a bank account:
- ```
  const transactions = [
    { type: "deposit", amount: 1000 },
    { type: "withdrawal", amount: 500 },
    { type: "deposit", amount: 1200 },
    { type: "withdrawal", amount: 300 },
    { type: "deposit", amount: 400 },
    { type: "withdrawal", amount: 700 },
  ];
  ```
  - Use `reduce` to find the total balance (start from 0).
  - Use `filter` to get all deposit transactions.
  - Use `map` to create a new array that includes each transaction but adds a `"status"` field that shows `"completed"` for deposits and `"pending"` for withdrawals.

  ```