Rules

- The use of AI tools like ChatGPT, GitHub Copilot, or any other AI-based assistance is strictly forbidden.
- You can use your handwritten or digital notes.
- Any form of browsing the internet is not allowed (Google, MDN, Stack Overflow, etc.).
- Do not share or discuss answers with others during the exam.
- Anyone caught sharing solutions will face strict action.
- Complete the exam honestly.

Short Questions: 3 * 5 ⇒ 15

1. Explain the difference between var, let, and const in terms of scope and hoisting. Provide an example where var behaves differently from let and const.

2. What is a closure in JavaScript? How does lexical scope enable closures? Provide an example of a closure where an inner function retains access to variables from its outer function.

3. What is the difference between map(), filter(), and reduce()?

4. Explain how JavaScript handles asynchronous operations using Promises and async/await.

5. How does JavaScript handle type coercion when using == vs. ===? Explain with examples where == results in unexpected behavior due to implicit type conversion.

JSON: 3*5 + 14*5 ⇒ 85

1. Answer the following questions using the provided JSON (questions a-e are worth 3 marks; questions f-j are worth 14 marks).

   a. Write a function that takes a category (e.g., "Electronics") as input and returns an array of product names (strings) for that category, sorted alphabetically. Example output for

"Electronics": ["Bluetooth Speaker", "Digital Camera", "Laptop Pro X", "Smart Watch", "Wireless Headphones"]

b. Write a function that returns an array of objects containing each product’s name and its average rating (rounded to 2 decimal places) based on the reviews. Example output: [{ name: "Laptop Pro X", avgRating: 4.00 }, { name: "Wireless Headphones", avgRating: 3.50 }, ...]

c. Write a function that takes a feature keyword (e.g., "Bluetooth") and returns an array of product IDs for products whose features array contains that keyword (case-insensitive). Example for "Bluetooth": ["P1002", "P1010"]

d. Write a function that groups products into three categories based on stock: "Low" (< 50), "Medium" (50-100), and "High" (> 100). Return an object where each key is a stock level and the value is an array of product names. Example output: { Low: ["Cookbook: World Cuisine", "Digital Camera"], Medium: [...], High: ["Cotton T-Shirt"] }

e. Write a function that returns all unique user IDs from the reviews across all products. Example output: {"U101", "U102", "U103", ..., "U141"}

f. Write a function that finds the top 3 products with the highest average ratings, but only includes products with at least 4 reviews. Return an array of objects with product_id, name, and avgRating (sorted by avgRating descending). Example output: [{ product_id: "P1003", name: "Running Shoes", avgRating: 4.20 }, ...]

g. Write a function that analyzes all review comments and returns an object where the keys are words (lowercase, no punctuation) and the values are the number of times they appear across all comments. Ignore words shorter than 3 characters. Example output: {"great" ⇒ 6, "good" ⇒ 5, "comfortable" ⇒ 4, ...}

h. Write a function that identifies products with "mixed reviews," defined as having at least one rating of 5 and at least one rating of 2 or lower. Return an array of product names. Example output: ["Wireless Headphones", "Smart Watch"]

i. Write a function that calculates the average price of products that have a specific feature (e.g., "Water Resistant") and compares it to the overall average price of all products. Return an object with both averages and the difference. Example output for "Water Resistant": { featureAvg: 165.00, overallAvg: 242.80, difference: -77.80 }

j. Write a function that categorizes products based on their review comments. For each product, count how many comments contain positive words ("great", "excellent", "good", "amazing", "comfortable") versus negative words ("poor", "bad", "heavy", "complex"). Return an object like this { "P1001": { positiveCount: 3, negativeCount: 1 }, "P1002": { positiveCount: 2, negativeCount: 2 }, ... }