1. **Variables, Data Types, and String Manipulation**

Create a function that accepts user input (name, age, and email), validates the input types, and returns a formatted string: "Name: [name], Age: [age], Email: [email]". Ensure the email includes "@".

1. **Arrays & Object:**

Create a JavaScript object representing a library system. The object should include properties for the library name and an array of books. Each book object should have the following properties: title, author, isbn, and isAvailable (a Boolean indicating availability).

Implement the following functions that work with this library object:

* **Add a Book**:  
  Create a function addBook(library, book) that adds a new book to the books array of the library. Console log "Book added" after adding the book.
* **Check Out a Book**:  
  Create a function checkOutBook(library, isbn) that sets isAvailable to false for the book with the specified isbn. If the book is not found, display "Book not found in the library".
* **Return a Book**:  
  Create a function returnBook(library, isbn) that sets isAvailable to true for the book with the specified isbn. If the book is not found, display "Book not found in the library".
* **List All Book Titles**:  
  Create a function listAllBookTitles(library) that lists all book titles using the map method. Use forEach to print each title to the console.
* **List Available Books**:  
  Create a function listAvailableBooks(library) that lists titles of only the available books using filter and map. Print each available book title to the console.
* **Count Books by Author**:  
  Create a function countBooksByAuthor(library, author) that uses reduce to count how many books are authored by the given author. Return the count.

**Hints:**

* + Use the **map** method to extract book titles and the **forEach** method to print them.
  + Use the **filter** method to find available books before using map.
  + Use the **reduce** method to accumulate the count of books by a specific author

1. **ES6+ Features**

Write a function that takes an object with nested properties (e.g., user profile) and uses destructuring to extract specific properties. Use the spread operator to create a new object with some updated properties.

1. **Asynchronous JavaScript & Error Handling**

Write a JavaScript function that performs the following tasks:

* **Fetch Data**:  
  Use async/await to fetch data from the public API endpoint **https://jsonplaceholder.typicode.com/posts**.
* **Process and Validate Data**:  
  Extract the title property from each post and create an array of titles. Validate this array to ensure it contains only strings.
* **Error Handling**:  
  Handle any errors that occur during the fetch operation or data processing. Log an appropriate error message if something goes wrong.

**Function Requirements:**

* **Function Signature**: async function fetchAndProcessTitles()
* **Use async/await** for fetching data and handle errors with try/catch.
* **Validate** that the titles array contains only strings.
* **Log** the array of titles or an error message as appropriate.

## Advanced Questions

**Real-World Scenario: Product Data Management**

You are tasked with building a small backend feature that handles product data for an e-commerce platform. You have access to a public API that provides product details in JSON format.

**Your tasks are as follows:**

1. **Fetch and Process Product Data:**
   * Use the Fakestore API to fetch a list of products.
   * URL: https://fakestoreapi.com/products
   * Use promises to handle the fetching operation.
   * Implement error handling to catch any issues during the fetch process and log an appropriate message.
2. **Calculate Average Rating:**
   * Once the data is fetched, write a function to calculate the average rating of all products.
   * Display the average rating, rounded to two decimal places.
3. **Filter Products by Category:**
   * Write a function that filters the products based on a given category (e.g., "men's clothing").
   * The function should return an array of product titles in the specified category.
4. **Find the Most Expensive Product:**
   * Implement a function that finds and returns the product with the highest price.
5. **Handle Empty and Invalid Data:**
   * Ensure your functions handle scenarios where the product list is empty or contains invalid data gracefully.

**Hints:**

* Use array methods like map, filter, and reduce to process the product data.
* Consider using the try/catch block to handle errors in the async functions.