Day 1 - JavaScript and TypeScript Fundamentals

Hour 1-2: Introduction to JavaScript and TypeScript

- **Topics:**
  - JavaScript Overview: Understand the history, uses, and evolution of JavaScript in web development.
  - TypeScript Introduction: Explore the advantages of TypeScript, including static typing and basic syntax.
- **Exercise:**
  - **Set up a TypeScript Environment:**
    - Install Node.js and TypeScript globally on your system.
    - Create a TypeScript file `mathOperations.ts`.
    - Write a TypeScript function named `addNumbers` that takes two parameters and returns their sum.
    - Compile and run the TypeScript file to verify the function's output.
    - There is 2 way to run typescript file, first way to convert the typescript file to javascript and run as javascript file and second is directly running typescript file without converting it to javascript

Hour 3-4: Variables, Data Types, and Operators

- **Topics:**
  - Variables (`var`, `let`, `const`): Understand variable declaration and scoping rules.
  - Data Types (numbers, strings, booleans, null, undefined, objects): Learn about different data types in JavaScript and TypeScript.
  - Operators (arithmetic, comparison, and logical operators): Explore various operators and their usage.
- **Exercises:**
  - **Complex Variable Usage:**
    - Declare a variable `fullName` using `let` and concatenate it with another string to form a sentence.
    - Use template literals to create a dynamic message using variables.
  - **Advanced Data Manipulation:**
    - Write a function `calculatePrice` that takes the base price and applies a discount based on user type (regular, premium).
    - Use object literals to represent user types and their respective discounts.

Hour 5-6: Control Structures and Functions

- **Topics:**
  - Control Structures (`if` statements, `for` and `while` loops): Understand conditional and loop structures in JavaScript.
  - Functions (function declarations, expressions, arrow functions): Explore different ways to define functions.
- **Exercises:**
  - **Advanced Looping:**
    - Write a function `printMultiples` that takes a number and prints its multiples from 1 to 10.
    - Use a `for` loop to accomplish this task.
  - **Function Complexity:**
    - Implement a recursive function `calculateFactorial` that calculates the factorial of a number.
    - Write a higher-order function that takes a function and a parameter, and applies the function multiple times to the parameter.

Hour 7: Objects and Arrays in JavaScript

- **Topics:**
  - Objects (creating, accessing properties): Learn about object creation and property access in JavaScript.
  - Arrays (creating, accessing elements, array methods): Understand arrays and their various methods for data manipulation.
- **Exercises:**
  - **Object Manipulation:**
    - Create an object representing a book with properties like `title`, `author`, and `pages`.
    - Implement a function that takes an array of books and returns a new array sorted by the number of pages.
  - **Advanced Array Operations:**
    - Create an array of numbers and use `map` and `filter` to transform and filter the array, respectively.
    - Implement a function that flattens a nested array structure into a single array.

Hour 8: Advanced JavaScript and TypeScript Concepts

- **Topics:**

  - Closures: Understand the concept of closures in JavaScript and their practical applications.
  - `this` Keyword: Learn about the context-binding behavior of the `this` keyword.
  - TypeScript Advanced Concepts (union types, intersection types, generics): Explore advanced type system features in TypeScript.
- **Exercises:**
  - **Closure Implementation:**
    - Write a function `counter` that returns a function. The inner function, when invoked, should increment a counter variable in the outer function's scope.
    - Demonstrate closure behavior by creating multiple counters and invoking them independently.
  - **TypeScript Generics:**
    - Implement a generic function that takes an array of any type and returns an array containing only unique elements.
    - Test the function with different data types (numbers, strings, objects) to ensure type safety.

These exercises are designed to challenge your understanding of JavaScript and TypeScript concepts. Spend time experimenting, debugging, and improving your solutions. Good luck with your learning!