2-week Roadmap

Week 1: Core JavaScript Concept

Day 1: Variables and Data Types

- Learn about `var`, `let`, `const`.

- Data types: numbers, strings, booleans, null, undefined, and symbols.

- Practice converting between data types (type coercion).

Practice: Create a simple program where you define variables of different types and output their values.

Day 2: Operators

- Arithmetic operators (`+`, `-`, `*`, `/`, `%`).

- Assignment and comparison operators (`=`, `==`, `===`, `!=`, `>`, `<`).

- Logical operators (`&&`, `||`, `!`).

Practice: Write a program that uses operators to perform basic calculations and comparisons.

Day 3: Conditional Statements

- If-else statements.

- Switch-case.

Practice: Create a program that takes user input and outputs different responses based on the conditions.

Day 4: Loops

- Learn `for`, `while`, `do-while` loops.

Practice: Write a program that loops through a list of numbers and outputs whether they are odd or even.

Day 5: Functions

- Function declarations, expressions, and arrow functions.

- Parameters and return values.

Practice: Write a program where you define and call functions to perform specific tasks (e.g., a function that calculates the sum of two numbers).

Day 6: Objects and Arrays

- Learn how to create and manipulate objects and arrays.

- Array methods (`push`, `pop`, `map`, `filter`, etc.).

Practice: Write a program that stores user data (like name, age) in an object, and create an array of objects.

Day 7: Project (Core Concepts)

- **Mini Project**: Build a small application (e.g., a basic calculator or a task manager) that uses variables, loops, functions, objects, and arrays.

Week 2: Advanced JavaScript Concepts

Day 8: Higher-Order Functions

- Functions as first-class citizens, function callbacks.

- `map()`, `filter()`, `reduce()`.

Practice: Use higher-order functions to manipulate an array of numbers (e.g., filtering even numbers, calculating the sum).

Day 9: Asynchronous JavaScript (Callbacks and Promises)

- Learn about synchronous vs. asynchronous JavaScript.

- Callbacks and Promises: how to handle asynchronous operations.

Practice: Create a simple program that simulates an asynchronous task using `setTimeout()` and handle it with a callback and Promise.

Day 10: Async/Await

- Learn how to simplify working with Promises using `async` and `await`.

Practice: Write a program that simulates fetching data from an API using `async/await`.

Day 11: Event Handling- Learn

n how to handle DOM events (click, submit, etc.).

- Event listeners (`addEventListener`).

Practice: Add interactivity to a webpage where clicking a button shows a message or toggles visibility of an element.

Day 12: Error Handling

- Try-catch blocks.

- Throwing and handling errors.

Practice: Write a program where you deliberately cause and catch an error (e.g., trying to divide by zero) and display an error message.

Day 13: Project (Advanced Concepts)

- **Mini Project**: Build an interactive application (like a quiz app or to-do list) that handles events, uses asynchronous operations, and incorporates higher-order functions.

Day 14: Final Project

- Final Project:- Combine everything you've learned by building a more complex web application.