■ Week 2 Exercises – Step-by-Step Instructions

Use these steps to guide you through each task. Try your best before asking for help!

■ function-exercises/

■ ex1: Write a function that receives a number and returns its square. After that, try writing the same function again using an arrow function.

■ ex2: Create a function that receives a number and prints double the value (no return needed). Then try it again using an arrow function.

■ ex3: Make a function with no parameters that returns the value of Pi. Then rewrite it using an arrow function.

■ ex4: Write a function with no parameter and no return that prints "Hello!" to the console. Do the same again using an arrow function.

■ template-literal-exercises/

■ ex1: Create a variable for a name, and use a template literal to say Hello to that name.

■ ex2: Make two variables: one for an item, and one for its price. Use a template literal to describe the item and its price.

■ ex3: Use a template literal to print a message that spans multiple lines.

■ ex4: Create two number variables and use a template literal to show the sum.

■ modules-exercises/

■ export.js

- Create four functions: addNumbers, subtractNumbers, multiplyNumbers, divideNumbers.
- Each function should take two parameters and return the result.
- Export all of them using different styles of export.

■ import.js

- Import all four functions from export.js.
- Call each function with test values.
- Print out each result.

■ destructuring-exercises/

■ ex1 – Object Destructuring

- Make an object with properties: name, age, and city.
- Use destructuring to get the values.
- Print those values to the console.

■ ex2 – Array Destructuring

- Create an array with at least three items (e.g. fruits).
- Use destructuring to get the first two items.
- Print them.

■ async-await-exercise/

■ exercise.js

- Call the fetchBankAccountBalanceSimulated function using await.
- Store the result in a variable.
- Print: "Your bank account balance is ${balance} USD" to the console.

■ big-exercise/

■ export.js

- Use any type of export to export the provided async function to import.js.
- The function simulates fetching student info and returns an object with extra properties after a delay.
- You do not need to change the code, just export the function.

■ import.js

- Import the async function from export.js.
- Call the function with a student info object and log the result.
- Try with different student info objects and observe the output.