**W08 Prepare – Q/A AI**

**Callback Functions**

 **Q: What is a callback function?**

* **A:** A function passed as an argument to another function. The calling function can then invoke the callback function at a specific point in its execution.

 **Q: What is the purpose of callback functions with JavaScript?**

* **A:**
  + **Handling Asynchronous Operations:** Allow code to continue executing while waiting for asynchronous operations to finish (like fetching data).
  + **Event Handling:** Define event listeners to react to user interactions and other events.

 **Q: When are callback functions most useful in JavaScript?**

* **A:** Examples include:
  + Fetching data from a server
  + Setting timeouts
  + Handling user interaction (clicks, form submissions)
  + Building complex asynchronous workflows

 **Q: What is an example of a poorly-designed callback function?**

* **A:** A function that makes assumptions about data types or lacks error handling, leading to potential issues.

 **Q: How do you use the typeof function for type checking with callback functions?**

* **A:** You can use typeof inside the function accepting the callback to ensure it receives a function. However, typeof has limitations, and consider alternatives like TypeScript or third-party libraries for more robust type checking.

**JS Array Methods**

 **Q: What is an array in programming?**

* **A:** An array in programming is a fundamental data structure that allows you to store a collection of elements of the same data type under a single variable name. Characteristics and benefits include:
  + Fixed size
  + Sequential, organized data storage
  + Indexing
  + Same data type
  + Efficient access
  + Iterating over data

 **Q: Why are there various array methods in JavaScript?**

* **A:** Array methods offer several benefits:
  + Improved readability and maintainability of code.
  + Reduced errors by handling common edge cases.
  + Promote functional programming style with immutability.
  + Enable code reusability for working with different arrays.
  + Allow chaining operations for complex tasks in a single line.

 **Q: How do you specify a fixed size when creating an array in JavaScript?**

* **A:** While JavaScript arrays don't have a true fixed size, you can achieve a similar effect with workarounds:
  + Pre-filling with undefined elements.
  + Using the spread syntax with fill to create an array of a desired size with placeholders.
  + Checking the length property before adding elements to avoid exceeding intended size (conceptually fixed).

 **Q: What is an example of a poorly-designed array?**

* **A:** A poorly-designed array mixes data types, has an unclear purpose, uses nested arrays unnecessarily, and lacks documentation. Separating data into well-defined arrays or objects with meaningful names improves maintainability.

 **Q: What are some good examples of using array methods in JavaScript?**

* **A:** Here are some common examples:
  + map: transforms each element in an array (e.g., calculating price with tax).
  + filter: creates a new array containing elements that pass a test condition (e.g., finding customers above a certain age).
  + forEach: iterates through the array and executes a provided function for each element (e.g., logging names).