Lab: Array Manipulation Web Service

Problem Description:

You will create a web service that performs basic array manipulation operations. The service will allow clients to send arrays of integers and perform various operations on them. The goal is to implement this service using two different architectures (XML/SOAP and pseudo-RESTful) and create clients in multiple programming languages to interact with it.

Array Operations:

1. Sorting an Array: Sort the input array of integers in ascending order.

2. Finding the Maximum Element: Return the maximum element from the input array.

3. Finding the Minimum Element: Return the minimum element from the input array.

4. Adding Two Arrays: Add two arrays element-wise and return a new array. Arrays must be of equal length, or an error is returned.

5. Multiplying Two Arrays: Multiply two arrays element-wise and return a new array. Arrays must be of equal length, or an error is returned.

Lab Steps:

Part 1: XML/SOAP Web Service Implementation

1. Server-Side Implementation:

- Implement the ArrayManipulatorService in Java using JAX-WS.
- The service should expose the following operations:
  - sortArray(int[] array): Sort an array of integers.
  - findMax(int[] array): Find the maximum element in an array.
  - findMin(int[] array): Find the minimum element in an array.
  - addArrays(int[] array1, int[] array2): Add two arrays element-wise.
  - multiplyArrays(int[] array1, int[] array2): Multiply two arrays element-wise.

2. WSDL Generation:

- Generate the WSDL for your web service using wsgen.

3. Write Client Code:

- **Python**: Write a Python client using the zeep module to interact with the SOAP service.
- **JavaScript**: Write a JavaScript client using the soap module.
- **Java**: Write a Java client by generating the stubs using wsimport.

4. Test the Clients:

- Run your webservice and test each client to ensure the array operations work correctly.

Part 2: Pseudo-RESTful Web Service Implementation

1. Server-Side Implementation:

- Implement the ArrayManipulatorService using Spring Boot and expose the same operations as REST endpoints:
  - POST /sortArray
  - POST /findMax
  - POST /findMin
  - POST /addArrays
  - POST /multiplyArrays

2. OpenAPI/Swagger Generation:

- Use Springdoc OpenAPI to generate the OpenAPI specification for your RESTful web service.

3. Python Client Generation:

- Use Swagger tools to generate a Python client from the OpenAPI specification.

4. Test the Python Client:

- Run your pseudo-RESTful service and test the generated Python client.

Operations Overview:

1. Sorting an Array:

- Input: [3, 1, 4, 1, 5]

- Output: [1, 1, 3, 4, 5]

3. Finding the Maximum Element:

- Input: [1, 2, 3, 4, 5]

- Output: 5

4. Finding the Minimum Element:

   - Input: [1, 2, 3, 4, 5]

   - Output: 1

2. Adding Two Arrays:

   - Input: [1, 2, 3], [4, 5, 6]

   - Output: [5, 7, 9]

5. Multiplying Two Arrays:

   - Input: [1, 2, 3], [4, 5, 6]

   - Output: [4, 10, 18]