**GALLETA, JHONELLE M. BSCS1A**

**A. Advanced Arrays in JavaScript**

**1. Advanced Types:**

In JavaScript, arrays are a special type of object that can hold multiple values in a single variable. Advanced array types include:

* **Multidimensional Arrays**: Arrays that contain other arrays as elements, allowing for the representation of matrices or grids.

**const matrix = [**

**[1, 2, 3],**

**[4, 5, 6],**

**[7, 8, 9]**

**];**

* **Sparse Arrays**: Arrays that do not have all their indices filled, which can lead to performance optimizations in certain scenarios.
* **Typed Arrays**: Arrays that provide a mechanism for accessing raw binary data buffers. They include types like **Int8Array**, **Uint8Array**, **Float32Array**, etc.

**2. Structuring and Destructuring:**

* **Structuring**: This refers to creating arrays and organizing data within them. For example:

**const fruits = ['apple', 'banana', 'cherry'];**

* **Destructuring**: This is a syntax that allows unpacking values from arrays into distinct variables.

**const colors = ['red', 'green', 'blue'];**

**const [firstColor, secondColor] = colors; // firstColor = 'red', secondColor = 'green'**

**3. Array Methods:**

* **map()**: Creates a new array populated with the results of calling a provided function on every element in the calling array.

**const numbers = [1, 2, 3];**

**const doubled = numbers.map(num => num \* 2); // [2, 4, 6]**

* **filter()**: Creates a new array with all elements that pass the test implemented by the provided function.

**const numbers = [1, 2, 3, 4, 5];**

**const evens = numbers.filter(num => num % 2 === 0); // [2, 4]**

* **reduce()**: Executes a reducer function on each element of the array, resulting in a single output value.

**const numbers = [1, 2, 3, 4];**

**const sum = numbers.reduce((acc, num) => acc + num, 0); // 10**

**4. Array Search Methods:**

* **indexOf()**: Returns the first index at which a given element can be found in the array, or -1 if it is not present.

**const fruits = ['apple', 'banana', 'cherry'];**

**const index = fruits.indexOf('banana'); // 1**

* **includes()**: Determines whether an array includes a certain value among its entries.

**const fruits = ['apple', 'banana', 'cherry'];**

**const hasBanana = fruits.includes('banana'); // true**

* **find()**: Returns the value of the first element in the array that satisfies the provided testing function.

**const numbers = [1, 2, 3, 4];**

**const found = numbers.find(num => num > 2); // 3**

**5. Array Sort Methods:**

* **sort()**: Sorts the elements of an array in place and returns the sorted array.

**const numbers = [4, 2, 3, 1];**

**numbers.sort(); // [1, 2, 3, 4]**

* **reverse()**: Reverses the elements of an array in place.

**const numbers = [1, 2, 3];**

**numbers.reverse(); // [3, 2, 1]**

* **sort() with a custom comparator**: Allows sorting based on specific criteria.

**const numbers = [4, 2, 3, 1];**

**numbers.sort((a, b) => a - b); // [1, 2, 3, 4]**

**6. Array Iteration Methods:**

* **forEach()**: Executes a provided function once for each array element.

**const numbers = [1, 2, 3];**

**numbers.forEach(num => console.log(num)); // 1, 2, 3**

* **every()**: Tests whether all elements in the array pass the test implemented by the provided function.

**const numbers = [1, 2, 3];**

**const allPositive = numbers.every(num => num > 0);**