common methods available for arrays in TypeScript:

1. **push**: Adds one or more elements to the end of an array and returns the new length of the array.
   ```typescript
   let numbers: number[] = [1, 2, 3];
   numbers.push(4, 5); // numbers is now [1, 2, 3, 4, 5]
   ```

2. **pop**: Removes the last element from an array and returns that element.
   ```typescript
   let numbers: number[] = [1, 2, 3];
   let lastElement = numbers.pop(); // lastElement is 3, numbers is now [1, 2]
   ```

3. **shift**: Removes the first element from an array and returns that element.
   ```typescript
   let numbers: number[] = [1, 2, 3];
   let firstElement = numbers.shift(); // firstElement is 1, numbers is now [2, 3]
   ```

4. **unshift**: Adds one or more elements to the beginning of an array and returns the new length of the array.
   ```typescript
   let numbers: number[] = [2, 3];
   numbers.unshift(0, 1); // numbers is now [0, 1, 2, 3]
   ```

5. **splice**: Adds or removes elements from an array.
   ```typescript
   let numbers: number[] = [1, 2, 3, 4, 5];
   numbers.splice(2, 1); // Removes 1 element at index 2, numbers is now [1, 2, 4, 5]
   ```

6. **slice**: Returns a shallow copy of a portion of an array into a new array.
   ```typescript
   let numbers: number[] = [1, 2, 3, 4, 5];
   let slicedArray = numbers.slice(2); // slicedArray is [3, 4, 5]
   ```

7. **concat**: Returns a new array that concatenates two or more arrays.
   ```typescript
   let numbers1: number[] = [1, 2];
   let numbers2: number[] = [3, 4];
   ```

   let concatenatedArray = numbers1.concat(numbers2); // concatenatedArray is [1, 2, 3, 4]

8. **indexOf**: Returns the first index at which a given element can be found in the array, or -1 if it is not present.
   ```typescript
   let numbers: number[] = [1, 2, 3, 4, 5];
   let index = numbers.indexOf(3); // index is 2
   ```

9. **forEach**: Executes a provided function once for each array element.
   ```typescript
   let numbers: number[] = [1, 2, 3];
   numbers.forEach(num => console.log(num)); // Logs 1, 2, 3
   ```

10. **map**: Creates a new array populated with the results of calling a provided function on every element in the array.
   ```typescript
   let numbers: number[] = [1, 2, 3];
   let doubledNumbers = numbers.map(num => num * 2); // doubledNumbers is [2, 4, 6]
   ```

11. **filter**: Creates a new array with all elements that pass the test implemented by the provided function.
   ```typescript
   let numbers: number[] = [1, 2, 3, 4, 5];
   let evenNumbers = numbers.filter(num => num % 2 === 0); // evenNumbers is [2, 4]
   ```

12. **reduce**: Applies a function against an accumulator and each element in the array (from left to right) to reduce it to a single value.
   ```typescript
   let numbers: number[] = [1, 2, 3, 4, 5];
   let sum = numbers.reduce((acc, curr) => acc + curr, 0); // sum is 15
   ```

13. **every**: Tests whether all elements in the array pass the test implemented by the provided function.
   ```typescript
   let numbers: number[] = [1, 2, 3, 4, 5];
   let allGreaterThanZero = numbers.every(num => num > 0); // true
   ```

14. **some**: Tests whether at least one element in the array passes the test implemented by the provided function.
   ```typescript
   let numbers: number[] = [1, 2, 3, 4, 5];
   let hasEvenNumber = numbers.some(num => num % 2 === 0); // true
   ```

15. **find**: Returns the value of the first element in the array that satisfies the provided testing function. Otherwise, it returns `undefined`.
   ```typescript
   let numbers: number[] = [1, 2, 3, 4, 5];
   let evenNumber = numbers.find(num => num % 2 === 0); // evenNumber is 2
   ```

16. **findIndex**: Returns the index of the first element in the array that satisfies the provided testing function. Otherwise, it returns `-1`.
   ```typescript
   let numbers: number[] = [1, 2, 3, 4, 5];
   let indexOfFirstEvenNumber = numbers.findIndex(num => num % 2 === 0); //
indexOfFirstEvenNumber is 1
   ```

17. **includes**: Determines whether an array includes a certain element, returning `true` or `false` as appropriate.
   ```typescript
   let numbers: number[] = [1, 2, 3, 4, 5];
   let includesThree = numbers.includes(3); // true
   ```

18. **reverse**: Reverses the elements of an array in place.
   ```typescript
   let numbers: number[] = [1, 2, 3, 4, 5];
   numbers.reverse(); // numbers is now [5, 4, 3, 2, 1]
   ```

19. **sort**: Sorts the elements of an array in place and returns the sorted array.
   ```typescript
   let numbers: number[] = [3, 1, 2, 5, 4];
   numbers.sort(); // numbers is now [1, 2, 3, 4, 5]
   ```

20. **join**: Joins all elements of an array into a string, optionally separated by a specified separator string.

```typescript
let fruits: string[] = ["apple", "banana", "orange"];
let fruitString = fruits.join(", "); // fruitString is "apple, banana, orange"
```

.