Task 1: Function vs Arrow Function

Problem Statement: Write two functions, one using traditional function syntax and the other using arrow function syntax. Both functions should take a list of numbers as input and return a new list containing the square of each number.

Example Input:

javascript
```
[1, 2, 3, 4, 5]
```

Example Output:

javascript
```
[1, 4, 9, 16, 25]
```

Task 2: Spread (...) Operator

Problem Statement: Create a function that takes three objects as arguments and merges them into one new object using the spread operator.

Example Input:

javascript
```
const obj1 = { a: 1, b: 2 };
const obj2 = { c: 3, d: 4 };
const obj3 = { e: 5, f: 6 };
```

Example Output:

javascript
```
{ a: 1, b: 2, c: 3, d: 4, e: 5, f: 6 }
```

Task 3: Map Objects

Problem Statement: Create a function that takes a list of students and their scores, stores them in a Map, and returns the score of a student given their name.

Example Input:

javascript
```
const students = [
  { name: 'Alice', score: 85 },
  { name: 'Bob', score: 92 },
  { name: 'Charlie', score: 78 }
];
'Bob'
```

Example Output:

javascript
```
92
```

Task 4: Set Objects

Problem Statement: Write a function that takes a list of numbers and returns a new list containing only the unique numbers, using a Set to remove duplicates.

Example Input:

javascript
```
[1, 2, 2, 3, 4, 4, 5]
```

Example Output:

javascript
```
[1, 2, 3, 4, 5]
```

Task 5: Array Filter

Problem Statement: Write a function that filters an array of words to include only those with more than 5 letters.

Example Input:

javascript

```
['apple', 'banana', 'cherry', 'date', 'elderberry', 'fig', 'grape']
```

Example Output:

javascript
```
['banana', 'cherry', 'elderberry']
```

Task 6: Array reduce

Problem Statement: Write a function that takes an array of numbers and returns the product of all numbers in the array using the reduce method.

Example Input:

javascript
```
[1, 2, 3, 4, 5]
```

Example Output:

javascript
```
120
```

Task 7: Array indexOf() and lastIndexOf()

Problem Statement: Write a function that finds the first and last occurrence of a specific element in an array and returns them as an object.

Example Input:

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

Example Output:

javascript
```
{ firstIndex: 1, lastIndex: 5 }
```

Task 8: Array.isArray()

Problem Statement: Write a function that takes a variable and returns whether it is an array or not.

Example Input:

javascript
```
[1, 2, 3]
'Hello'
```

Example Output:

javascript
```
true
false
```

Task 9: String includes

Problem Statement: Write a function that takes a sentence and a word, and returns whether the word is present in the sentence using the includes() method.

Example Input:

javascript
```
'The quick brown fox jumps over the lazy dog'
'fox'
```

Example Output:

javascript
```
true
```

Task 10: Array keys()

Problem Statement: Write a function that takes an array of items and prints each item's index using the keys() method.

Example Input:

javascript
```
['apple', 'banana', 'cherry']
```

Example Output:

javascript
```
0
1
2
```

Task 11: String replaceAll()

Problem Statement: Write a function that takes a string and replaces all occurrences of a given substring with a new substring using replaceAll().

Example Input:

javascript
```
'apple banana apple grape apple'
'apple'
'orange'
```

Example Output:

javascript
```
'orange banana orange grape orange'
```

Task 12: Array includes()

Problem Statement: Write a function that takes an array and a value, and returns whether the array includes that value using the includes() method.

Example Input:

javascript
```
['apple', 'banana', 'cherry']
'banana'
```

Example Output:

javascript
```
true
```

Task 13: Async, await, promise, fetch, axios

Problem Statement: Write an asynchronous function that fetches data from a public API and logs the response using async/await. Use either fetch or axios for the HTTP request.

Example Input:

javascript
```
// URL: 'https://jsonplaceholder.typicode.com/todos/1'
```

Example Output:

javascript
```
{
  userId: 1,
  id: 1,
  title: "delectus aut autem",
  completed: false
}
```

Task 14: Exception Handling

Problem Statement: Write a function that takes two numbers and divides the first by the second. Use try-catch to handle any potential division by zero errors.

Example Input:

javascript
```
4, 2
4, 0
```

Example Output:

javascript
```
2
'Division by zero is not allowed.'
```

Task 15: Template Literals

Problem Statement: Write a function that takes a person's name and age and returns a string formatted using template literals.

Example Input:

javascript
```
'John', 30
```

Example Output:

javascript
```
'Hello, my name is John and I am 30 years old.'
```

Task 16: Destructuring Assignment

Problem Statement: Write a function that takes an object with properties `name`, `age`, and `city`, and logs each property using destructuring assignment.

Example Input:

javascript
```
{ name: 'Alice', age: 25, city: 'New York' }
```

Example Output:

javascript
```
'Alice'
'25'
'New York'
```

Task 17: Default Parameters

Problem Statement: Write a function that takes two parameters and returns their sum. If the second parameter is not provided, it should default to 10.

Example Input:

javascript
```
5, 20
5
```

Example Output:

javascript
```
25
15
```

Task 18: Rest Parameters

Problem Statement: Write a function that takes any number of arguments and returns their sum using rest parameters.

Example Input:

javascript
```
1, 2, 3, 4, 5
```

Example Output:

javascript
```
15
```

Task 19: Sum of Numbers

Problem Statement: Write a function that takes an array of numbers and returns the sum of all the numbers using the `reduce` method.

Example Input:

javascript
```
[1, 2, 3, 4, 5]
```

Example Output:

javascript
```
15
```

Task 20: Product of Numbers

Problem Statement: Write a function that takes an array of numbers and returns the product of all the numbers using the `reduce` method.

Example Input:

javascript
```
[1, 2, 3, 4, 5]
```

Example Output:

javascript
```
120
```

Task 21: Longest String

Problem Statement: Write a function that takes an array of strings and returns the longest string using the `reduce` method.

Example Input:

javascript
```
['apple', 'banana', 'cherry', 'date']
```

Example Output:

javascript
```
'banana'
```

Task 22: Flatten Array

Problem Statement: Write a function that takes a nested array of arrays and returns a single flattened array using the `reduce` method.

Example Input:

javascript
```
[[1, 2, 3], [4, 5], [6, 7, 8, 9]]
```

Example Output:

javascript
```
[1, 2, 3, 4, 5, 6, 7, 8, 9]
```

Task 23: Count Occurrences

Problem Statement: Write a function that takes an array of words and returns an object that counts the occurrences of each word using the `reduce` method.

Example Input:

javascript
```
['apple', 'banana', 'apple', 'orange', 'banana', 'apple']
```

Example Output:

javascript
```
{ apple: 3, banana: 2, orange: 1 }
```

Task 24: Group by Property

Problem Statement: Write a function that takes an array of objects and groups them by a specified property using the `reduce` method.

Example Input:

javascript
```
const people = [
  { name: 'Alice', age: 21 },
  { name: 'Bob', age: 25 },
  { name: 'Charlie', age: 21 },
  { name: 'David', age: 25 },
  { name: 'Eve', age: 22 }
];
groupBy(people, 'age');
```

Example Output:

javascript
```
{
  21: [{ name: 'Alice', age: 21 }, { name: 'Charlie', age: 21 }],
  25: [{ name: 'Bob', age: 25 }, { name: 'David', age: 25 }],
  22: [{ name: 'Eve', age: 22 }]
}
```

Task 25: Calculate Average

Problem Statement: Write a function that takes an array of numbers and returns the average of those numbers using the `reduce` method.

Example Input:

javascript
```
[10, 20, 30, 40, 50]
```

Example Output:

javascript
```
30
```

Task 26: Total Price of Items in Cart

Problem Statement: Write a function that takes an array of objects representing items in a shopping cart and returns the total price using the `reduce` method. Each object contains a `price` property.

Example Input:

javascript
```
const cart = [
  { item: 'apple', price: 1.5 },
  { item: 'banana', price: 2.0 },
  { item: 'orange', price: 1.25 }
];
calculateTotal(cart);
```

Example Output:

javascript
```
4.75
```

Task 27: Find First Even Number

Problem Statement: Write a function that takes an array of numbers and returns the first even number in the array using the `find` method.

Example Input:

javascript
```
[1, 3, 7, 10, 2]
```

Example Output:

javascript
```
10
```

Task 28: Find Student by Name

Problem Statement: Write a function that takes an array of student objects and a name, and returns the student object that matches the given name using the `find` method. Each student object has properties `name` and `age`.

Example Input:

javascript
```
const students = [
  { name: 'Alice', age: 21 },
  { name: 'Bob', age: 25 },
  { name: 'Charlie', age: 23 }
];
findStudentByName(students, 'Bob');
```

Example Output:

javascript
```
{ name: 'Bob', age: 25 }
```

Task 29: Find Product by ID

Problem Statement: Write a function that takes an array of product objects and a product ID, and returns the product object that matches the given ID using the `find` method. Each product object has properties `id` and `name`.

Example Input:

javascript
```
const products = [
  { id: 101, name: 'Laptop' },
  { id: 102, name: 'Phone' },
  { id: 103, name: 'Tablet' }
];
findProductById(products, 102);
```

Example Output:

javascript
```
{ id: 102, name: 'Phone' }
```

Task 30: Find Overdue Task

Problem Statement: Write a function that takes an array of task objects and returns the first task that is overdue. Each task object has properties `taskName` and `dueDate`. Assume the due date is in the format 'YYYY-MM-DD'.

Example Input:

javascript
```
const tasks = [
  { taskName: 'Task 1', dueDate: '2023-06-01' },
  { taskName: 'Task 2', dueDate: '2024-05-01' },
  { taskName: 'Task 3', dueDate: '2024-01-01' }
];
findOverdueTask(tasks);
```

Example Output:

javascript
```
{ taskName: 'Task 1', dueDate: '2023-06-01' }
```

Task 31: Find First Positive Number

Problem Statement: Write a function that takes an array of numbers and returns the first positive number in the array using the `find` method.

Example Input:

javascript
```
[-5, -3, 0, 9, 2]
```

Example Output:

javascript
```
9
```

Task 32: Find Book by Title

Problem Statement: Write a function that takes an array of book objects and a title, and returns the book object that matches the given title using the `find` method. Each book object has properties `title` and `author`.

Example Input:

javascript
```
const books = [
  { title: '1984', author: 'George Orwell' },
  { title: 'To Kill a Mockingbird', author: 'Harper Lee' },
  { title: 'The Great Gatsby', author: 'F. Scott Fitzgerald' }
];
findBookByTitle(books, '1984');
```

Example Output:

javascript
```
{ title: '1984', author: 'George Orwell' }
```

Task 33: Find Employee by ID

Problem Statement: Write a function that takes an array of employee objects and an employee ID, and returns the employee object that matches the given ID using the `find` method. Each employee object has properties `id`, `name`, and `position`.

Example Input:

javascript
```
const employees = [
  { id: 1, name: 'Alice', position: 'Manager' },
  { id: 2, name: 'Bob', position: 'Engineer' },
  { id: 3, name: 'Charlie', position: 'Technician' }
];
findEmployeeById(employees, 2);
```

Example Output:

javascript
```
{ id: 2, name: 'Bob', position: 'Engineer' }
```

Task 34: Find First Prime Number

Problem Statement: Write a function that takes an array of numbers and returns the first prime number in the array using the `find` method.

Example Input:

javascript
```
[4, 6, 8, 9, 11, 15]
```

Example Output:

javascript
```
11
```

Task 35: Destructuring and Template Literals

Problem Statement: Write a function that takes an object representing a person with properties firstName, lastName, and age. Use destructuring to extract these properties and return a string formatted using template literals.

Example Input:

javascript
```
{ firstName: 'John', lastName: 'Doe', age: 30 }
```

Example Output:

javascript
'John Doe is 30 years old.'

Task 36: Array Methods and Arrow Functions

Problem Statement: Write a function that takes an array of numbers and returns a new array containing the squares of only the even numbers. Use the filter, map, and arrow functions.

Example Input:

javascript
[1, 2, 3, 4, 5, 6]

Example Output:

javascript
[4, 16, 36]

Task 37: Default Parameters and Rest Parameters

Problem Statement: Write a function that takes a string separator and any number of words. The separator should default to a comma if not provided. The function should return a single string with all the words joined by the separator.

Example Input:

javascript
'-', 'apple', 'banana', 'cherry'

Example Output:

javascript
'apple-banana-cherry'

Example Input:

javascript
'apple', 'banana', 'cherry'

Example Output:

```
javascript
'apple,banana,cherry'
```

Task 38: Spread Operator and Array Methods

Problem Statement: Write a function that takes two arrays and returns a new array containing only the unique elements from both arrays combined. Use the spread operator and the Set object.

Example Input:

```
javascript
[1, 2, 3], [3, 4, 5]
```

Example Output:

```
javascript
[1, 2, 3, 4, 5]
```

Task 39: Promises and Async/Await

Problem Statement: Write an asynchronous function that fetches data from a public API (e.g., https://jsonplaceholder.typicode.com/todos/1) using fetch. Use async/await to handle the asynchronous operation and return the result.

Example Input:

```
javascript
fetchTodo();
```

Example Output:

```
javascript
{
  userId: 1,
  id: 1,
  title: "delectus aut autem",
  completed: false
}
```

Task 40: Array Methods, Destructuring, and Reduce

Problem Statement: Write a function that takes an array of objects representing products with properties name and price. Calculate the total price of all products using reduce, and return an array of product names whose prices are greater than the average price. Use destructuring within the reduce function.

Example Input:

javascript
```
[
  { name: 'Laptop', price: 1000 },
  { name: 'Phone', price: 500 },
  { name: 'Tablet', price: 700 }
]
```

Example Output:

javascript
```
['Laptop']
```