

---

## ✓ 1. Communicating with End Users from JavaScript

◆ Example:

```
<script>
  alert("Welcome to our website!");
  const userName = prompt("What's your name?");
  confirm("Are you sure you entered your name correctly?");
  console.log("User entered:", userName);
</script>
```

◆ Exercises:

1. Show a greeting message using `alert`.
  2. Ask for the user's age and log it using `prompt`.
  3. Use `confirm` to ask if the user wants to continue. Display the result in the console.
- 

## ✓ 2. Separating HTML and JavaScript Sources

◆ Example:

index.html

```
<!DOCTYPE html>
<html>
<head>
  <title>External JS Example</title>
  <script src="script.js" defer></script>
</head>
<body>
  <h1>Hello!</h1>
</body>
</html>
```

script.js

```
alert("This alert is from an external JavaScript file!");
```

◆ Exercises:

1. Create a new HTML file and link it to an external JavaScript file.
2. Write a message in the external JS file that logs a message to the console.
3. Modify the script to display an alert when the page loads.

---

### ✓ 3. Accessing the DOM from JavaScript

◆ Example:

```
<body>
  <p id="demo">Hello!</p>
  <button onclick="changeText()">Click Me</button>

  <script>
    function changeText() {
      document.getElementById("demo").textContent = "You clicked the button!";
    }
  </script>
</body>
```

◆ Exercises:

1. Access an element by ID and change its text.
  2. Access an element by class name and apply a style change.
  3. Create a button that changes the background color of the page.
- 

### ✓ 4. Variable Declarations: var, let and const

◆ Example:

```
var x = 10;
let y = 20;
const z = 30;

x = 15;
y = 25;
// z = 35; // This will throw an error

console.log(x, y, z);
```

◆ Exercises:

1. Declare variables using var, let, and const. Try reassigning them.
  2. Create a block-scoped variable using let and check if it's accessible outside the block.
  3. Use const to declare an object. Try modifying a property of that object.
-

## ✓ 5. Empty Values in JavaScript: undefined and null

◆ Example:

```
let a;  
console.log(a); // undefined  
  
let b = null;  
console.log(b); // null
```

◆ Exercises:

1. Declare a variable without assigning a value. Log its value.
  2. Assign `null` to a variable. Compare it to `undefined` using `==` and `===`.
  3. Create a function that returns `undefined` if no argument is passed.
- 

## ✓ 6. User Interactions Using `alert`, `prompt`, and `confirm`

◆ Example:

```
alert("Welcome!");  
  
let name = prompt("What is your name?");  
let sure = confirm("Are you sure your name is " + name + "?");  
  
console.log("Name entered:", name);  
console.log("Confirmation:", sure);
```

◆ Exercises:

1. Use `prompt` to get the user's favorite color and display it in an alert.
  2. Ask the user if they want to subscribe using `confirm` and log the result.
  3. Combine `prompt` and `confirm` to create a simple sign-up interaction.
- 
- 

## ✓ 7. Numbers in JavaScript

◆ Example:

```
let x = 5;  
let y = 3.14;  
let total = x + y;  
console.log("Total:", total);
```

◆ Exercises:

1. Declare two number variables and log their sum, difference, and product.
  2. Use `typeof` to verify the type of a number.
  3. Convert a string like "42" to a number and add 10 to it.
- 

## ✓ 8. Initializing and Manipulating Strings in JavaScript

◆ Example:

```
let greeting = "Hello";
let name = "Alice";
let message = greeting + ", " + name + "!";
console.log(message);
```

◆ Exercises:

1. Create two string variables and concatenate them.
  2. Use template literals to display a full sentence.
  3. Find the length of a string and log it.
- 

## ✓ 9. Analysing and Modifying Strings in JavaScript

◆ Example:

```
let sentence = "JavaScript is fun!";
console.log(sentence.toUpperCase());
console.log(sentence.indexOf("fun"));
console.log(sentence.replace("fun", "awesome"));
```

◆ Exercises:

1. Convert a string to lowercase.
  2. Extract the first 5 characters from a string.
  3. Replace a word in a sentence with another word.
- 

## ✓ 10. Dates in JavaScript

◆ Example:

```
let now = new Date();
console.log("Current Date:", now);
console.log("Year:", now.getFullYear());
```

◆ Exercises:

1. Display the current date and time.
  2. Get and display only the current year and month.
  3. Create a specific date object and format it as DD/MM/YYYY.
- 

## ✓ 11. Using the Math Library for Common Math Operations

◆ Example:

```
let num = -8.6;
console.log(Math.abs(num));
console.log(Math.ceil(num));
console.log(Math.floor(num));
console.log(Math.round(num));
```

◆ Exercises:

1. Generate a random number between 0 and 100.
  2. Use `Math.pow()` to calculate  $3^4$ .
  3. Use `Math.sqrt()` to find the square root of a number.
- 

## ✓ 12. Arithmetic Operators

◆ Example:

```
let a = 10;
let b = 3;
console.log(a + b, a - b, a * b, a / b, a % b);
```

◆ Exercises:

1. Write a function that takes two numbers and returns their average.
  2. Calculate the area of a rectangle using width and height.
  3. Use modulus to check if a number is even or odd.
- 

## ✓ 13. Logical and Conditional Operators

◆ Example:

```
let age = 18;
if (age >= 18 && age <= 65) {
```

```

    console.log("You are an adult.");
  } else {
    console.log("Age not in adult range.");
  }
}

```

◆ Exercises:

1. Use `&&` and `||` to create a login simulation with username/password.
  2. Check if a number is in a specific range using logical operators.
  3. Use `!` to reverse a Boolean value.
- 

## ✓ 14. Type Casting

◆ Example:

```

let strNum = "100";
let num = Number(strNum);
console.log(num + 50); // 150

```

◆ Exercises:

1. Convert a string to a number using `Number()` and `parseInt()`.
  2. Convert a number to a string and concatenate with another string.
  3. Check what happens if you try to cast "abc" to a number.
- 

## ✓ 15. Looping Control Structures

◆ Example:

```

for (let i = 1; i <= 5; i++) {
  console.log("Iteration:", i);
}

```

◆ Exercises:

1. Print numbers 1 to 10 using a `for` loop.
  2. Use a `while` loop to count down from 10 to 1.
  3. Loop through an array of numbers and print only even numbers.
- 
-

## ✓ 16. An Introduction to Functions in JavaScript

◆ Example:

```
function greet(name) {  
  return `Hello, ${name}!`;  
}  
console.log(greet("Alice"));
```

◆ Exercises:

1. Create a function that returns the square of a number.
  2. Write a function that takes two numbers and returns the greater one.
  3. Create a function that prints "Welcome!" to the console.
- 

## ✓ 17. Global and Local Variables

◆ Example:

```
let globalVar = "I'm global";  
  
function testScope() {  
  let localVar = "I'm local";  
  console.log(globalVar);  
  console.log(localVar);  
}  
testScope();  
// console.log(localVar); // Will throw an error
```

◆ Exercises:

1. Create a function that accesses a global variable.
  2. Try logging a local variable outside its function (note the error).
  3. Modify a global variable inside a function.
- 

## ✓ 18. Working with Functions

◆ Example:

```
const add = (a, b) => a + b;  
console.log(add(5, 10));
```

◆ Exercises:

1. Create an arrow function that multiplies two numbers.
  2. Define a function expression that returns the length of a string.
  3. Use a function as an argument to another function.
- 

## ✓ 19. The Fundamentals of Error Handling

◆ Example:

```
try {  
  let x = y + 1; // y is not defined  
} catch (error) {  
  console.error("An error occurred:", error.message);  
}
```

◆ Exercises:

1. Trigger and catch a reference error.
  2. Use `try...catch` to validate a number input.
  3. Throw a custom error if a function receives no argument.
- 

## ✓ 20. Creating Arrays

◆ Example:

```
let fruits = ["apple", "banana", "mango"];  
console.log(fruits);
```

◆ Exercises:

1. Create an array of 5 numbers.
  2. Access the first and last element of an array.
  3. Add a new element using `push()`.
- 

## ✓ 21. Copying Arrays

◆ Example:

```
let original = [1, 2, 3];  
let copy = [...original];  
console.log(copy);
```



◆ Exercises:

1. Copy an array using the spread operator.
  2. Use `slice()` to copy part of an array.
  3. Show the difference between reference copy and shallow copy.
- 

## ✓ 22. Splicing and Slicing Arrays

◆ Example:

```
let arr = ["a", "b", "c", "d"];
let removed = arr.splice(1, 2); // Removes "b" and "c"
console.log(arr);
console.log(removed);
```

◆ Exercises:

1. Use `splice()` to remove 2 elements from an array.
  2. Use `slice()` to extract a subarray.
  3. Add new elements using `splice()`.
- 

## ✓ 23. Concatenating and Sorting Arrays

◆ Example:

```
let a = [3, 1, 4];
let b = [2, 5];
let combined = a.concat(b).sort();
console.log(combined);
```

◆ Exercises:

1. Concatenate two arrays of strings.
  2. Sort an array of numbers (hint: use `sort((a, b) => a - b)`).
  3. Reverse the order of an array.
- 

## ✓ 24. An Introduction to JavaScript Objects

◆ Example:

```
let car = {
  brand: "Toyota",
```

```
    model: "Camry",  
    year: 2020  
};  
console.log(car.model);
```

◆ Exercises:

1. Create an object with at least 3 properties.
  2. Access and modify one of the properties.
  3. Add a new property to the object.
- 

## ✓ 25. Removing Properties from Objects

◆ Example:

```
delete car.year;  
console.log(car);
```

◆ Exercises:

1. Create an object and remove a property using `delete`.
  2. Check if a property exists before and after deleting.
  3. Try deleting a non-existent property.
- 

## ✓ 26. The "this" Keyword in JavaScript Objects

◆ Example:

```
let person = {  
  name: "Alice",  
  greet() {  
    return `Hi, I'm ${this.name}`;  
  }  
};  
console.log(person.greet());
```

◆ Exercises:

1. Create an object with a method using `this`.
  2. Log the value of `this` inside a method.
  3. Compare `this` in regular vs arrow functions.
-

## ✓ 27. Linking Functions to Objects

◆ Example:

```
let dog = {  
  name: "Buddy",  
  bark: function() {  
    console.log(`${this.name} says woof!`);  
  }  
};  
dog.bark();
```

◆ Exercises:

1. Attach a method to an object that logs a message.
  2. Create a function separately and assign it to an object.
  3. Modify the function to use `this` keyword.
- 

## ✓ 28. Object Constructors

◆ Example:

```
function Person(name, age) {  
  this.name = name;  
  this.age = age;  
}  
const john = new Person("John", 25);  
console.log(john);
```

◆ Exercises:

1. Create a constructor function for Car.
  2. Add a method inside the constructor.
  3. Instantiate multiple objects using `new`.
- 

## ✓ 29. Creating New Objects from Existing Ones

◆ Example:

```
let user = { name: "Alice" };  
let admin = Object.create(user);  
admin.role = "admin";  
console.log(admin.name); // Inherited
```

◆ Exercises:

1. Use `Object.create()` to inherit properties.
  2. Override a property in the new object.
  3. Add a new property to the child object.
- 

## ✓ 30. Object Methods

◆ Example:

```
let calculator = {  
  add(a, b) {  
    return a + b;  
  }  
};  
console.log(calculator.add(5, 3));
```

◆ Exercises:

1. Add a subtract method to an object.
  2. Create a method that logs an object's info.
  3. Use `this` inside a method to reference a property.
- 

## ✓ 31. Freezing Objects

◆ Example:

```
let config = { api: "v1" };  
Object.freeze(config);  
config.api = "v2"; // Won't change  
console.log(config.api);
```

◆ Exercises:

1. Freeze an object and attempt to change it.
  2. Try adding a property after freezing.
  3. Check if an object is frozen with `Object.isFrozen()`.
-

## ✓ 32. The `map` Method for JavaScript Arrays

◆ Example:

```
let nums = [1, 2, 3];
let doubled = nums.map(n => n * 2);
console.log(doubled);
```

◆ Exercises:

1. Use `map` to convert strings to uppercase.
  2. Add 5 to every number in an array using `map`.
  3. Convert an array of numbers to their squares.
- 

## ✓ 33. The `reduce` and `filter` Methods for JavaScript Arrays

◆ Example:

```
let numbers = [10, 5, 8, 3];

let total = numbers.reduce((sum, n) => sum + n, 0);
let filtered = numbers.filter(n => n > 5);

console.log(total); // 26
console.log(filtered); // [10, 8]
```

◆ Exercises:

1. Use `reduce` to find the product of all numbers.
  2. Use `filter` to get even numbers from an array.
  3. Combine `filter` and `map` to get squares of only positive numbers.
- 

## ✓ 34. The `instanceof` Operator

◆ Example:

```
function Animal() {}
let dog = new Animal();
console.log(dog instanceof Animal); // true
```

◆ Exercises:

1. Create a constructor and check if an object is an instance of it.
2. Check `instanceof` with built-in types like `Array` and `Date`.

3. Create a class and test an object with `instanceof`.

---