JS

JavaScript Tutorials

JavaScript Course Content

- → JavaScript Introduction
- → JavaScript Fundamentals
  1. Identifiers
  2. Reserved words
  3. Data Types
  4. Literals
  5. Arrays
  6. Types of Variables
  7. Functions
- → Conditional statements
  1. if
  2. if...else
  3. if...else if...else
  4. Switch
- → Loops

  6. while

  7. do...while

  8. for

  9. for...in Statement

  10. Break

  11.Continue
- → JS HTML DOM
  1. DOM Intro
  2. DOM Methods
  3. DOM Document
  4. DOM Elements
  5. DOM HTML
  6. DOM CSS
  7. DOM Animations
  8. DOM Events
  9. DOM Event Listener
  10. DOM Navigation
  11. DOM Nodes
  12. DOM Collections
  13. DOM Node Lists
- → JS Browser BOM
  1. JS Window
  2. JS Screen
  3. JS Location
  4. JS History
  5. JS Navigator
  6. JS Popup Alert
  7. JS Timing
  8. JS Cookies
- → JS AJAX
  - AJAX Intro
  - AJAX XMLHttp
  - AJAX Request
  - AJAX Response
  - AJAX XML File
  - AJAX PHP
  - AJAX ASP
  - AJAX Database
  - AJAX Examples
- → JS JSON
  - USON VS XML
  - JSON Data Types
  - JSON Parse
  - JSON Stringify
  - JSON Objects
  - JSON Arrays
  - JSON PHP
  - JSON HTML
  - JSON JSONP
- → JS Web APIs
  - Web API Intro
  - Web History API
  - Web Storage API
  - Web Geolocation API
- → JS vs jQuery
  - jQuery Selectors
  - jQuery HTML
  - jQuery CSS
  - jQuery DOM
- → JavaScript Interview Questions

JanBask TRAINING
Learn and Grow

JS Environment Setup

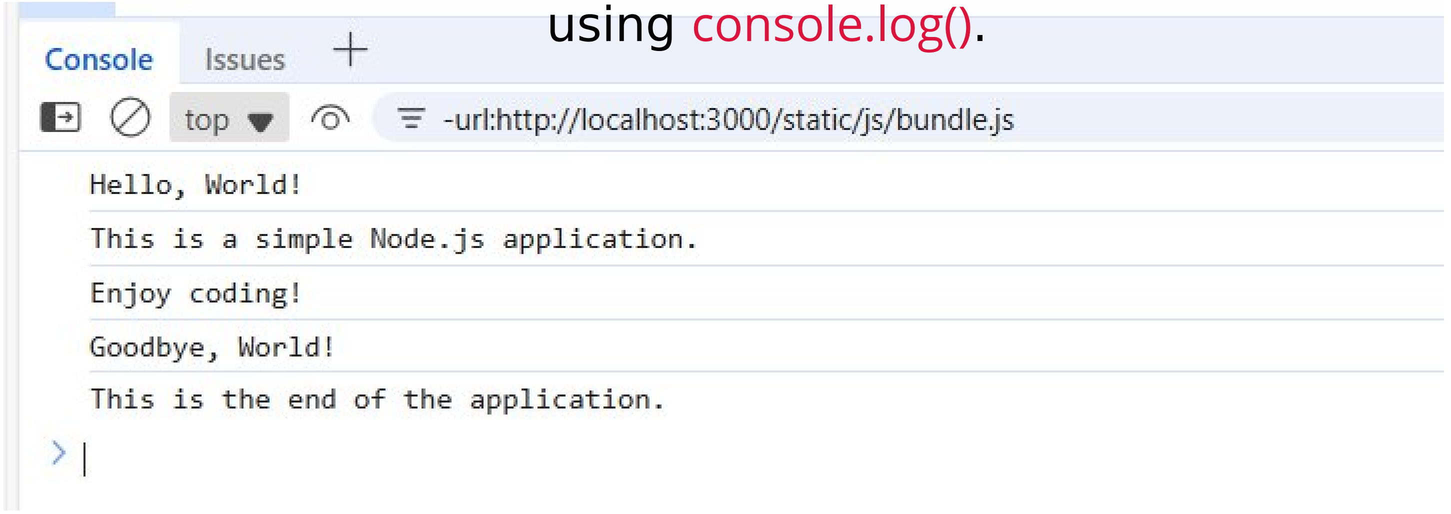
- Open Vs code
- Create Folder and give name javascript
- Create index.html and index.js
- Print statements

```
console.log("Hello, World!");
console.log("This is a simple Node.js application.");
console.log("Enjoy coding!");
console.log("Goodbye, World!");
console.log("This is the end of the application.");
```

- Install live server
- See the output from browser

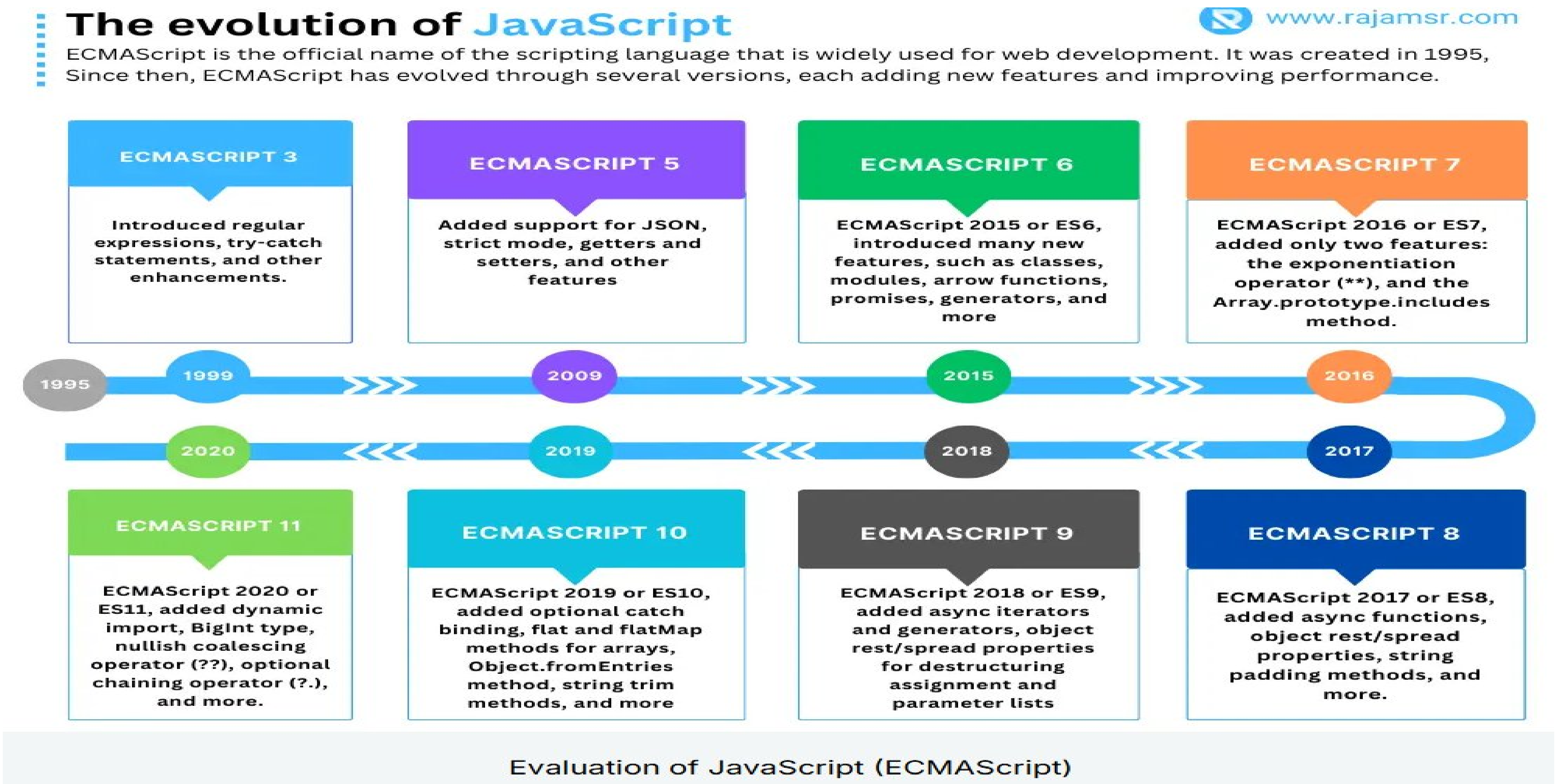
JavaScript can "display" data in different ways:

- Writing into an HTML element, using innerHTML or innerText.
- Writing into the HTML output using document.write().
- Writing into an alert box, using window.alert().
- Writing into the browser console, using console.log().



```
Hello, World!
This is a simple Node.js application.
Enjoy coding!
Goodbye, World!
This is the end of the application.
```

JavaScript ES6



The evolution of JavaScript

www.rajamsr.com

ECMAScript is the official name of the scripting language that is widely used for web development. It was created in 1995, Since then, ECMAScript has evolved through several versions, each adding new features and improving performance.

ECMASCRIPT 3

Introduced regular expressions, try-catch statements, and other enhancements.

ECMASCRIPT 5

Added support for JSON, strict mode, getters and setters, and other features

ECMASCRIPT 6

ECMAScript 2015 or ES6, introduced many new features, such as classes, modules, arrow functions, promises, generators, and more

ECMASCRIPT 7

ECMAScript 2016 or ES7, added only two features: the exponentiation operator (**), and the Array.prototype.includes method.

ECMASCRIPT 8

ECMAScript 2017 or ES8, added async functions, object rest/spread properties, string padding methods, and more.

ECMASCRIPT 9

ECMAScript 2018 or ES9, added async iterators and generators, object rest/spread properties for destructuring assignment and parameter lists

ECMASCRIPT 10

ECMAScript 2019 or ES10, added optional catch binding, flat and flatMap methods for arrays, Object.fromEntries method, string trim methods, and more

ECMASCRIPT 11

ECMAScript 2020 or ES11, added dynamic import, BigInt type, nullish coalescing operator (??), optional chaining operator (?.), and more.

Evaluation of JavaScript (ECMAScript)

JavaScript Variable and Data Type

What is JavaScript?

JavaScript is the programming language of the web.

Variables are Containers for Data

JavaScript variables can be **declared** in 4 ways:

Modern JavaScript

- Using let
- Using const

Older JavaScript

- Using var
- Automatically (Not Recommended)

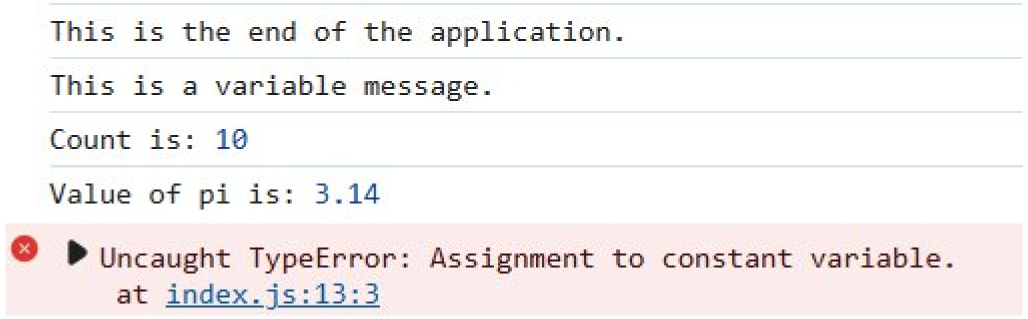
The general rules for constructing names for variables (unique identifiers) are:

- Names can contain letters, digits, underscores, and dollar signs.
- Names must begin with a letter.
- Names can also begin with $ and _ (but we will not use it in this tutorial).
- Names are case sensitive (y and Y are different variables).
- Reserved words (like JavaScript keywords) cannot be used as names.

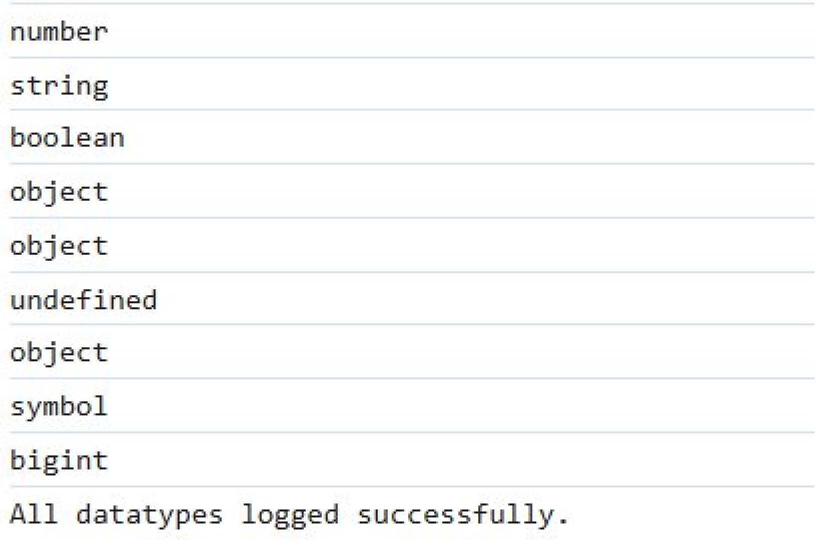
JavaScript Variable and Data Type

```
//variable name
var message = "This is a variable message.";
console.log(message);
let count = 10;
console.log("Count is:", count);
const pi = 3.14;
console.log("Value of pi is:", pi);
pi=3.14159; // This will cause an error
console.log("Updated value of pi is:", pi);
console.log("This line will not be executed due to the
error above.");
```

```
//datatype
let num = 42; // Number
let str = "Hello, World!"; // String
let bool = true; // Boolean
let arr = [1, 2, 3]; // Array
let obj = { name: "Alice", age: 30 }; // Object
let und; // Undefined
let nul = null; // Null
let sym = Symbol("sym"); // Symbol
let bigInt = 9007199254740991n; // BigInt
console.log(typeof num); // "number"
console.log(typeof str); // "string"
console.log(typeof bool); // "boolean"
console.log(typeof arr); // "object"
console.log(typeof obj); // "object"
console.log(typeof und); // "undefined"
console.log(typeof nul); // "object"
console.log(typeof sym); // "symbol"
console.log(typeof bigInt); // "bigint"
console.log("All datatypes logged successfully.");
```



```
This is the end of the application.
This is a variable message.
Count is: 10
Value of pi is: 3.14
✖ ▶Uncaught TypeError: Assignment to constant variable.
    at index.js:13:3
```

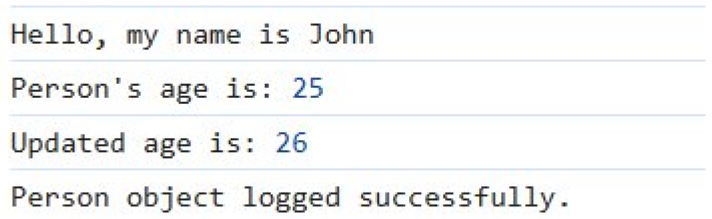


```
number
string
boolean
object
object
undefined
object
symbol
bigint
All datatypes logged successfully.
```

JavaScript Objects

An **Object** is a variable that can hold many variables.
Objects are collections of **key-value pairs**, where each key (known as **property names**) has a value.
Objects can describe anything like houses, cars, people, animals, or any other subjects.

```javascript
//object
let person = {
    name: "John",
    age: 25,
    greet: function() {
        console.log("Hello, my name is " + this.name);
    }
};
person.greet();
console.log("Person's age is:", person.age);
person.age = 26;
console.log("Updated age is:", person.age);
console.log("Person object logged successfully.");
```



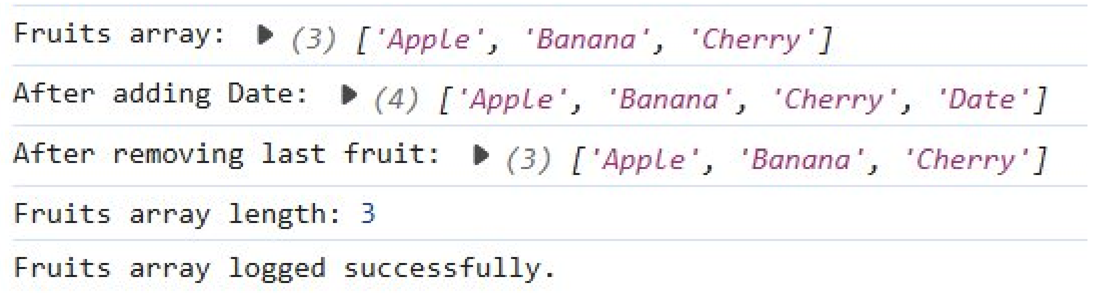
```
Hello, my name is John
Person's age is: 25
Updated age is: 26
Person object logged successfully.
```

JavaScript Arrays

An Array is an object type designed for storing data collections.
Key characteristics of JavaScript arrays are:

- **Elements**: An array is a list of values, known as elements.
- **Ordered**: Array elements are ordered based on their index.
- **Zero indexed**: The first element is at index 0, the second at index 1, and so on.
- **Dynamic size**: Arrays can grow or shrink as elements are added or removed.
- **Heterogeneous**: Arrays can store elements of different data types (numbers, strings, objects and other arrays).

```
//array
let fruits = ["Apple", "Banana", "Cherry"];
console.log("Fruits array:", fruits);
fruits.push("Date");
console.log("After adding Date:", fruits);
fruits.pop();
console.log("After removing last fruit:", fruits);
console.log("Fruits array length:", fruits.length);
console.log("Fruits array logged successfully.");
```



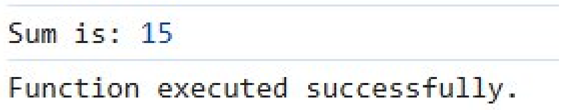
```
Fruits array:  ▶ (3) ['Apple', 'Banana', 'Cherry']
After adding Date:  ▶ (4) ['Apple', 'Banana', 'Cherry', 'Date']
After removing last fruit:  ▶ (3) ['Apple', 'Banana', 'Cherry']
Fruits array length: 3
Fruits array logged successfully.
```

JavaScript Functions

What are Functions?

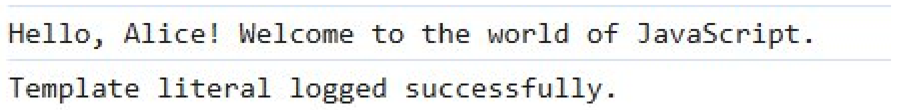
Functions are **fundamental building blocks** in all programming.
Functions enable **better code organization**, modularity, and efficiency.
Functions are **reusable block of code** designed to perform a particular task.
Functions **execute** when they are "called" or "invoked".

```javascript
//function
function add(a, b) {
    return a + b;
}
let sum = add(5, 10);
console.log("Sum is:", sum);
console.log("Function executed successfully.");
```



```
Sum is: 15
Function executed successfully.
```

```javascript
//template literals
let name = "Alice";
let greeting = `Hello, ${name}! Welcome to the world of
JavaScript.`;
console.log(greeting);
console.log("Template literal logged successfully.");
```



```
Hello, Alice! Welcome to the world of JavaScript.
Template literal logged successfully.
```

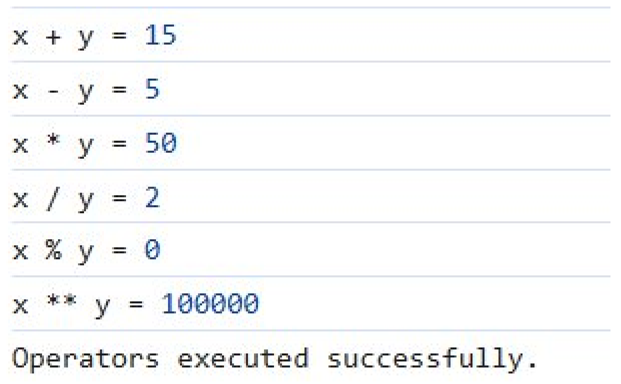
JavaScript Operator

Types of JavaScript Operators

There are different types of JavaScript operators:
- Arithmetic Operators
- Assignment Operators
- Comparison Operators
- Logical Operators
- And more ...

```
//operators
let x = 10;
let y = 5;
console.log("x + y =", x + y);
console.log("x - y =", x - y);
console.log("x * y =", x * y);
console.log("x / y =", x / y);
console.log("x % y =", x % y);
console.log("x ** y =", x ** y);
console.log("Operators executed successfully.");
```

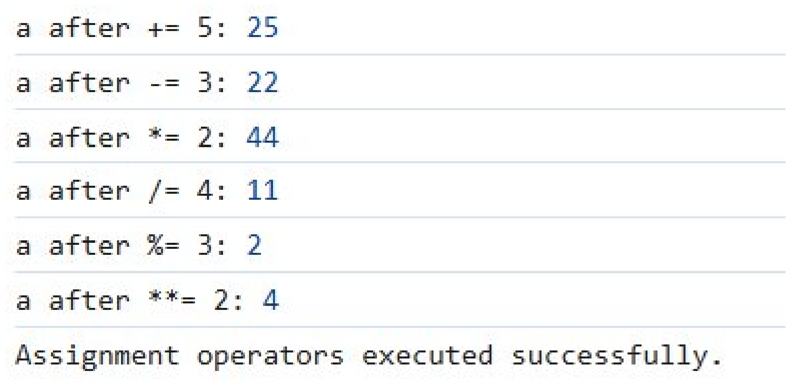
Operator	Description
+	Addition
-	Subtraction
*	Multiplication
**	Exponentiation
/	Division
%	Modulus (Division Remainder)
++	Increment
--	Decrement



```
x + y = 15
x - y = 5
x * y = 50
x / y = 2
x % y = 0
x ** y = 100000
Operators executed successfully.
```

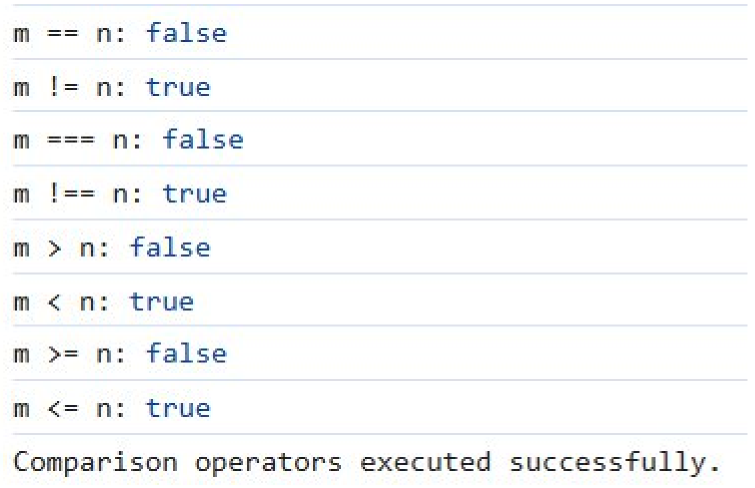
JavaScript Operator

```
//assignment operators
let a = 20;
a += 5; // a = a + 5
console.log("a after += 5:", a);
a -= 3; // a = a - 3
console.log("a after -= 3:", a);
a *= 2; // a = a * 2
console.log("a after *= 2:", a);
a /= 4; // a = a / 4
console.log("a after /= 4:", a);
a %= 3; // a = a % 3
console.log("a after %= 3:", a);
a **= 2; // a = a ** 2
console.log("a after **= 2:", a);
console.log("Assignment operators executed
successfully.");
```



```
a after += 5: 25
a after -= 3: 22
a after *= 2: 44
a after /= 4: 11
a after %= 3: 2
a after **= 2: 4
Assignment operators executed successfully.
```

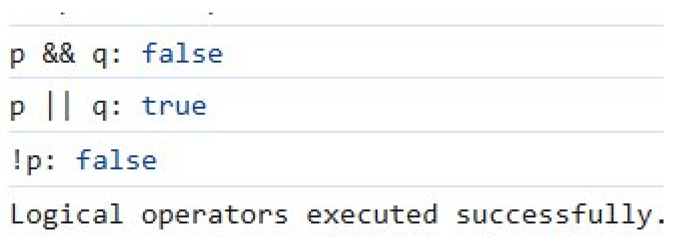
```
//comparison operators
let m = 15;
let n = 20;
console.log("m == n:", m == n);
console.log("m != n:", m != n);
console.log("m === n:", m === n);
console.log("m !== n:", m !== n);
console.log("m > n:", m > n);
console.log("m < n:", m < n);
console.log("m >= n:", m >= n);
console.log("m <= n:", m <= n);
console.log("Comparison operators executed
successfully.");
```



```
m == n: false
m != n: true
m === n: false
m !== n: true
m > n: false
m < n: true
m >= n: false
m <= n: true
Comparison operators executed successfully.
```

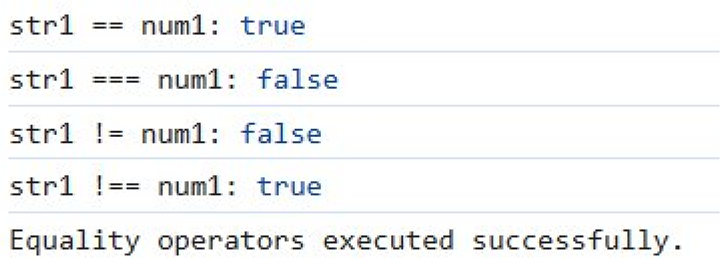
JavaScript Operator

```
//logical operators
let p = true;
let q = false;
console.log("p && q:", p && q);
console.log("p || q:", p || q);
console.log("!p:", !p);
console.log("Logical operators executed successfully.");
```



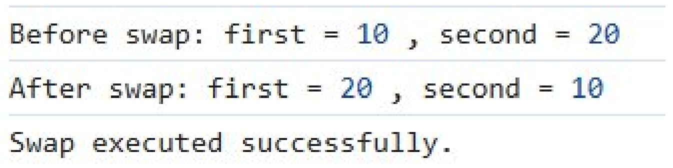
```
p && q: false
p || q: true
!p: false
Logical operators executed successfully.
```

```
//equality operators
let str1 = "5";
let num1 = 5;
console.log("str1 == num1:", str1 == num1);
console.log("str1 === num1:", str1 === num1);
console.log("str1 != num1:", str1 != num1);
console.log("str1 !== num1:", str1 !== num1);
console.log("Equality operators executed
successfully.");
```



```
str1 == num1: true
str1 === num1: false
str1 != num1: false
str1 !== num1: true
Equality operators executed successfully.
```

```
//exercise swap two numbers
let first = 10;
let second = 20;
console.log("Before swap: first =", first, ", second =",
second);
let temp = first;
first = second;
second = temp;
console.log("After swap: first =", first, ", second =",
second);
console.log("Swap executed successfully.");
```



```
Before swap: first = 10 , second = 20
After swap: first = 20 , second = 10
Swap executed successfully.
```

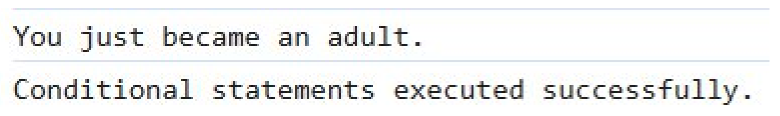
JavaScript Conditional Statements

In JavaScript we have the following conditional statements:

- Use if to specify a block of code to be executed, if a specified condition is true
- Use else to specify a block of code to be executed, if the same condition is false
- Use else if to specify a new condition to test, if the first condition is false
- Use switch to specify many alternative blocks of code to be executed

```
if (condition1) {
  //  block of code to be executed if condition1 is true
} else if (condition2) {
  //  block of code to be executed if the condition1 is false and condition2 is true
} else {
  //  block of code to be executed if the condition1 is false and condition2 is false
}
```

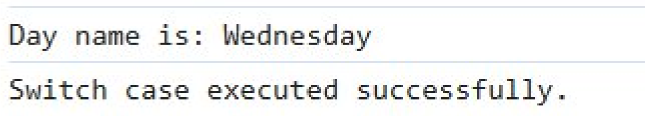
```
//conditional statements
let age = 18;
if (age < 18) {
    console.log("You are a minor.");
}
else if (age === 18) {
    console.log("You just became an adult.");
}
else {
    console.log("You are an adult.");
}
console.log("Conditional statements executed successfully.");
```



```
You just became an adult.
Conditional statements executed successfully.
```

JavaScript Conditional Statements

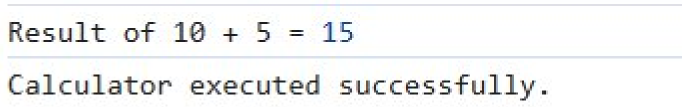
```
//switch case
let day = 3;
let dayName;
switch (day) {
    case 1:
        dayName = "Monday";
        break;
    case 2:
        dayName = "Tuesday";
        break;
    case 3:
        dayName = "Wednesday";
        break;
    case 4:
        dayName = "Thursday";
        break;
    case 5:
        dayName = "Friday";
        break;
    case 6:
        dayName = "Saturday";
        break;
    case 7:
        dayName = "Sunday";
        break;
    default:
        dayName = "Invalid day";
}
console.log("Day name is:", dayName);
console.log("Switch case executed successfully.");
```



```
Day name is: Wednesday
Switch case executed successfully.
```

JavaScript Conditional Statements

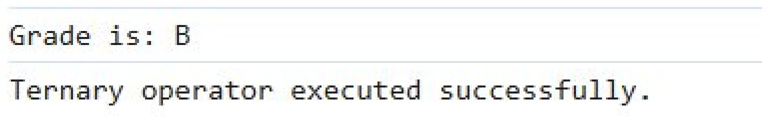
```
//swich case calculator
let numA = 10;
let numB = 5;
let operator = '+';
let result;
switch (operator) {
    case '+':
        result = numA + numB;
        break;
    case '-':
        result = numA - numB;
        break;
    case '*':
        result = numA * numB;
        break;
    case '/':
        result = numA / numB;
        break;
    case '%':
        result = numA % numB;
        break;
    default:
        result = "Invalid operator";
}
console.log(`Result of ${numA} ${operator} ${numB}
=`, result);
console.log("Calculator executed successfully.");
```



```
Result of 10 + 5 = 15
Calculator executed successfully.
```

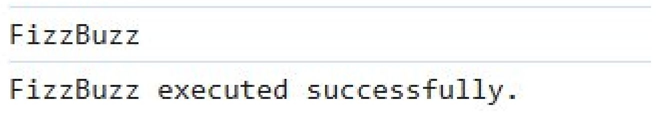
JavaScript Conditional Statements

```javascript
//ternary operator
let score = 85;
let grade = (score >= 90) ? 'A' :
    (score >= 80) ? 'B' :
        (score >= 70) ? 'C' :
            (score >= 60) ? 'D' : 'F';
console.log("Grade is:", grade);
console.log("Ternary operator executed successfully.");
```



```
Grade is: B
Ternary operator executed successfully.
```

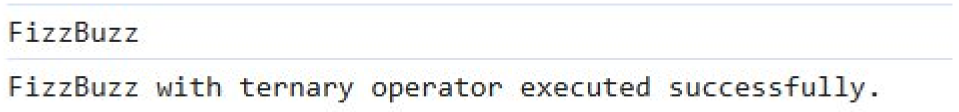
```javascript
//exercise fizz buzz
let number = 15;
if (number % 3 === 0 && number % 5 === 0) {
    console.log("FizzBuzz");
}
else if (number % 3 === 0) {
    console.log("Fizz");
}
else if (number % 5 === 0) {
    console.log("Buzz");
}
else {
    console.log(number);
}
console.log("FizzBuzz executed successfully.");
```



```
FizzBuzz
FizzBuzz executed successfully.
```

JavaScript Conditional Statements

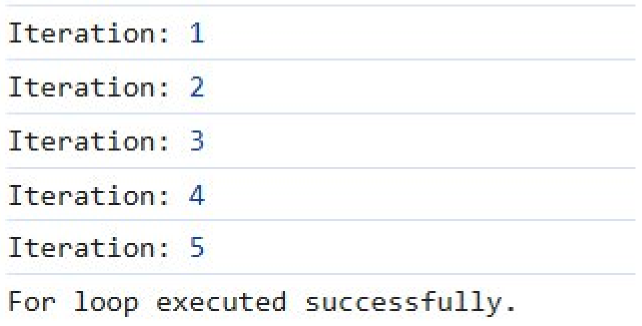
```
//excercise fizz buzz using ternary operator
let numFizzBuzz = 30;
let output = (numFizzBuzz % 3 === 0 && numFizzBuzz % 5 ===
0) ? "FizzBuzz" :
    (numFizzBuzz % 3 === 0) ? "Fizz" :
        (numFizzBuzz % 5 === 0) ? "Buzz" : numFizzBuzz;
console.log(output);
console.log("FizzBuzz with ternary operator executed
successfully.");
```



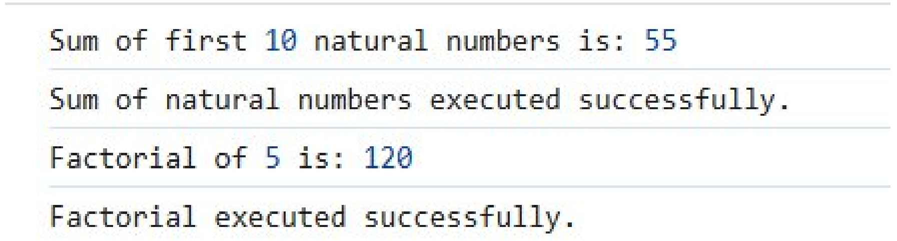
```
FizzBuzz
FizzBuzz with ternary operator executed successfully.
```

JavaScript For loop

```
//For loop
for (let i = 1; i <= 5; i++) {
    console.log("Iteration:", i);
}
console.log("For loop executed successfully.");


//For loop excercise sum of first n natural numbers
let n = 10;
let sumNatural = 0;
for (let i = 1; i <= n; i++) {
    sumNatural += i;
}
console.log("Sum of first", n, "natural numbers is:", 
sumNatural);
console.log("Sum of natural numbers executed 
successfully.");

//For loop excercise factorial of a number
let numFactorial = 5;
let factorial = 1;
for (let i = 1; i <= numFactorial; i++) {
    factorial *= i;
}
console.log("Factorial of", numFactorial, "is:", factorial);
console.log("Factorial executed successfully.");
```



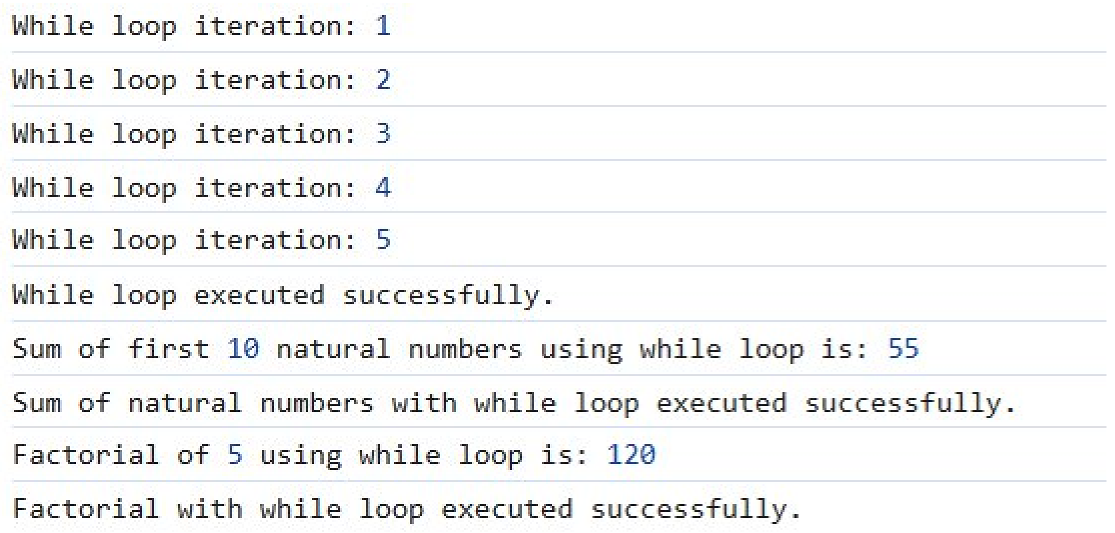
```
Iteration: 1
Iteration: 2
Iteration: 3
Iteration: 4
Iteration: 5
For loop executed successfully.
```



```
Sum of first 10 natural numbers is: 55
Sum of natural numbers executed successfully.
Factorial of 5 is: 120
Factorial executed successfully.
```

JavaScript While loop

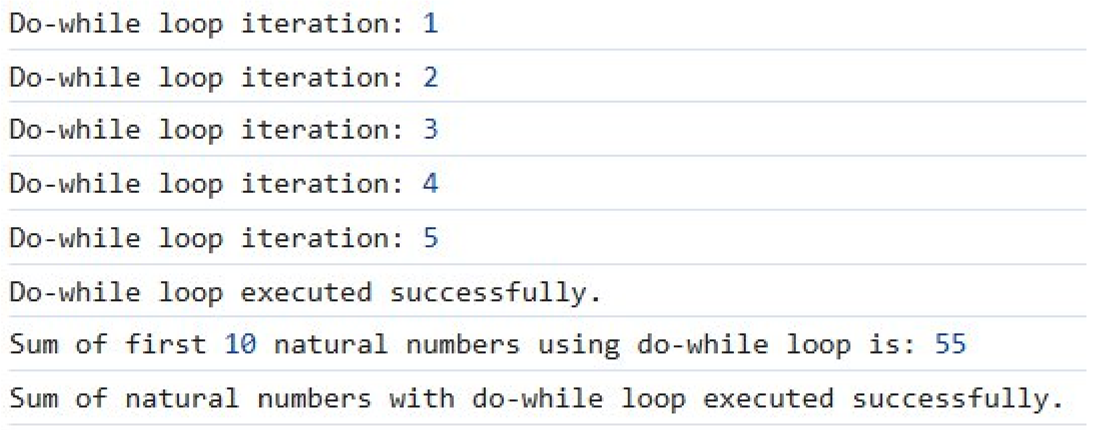
```
//While loop
let countWhile = 1;
while (countWhile <= 5) {
    console.log("While loop iteration:", countWhile);
    countWhile++;
}
console.log("While loop executed successfully.");
//While loop excercise sum of first n natural numbers
let m = 10;
let sumWhile = 0;
let i = 1;
while (i <= m) {
    sumWhile += i;
    i++;
}
console.log("Sum of first", m, "natural numbers using
while loop is:", sumWhile);
console.log("Sum of natural numbers with while loop
executed successfully.");
//While loop excercise factorial of a number
let numFactWhile = 5;
let factWhile = 1;
let j = 1;
while (j <= numFactWhile) {
    factWhile *= j;
    j++;
}
console.log("Factorial of", numFactWhile, "using while
loop is:", factWhile);
console.log("Factorial with while loop executed
successfully.");
```



```
While loop iteration: 1
While loop iteration: 2
While loop iteration: 3
While loop iteration: 4
While loop iteration: 5
While loop executed successfully.
Sum of first 10 natural numbers using while loop is: 55
Sum of natural numbers with while loop executed successfully.
Factorial of 5 using while loop is: 120
Factorial with while loop executed successfully.
```

JavaScript do While loop

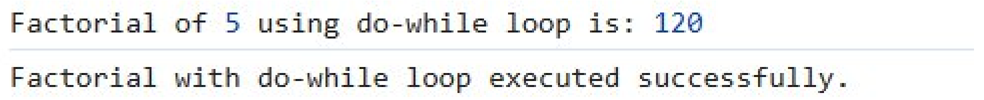
```
//Do-while loop
let countDoWhile = 1;
do {
    console.log("Do-while loop iteration:", countDoWhile);
    countDoWhile++;
}
while (countDoWhile <= 5);
console.log("Do-while loop executed successfully.");

//Do-while loop excercise sum of first n natural numbers
let k = 10;
let sumDoWhile = 0;
let l = 1;
do {
    sumDoWhile += l;
    l++;
}
while (l <= k);
console.log("Sum of first", k, "natural numbers using do-while loop is:", sumDoWhile);
console.log("Sum of natural numbers with do-while loop executed successfully.");
```



```
Do-while loop iteration: 1
Do-while loop iteration: 2
Do-while loop iteration: 3
Do-while loop iteration: 4
Do-while loop iteration: 5
Do-while loop executed successfully.
Sum of first 10 natural numbers using do-while loop is: 55
Sum of natural numbers with do-while loop executed successfully.
```

JavaScript do While loop

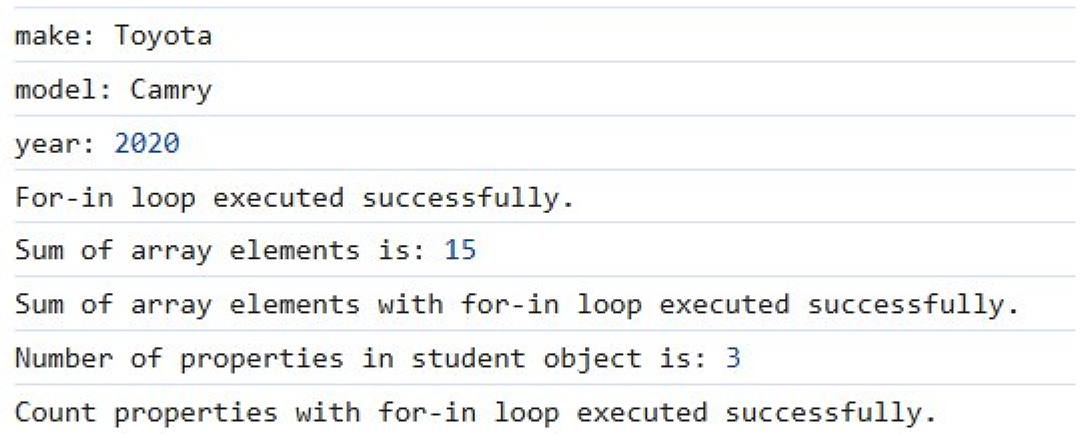
```javascript
//Do-while loop excercise factorial of a number
let numFactDoWhile = 5;
let factDoWhile = 1;
let m1 = 1;
do {
    factDoWhile *= m1;
    m1++;
}
while (m1 <= numFactDoWhile);
console.log("Factorial of", numFactDoWhile, "using do-
while loop is:", factDoWhile);
console.log("Factorial with do-while loop executed
successfully.");
```



```
Factorial of 5 using do-while loop is: 120
Factorial with do-while loop executed successfully.
```

JavaScript for in loop

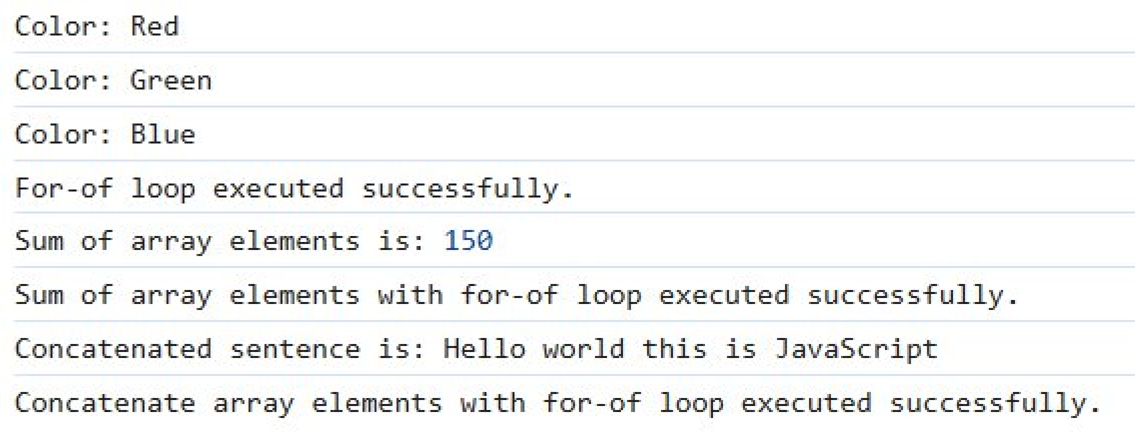
```
//for in loop
let car = {
   make: "Toyota",
   model: "Camry",
   year: 2020
};
for (let key in car) {
   console.log(key + ":", car[key]);
}
console.log("For-in loop executed successfully.");
//for in loop excercise sum of array elements
let numbersArray = [1, 2, 3, 4, 5];
let sumArray = 0;
for (let index in numbersArray) {
   sumArray += numbersArray[index];
}
console.log("Sum of array elements is:", sumArray);
console.log("Sum of array elements with for-in loop executed
successfully.");
//for in loop excercise count properties in an object
let student = {
   name: "Alice",
   age: 22,
   major: "Computer Science"
};
let propertyCount = 0;
for (let prop in student) {
   propertyCount++;
}
console.log("Number of properties in student object is:",
propertyCount);
console.log("Count properties with for-in loop executed successfully.");
```



```
make: Toyota
model: Camry
year: 2020
For-in loop executed successfully.
Sum of array elements is: 15
Sum of array elements with for-in loop executed successfully.
Number of properties in student object is: 3
Count properties with for-in loop executed successfully.
```

JavaScript for of loop

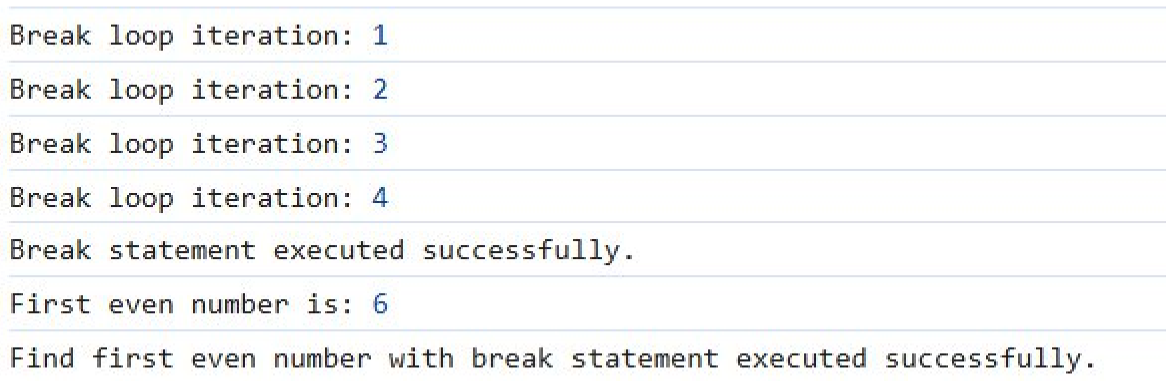
```
//for of loop
let colors = ["Red", "Green", "Blue"];
for (let color of colors) {
    console.log("Color:", color);
}
console.log("For-of loop executed successfully.");
//for of loop excercise sum of array elements
let numsArray = [10, 20, 30, 40, 50];
let sumNums = 0;
for (let num of numsArray) {
    sumNums += num;
}
console.log("Sum of array elements is:", sumNums);
console.log("Sum of array elements with for-of loop executed
successfully.");
//for of loop excercise concatenate array elements
let words = ["Hello", "world", "this", "is", "JavaScript"];
let sentence = "";
for (let word of words) {
    sentence += word + " ";
}
console.log("Concatenated sentence is:", sentence.trim());
console.log("Concatenate array elements with for-of loop
executed successfully.");
```



```
Color: Red
Color: Green
Color: Blue
For-of loop executed successfully.
Sum of array elements is: 150
Sum of array elements with for-of loop executed successfully.
Concatenated sentence is: Hello world this is JavaScript
Concatenate array elements with for-of loop executed successfully.
```

JavaScript break continue

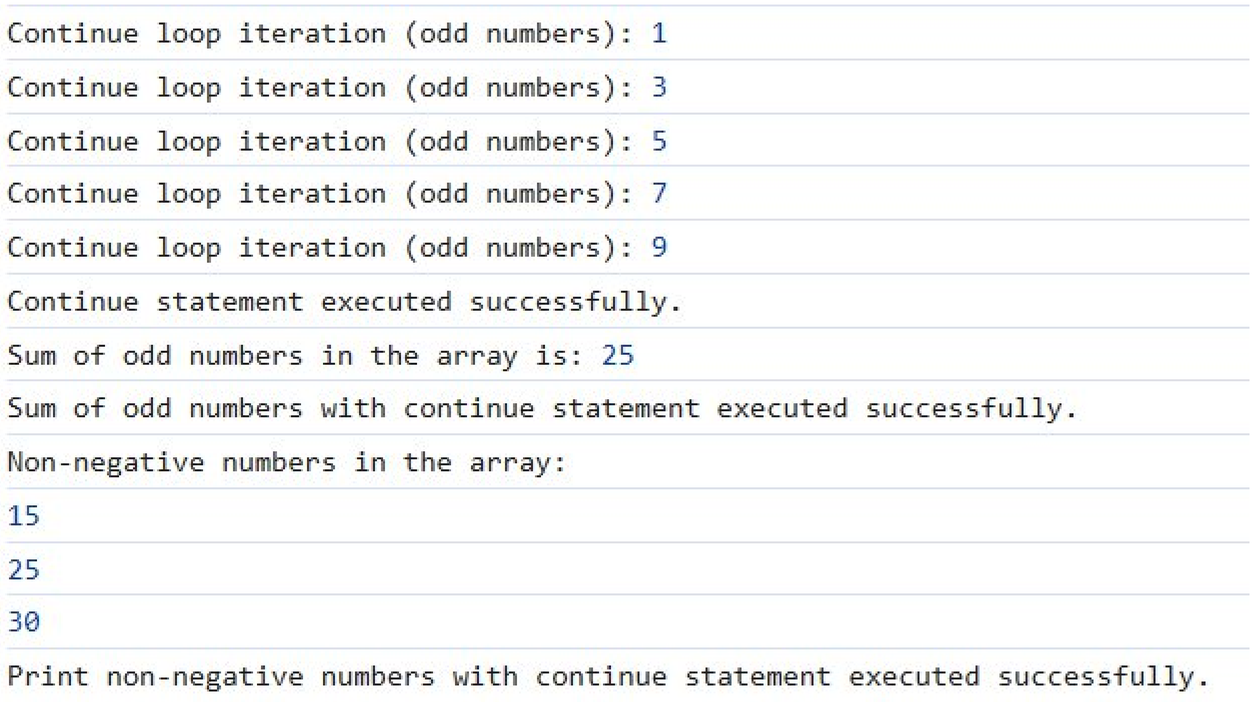
```
//break statement
for (let i = 1; i <= 10; i++) {
    if (i === 5) {
        break;
    }
    console.log("Break loop iteration:", i);
}
console.log("Break statement executed successfully."
//break statement excercise find first even number in
an array
let numList = [1, 3, 5, 6, 7, 8];
let firstEven = null;
for (let num of numList) {
    if (num % 2 === 0) {
        firstEven = num;
        break;
    }
}
if (firstEven !== null) {
    console.log("First even number is:", firstEven);
}
else {
    console.log("No even number found.");
}
console.log("Find first even number with break
statement executed successfully.");
```



```
Break loop iteration: 1
Break loop iteration: 2
Break loop iteration: 3
Break loop iteration: 4
Break statement executed successfully.
First even number is: 6
Find first even number with break statement executed successfully.
```

JavaScript break continue

```javascript
//continue statement
for (let i = 1; i <= 10; i++) {
   if (i % 2 === 0) {
      continue;
   }
   console.log("Continue loop iteration (odd numbers):", i);
}
console.log("Continue statement executed successfully.");
//continue statement excercise sum of odd numbers in an array
let numArray = [1, 2, 3, 4, 5, 6, 7, 8, 9];
let sumOdd = 0;
for (let num of numArray) {
   if (num % 2 === 0) {
      continue;
   }
   sumOdd += num;
}
console.log("Sum of odd numbers in the array is:", sumOdd);
console.log("Sum of odd numbers with continue statement executed
successfully.");
//continue statement excercise print non-negative numbers from an array
let mixedNumbers = [-10, 15, -20, 25, 30, -5];
console.log("Non-negative numbers in the array:");
for (let num of mixedNumbers) {
   if (num < 0) {
      continue;
   }
   console.log(num);
}
console.log("Print non-negative numbers with continue statement executed
successfully.");
```



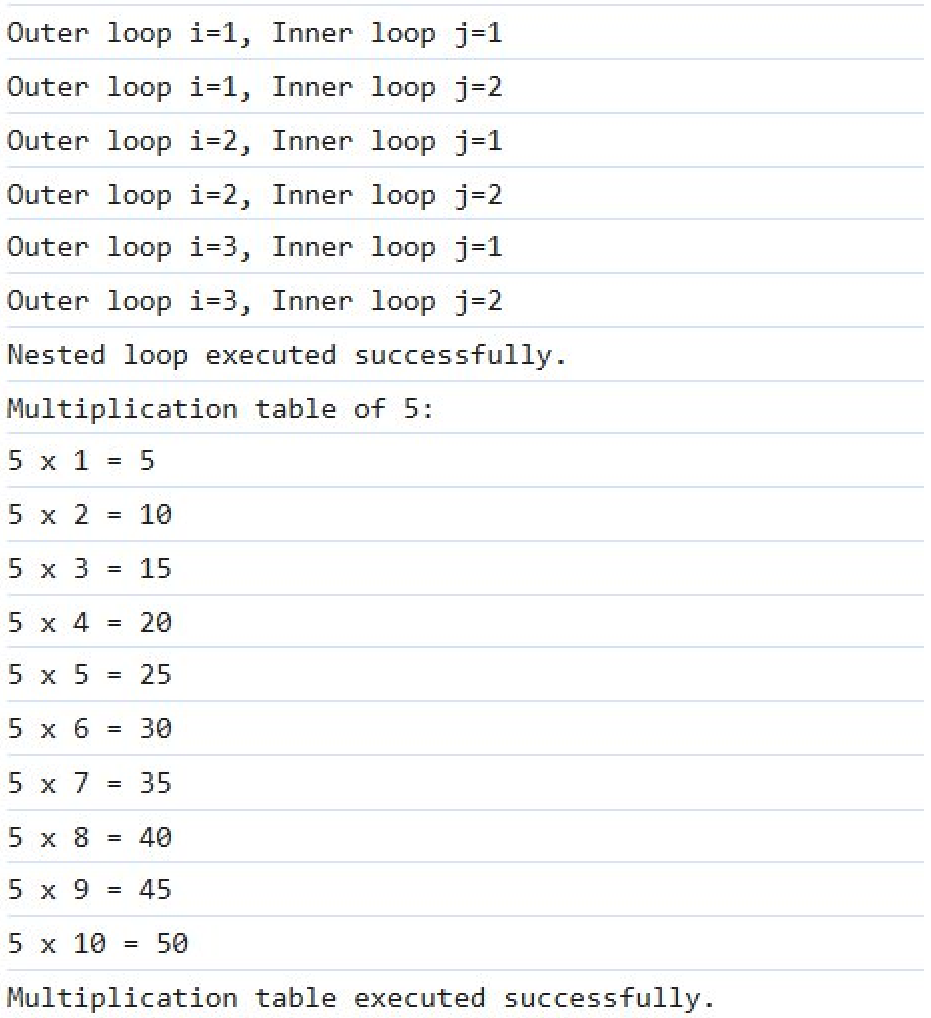
```
Continue loop iteration (odd numbers): 1
Continue loop iteration (odd numbers): 3
Continue loop iteration (odd numbers): 5
Continue loop iteration (odd numbers): 7
Continue loop iteration (odd numbers): 9
Continue statement executed successfully.
Sum of odd numbers in the array is: 25
Sum of odd numbers with continue statement executed successfully.
Non-negative numbers in the array:
15
25
30
Print non-negative numbers with continue statement executed successfully.
```

JavaScript infinite loops

```
let infiniteNum = 1;
while (true) {
    console.log("Infinite numbers:", infiniteNum);
    infiniteNum++;
    if (infiniteNum > 5) { // Added a break condition to
prevent actual infinite loop
        break;
    }
}
console.log("Print numbers indefinitely executed
successfully (with break condition).");
```

JavaScript nested loops

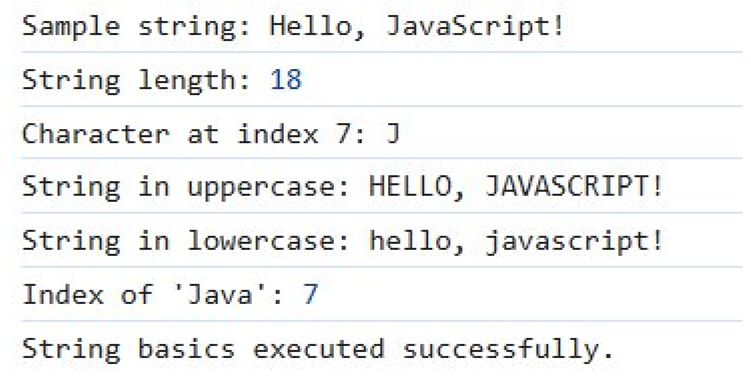
```javascript
//nested loop
for (let i = 1; i <= 3; i++) {
    for (let j = 1; j <= 2; j++) {
        console.log(`Outer loop i=${i}, Inner loop j=${j}`);
    }
}
console.log("Nested loop executed successfully.");

//nested loop excercise multiplication table
let tableNum = 5;
console.log(`Multiplication table of ${tableNum}:`);
for (let i = 1; i <= 10; i++) {
    console.log(`${tableNum} x ${i} = ${tableNum * i}`);
}
console.log("Multiplication table executed
successfully.");
```



```
Outer loop i=1, Inner loop j=1
Outer loop i=1, Inner loop j=2
Outer loop i=2, Inner loop j=1
Outer loop i=2, Inner loop j=2
Outer loop i=3, Inner loop j=1
Outer loop i=3, Inner loop j=2
Nested loop executed successfully.
Multiplication table of 5:
5 x 1 = 5
5 x 2 = 10
5 x 3 = 15
5 x 4 = 20
5 x 5 = 25
5 x 6 = 30
5 x 7 = 35
5 x 8 = 40
5 x 9 = 45
5 x 10 = 50
Multiplication table executed successfully.
```

JavaScript string

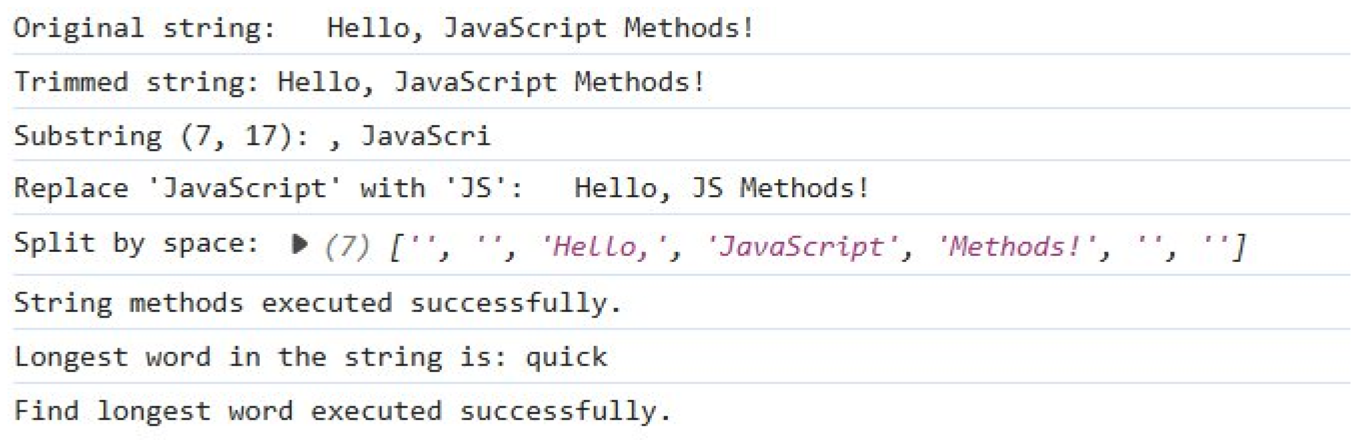
```
//string basics
let sampleString = "Hello, JavaScript!";
console.log("Sample string:", sampleString);
console.log("String length:", sampleString.length);
console.log("Character at index 7:", sampleString.charAt(7));
console.log("String in uppercase:", sampleString.toUpperCase());
console.log("String in lowercase:", sampleString.toLowerCase());
console.log("Index of 'Java':", sampleString.indexOf("Java"));
console.log("String basics executed successfully.");
```



```
Sample string: Hello, JavaScript!
String length: 18
Character at index 7: J
String in uppercase: HELLO, JAVASCRIPT!
String in lowercase: hello, javascript!
Index of 'Java': 7
String basics executed successfully.
```

JavaScript string methods

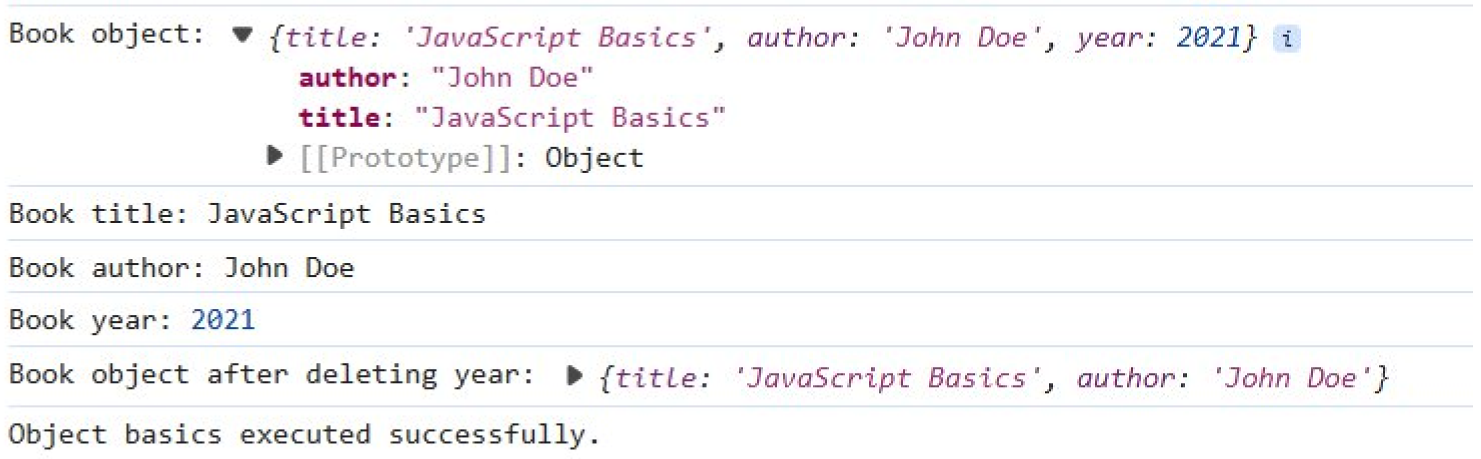
```
//string methods
let methodString = "  Hello, JavaScript Methods!  ";
console.log("Original string:", methodString);
console.log("Trimmed string:", methodString.trim());
console.log("Substring (7, 17):", methodString.substring(7, 17));
console.log("Replace 'JavaScript' with 'JS':", methodString.replace("JavaScript",
"JS"));
console.log("Split by space:", methodString.split(" "));
console.log("String methods executed successfully.");

//string methods excercise find the longest word in a string
let longString = "The quick brown fox jumps over the lazy dog";
let wordsArray = longString.split(" ");
let longestWord = "";
for (let word of wordsArray) {
    if (word.length > longestWord.length) {
        longestWord = word;
    }
}
console.log("Longest word in the string is:", longestWord);
console.log("Find longest word executed successfully.");
```



```
Original string:   Hello, JavaScript Methods!
Trimmed string: Hello, JavaScript Methods!
Substring (7, 17): , JavaScri
Replace 'JavaScript' with 'JS':   Hello, JS Methods!
Split by space: ▶ (7) ['', '', 'Hello,', 'JavaScript', 'Methods!', '', '']
String methods executed successfully.
Longest word in the string is: quick
Find longest word executed successfully.
```

JavaScript object

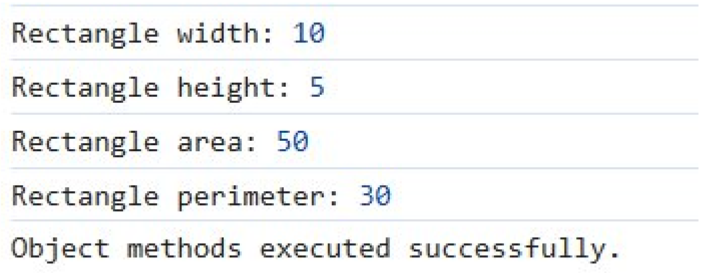
```
//object basics
let book = {
    title: "JavaScript Basics",
};
book.author = "John Doe";
book.year = 2021;
console.log("Book object:", book);
console.log("Book title:", book.title);
console.log("Book author:", book.author);
console.log("Book year:", book.year);
delete book.year;
console.log("Book object after deleting year:", book);
console.log("Object basics executed successfully.");
```



```
Book object:  ▼ {title: 'JavaScript Basics', author: 'John Doe', year: 2021} i
                  author: "John Doe"
                  title: "JavaScript Basics"
                ▶ [[Prototype]]: Object
Book title: JavaScript Basics
Book author: John Doe
Book year: 2021
Book object after deleting year:  ▶ {title: 'JavaScript Basics', author: 'John Doe'}
Object basics executed successfully.
```

JavaScript object

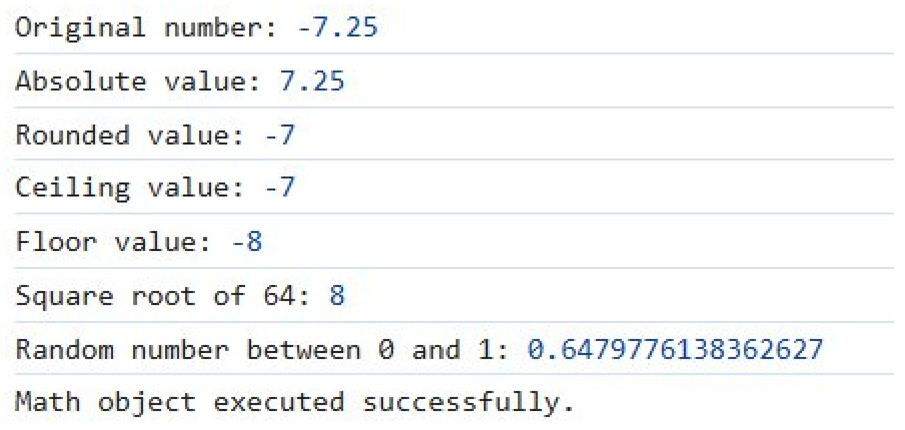
```
//object methods This keyword
let rectangle = {
   width: 10,
   height: 5,
   area: function() {
      return this.width * this.height;
   },
   perimeter: function() {
      return 2 * (this.width + this.height);
   }
};
console.log("Rectangle width:", rectangle.width);
console.log("Rectangle height:", rectangle.height);
console.log("Rectangle area:", rectangle.area());
console.log("Rectangle perimeter:", rectangle.perimeter());
console.log("Object methods executed successfully.");
```



```
Rectangle width: 10
Rectangle height: 5
Rectangle area: 50
Rectangle perimeter: 30
Object methods executed successfully.
```

JavaScript math object

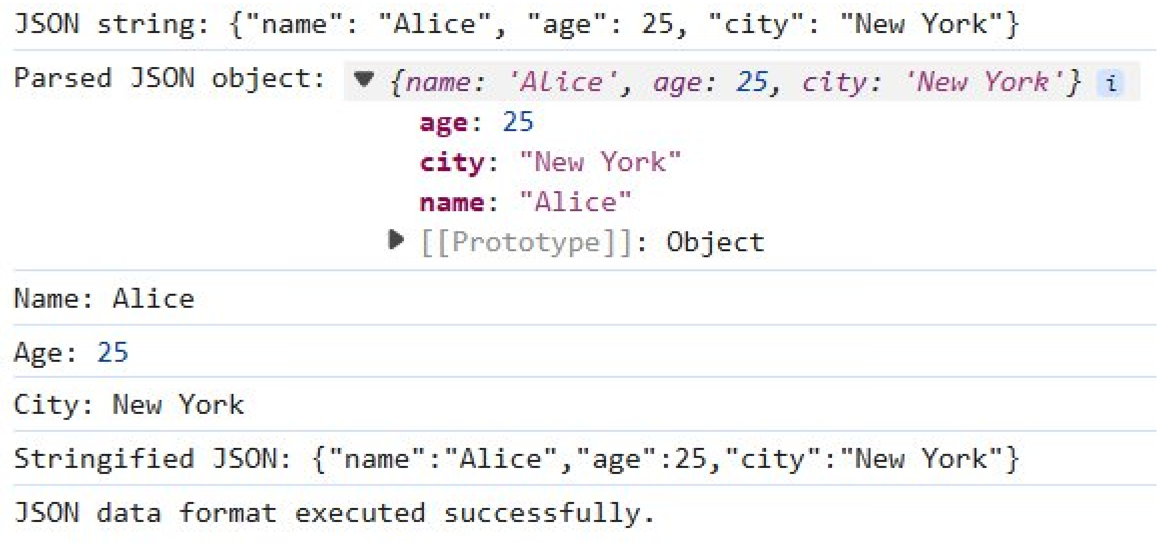
```
//math object
let numMath = -7.25;
console.log("Original number:", numMath);
console.log("Absolute value:", Math.abs(numMath));
console.log("Rounded value:", Math.round(numMath));
console.log("Ceiling value:", Math.ceil(numMath));
console.log("Floor value:", Math.floor(numMath));
console.log("Square root of 64:", Math.sqrt(64));
console.log("Random number between 0 and 1:", Math.random());
console.log("Math object executed successfully.");
```



```
Original number: -7.25
Absolute value: 7.25
Rounded value: -7
Ceiling value: -7
Floor value: -8
Square root of 64: 8
Random number between 0 and 1: 0.6479776138362627
Math object executed successfully.
```

JavaScript json data format

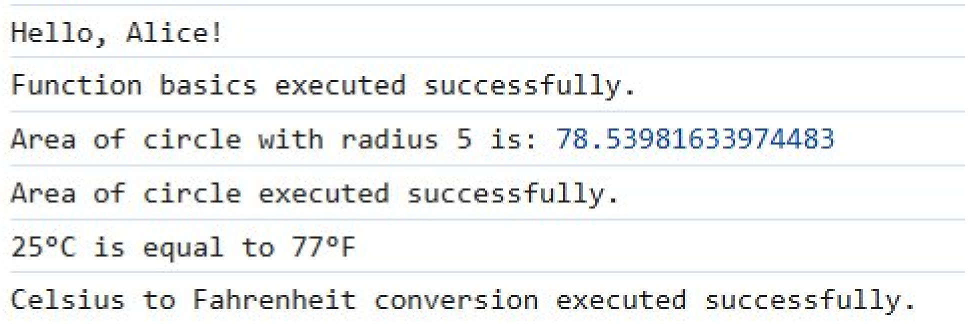
```
//json data format
let jsonString = '{"name": "Alice", "age": 25, "city": "New York"}';
let jsonObject = JSON.parse(jsonString);
console.log("JSON string:", jsonString);
console.log("Parsed JSON object:", jsonObject);
console.log("Name:", jsonObject.name);
console.log("Age:", jsonObject.age);
console.log("City:", jsonObject.city);
let newJsonString = JSON.stringify(jsonObject);
console.log("Stringified JSON:", newJsonString);
console.log("JSON data format executed successfully.");
```



```
JSON string: {"name": "Alice", "age": 25, "city": "New York"}
Parsed JSON object: ▼ {name: 'Alice', age: 25, city: 'New York'} i
                        age: 25
                        city: "New York"
                        name: "Alice"
                      ▶ [[Prototype]]: Object
Name: Alice
Age: 25
City: New York
Stringified JSON: {"name":"Alice","age":25,"city":"New York"}
JSON data format executed successfully.
```

JavaScript function

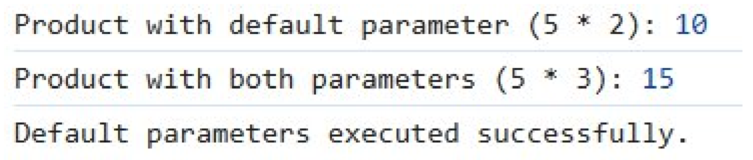
```
//function basics
function greetUser(name) {
    return `Hello, ${name}!`;
}
let greetingMessage = greetUser("Alice");
console.log(greetingMessage);
console.log("Function basics executed successfully.");
//function excercise calculate the area of a circle
function areaOfCircle(radius) {
    return Math.PI * radius * radius;
}
let radius = 5;
let area = areaOfCircle(radius);
console.log(`Area of circle with radius ${radius} is:`, area);
console.log("Area of circle executed successfully.");
//function excercise convert Celsius to Fahrenheit
function celsiusToFahrenheit(celsius) {
    return (celsius * 9 / 5) + 32;
}
let celsiusTemp = 25;
let fahrenheitTemp = celsiusToFahrenheit(celsiusTemp);
console.log(`${celsiusTemp}°C is equal to ${fahrenheitTemp}°F`);
console.log("Celsius to Fahrenheit conversion executed successfully.");
```



```
Hello, Alice!
Function basics executed successfully.
Area of circle with radius 5 is: 78.53981633974483
Area of circle executed successfully.
25°C is equal to 77°F
Celsius to Fahrenheit conversion executed successfully.
```

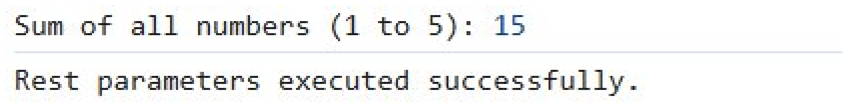
JavaScript function

```
//default parameters
function multiply(a, b = 2) {
   return a * b;
}
let product1 = multiply(5);
let product2 = multiply(5, 3);
console.log("Product with default parameter (5 * 2):", product1);
console.log("Product with both parameters (5 * 3):", product2);
console.log("Default parameters executed successfully.");
```



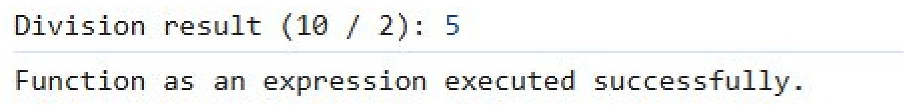
```
Product with default parameter (5 * 2): 10
Product with both parameters (5 * 3): 15
Default parameters executed successfully.
```

```
//rest parameters
function sumAll(...numbers) {
   return numbers.reduce((acc, curr) => acc + curr, 0);
}
let totalSum = sumAll(1, 2, 3, 4, 5);
console.log("Sum of all numbers (1 to 5):", totalSum);
console.log("Rest parameters executed successfully.");
```



```
Sum of all numbers (1 to 5): 15
Rest parameters executed successfully.
```

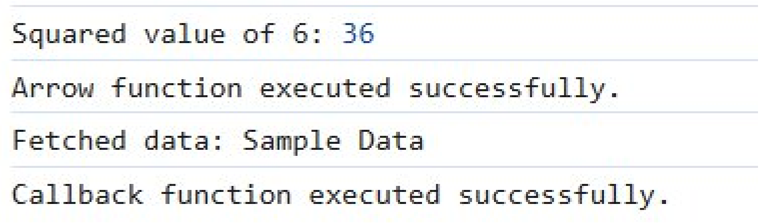
```
//function as an expression
const divide = function(a, b) {
   return a / b;
};
let divisionResult = divide(10, 2);
console.log("Division result (10 / 2):", divisionResult);
console.log("Function as an expression executed successfully.");
```



```
Division result (10 / 2): 5
Function as an expression executed successfully.
```

JavaScript function

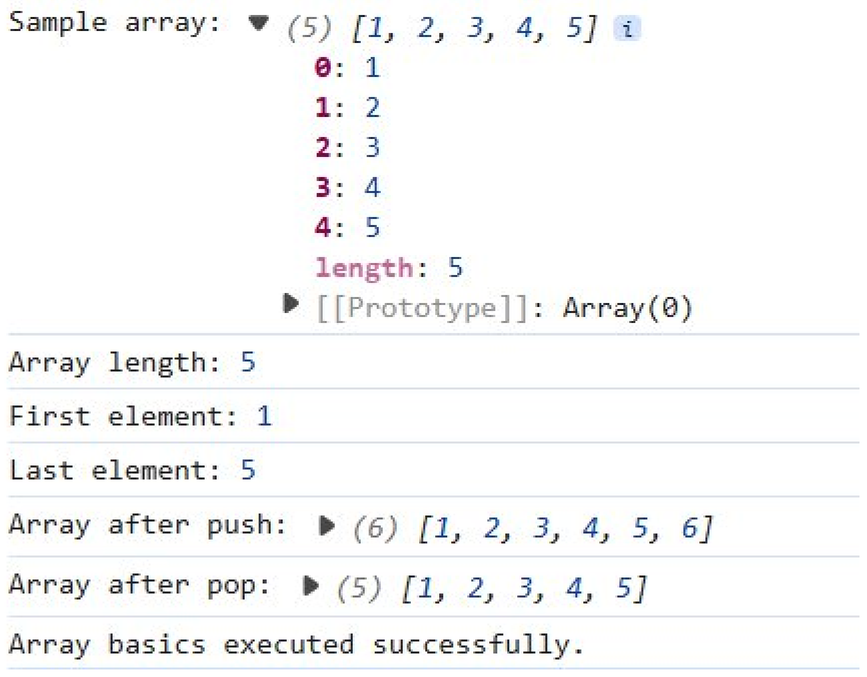
```
//arrow function
const square = (x) => x * x;
let squaredValue = square(6);
console.log("Squared value of 6:", squaredValue);
console.log("Arrow function executed successfully.");


//callback function
function fetchData(callback) {
   setTimeout(() => {
      const data = "Sample Data";
      callback(data);
   }
   , 1000);
}
fetchData((data) => {
   console.log("Fetched data:", data);
   console.log("Callback function executed successfully.");
});
```



```
Squared value of 6: 36
Arrow function executed successfully.
Fetched data: Sample Data
Callback function executed successfully.
```

JavaScript Array

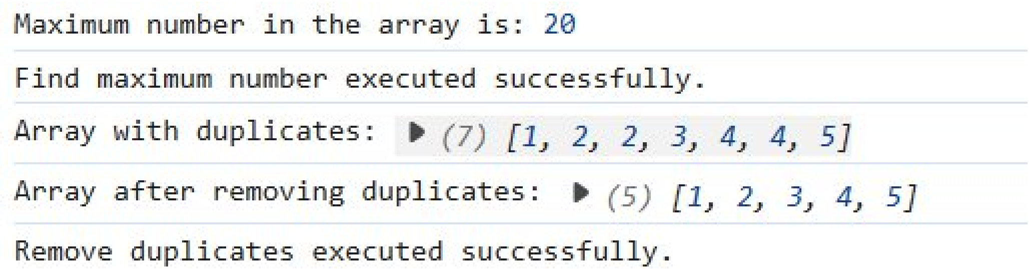
```
//array basics
let sampleArray = [1, 2, 3, 4, 5];
console.log("Sample array:", sampleArray);
console.log("Array length:", sampleArray.length);
console.log("First element:", sampleArray[0]);
console.log("Last element:", sampleArray[sampleArray.length - 1]);
sampleArray.push(6);
console.log("Array after push:", sampleArray);
sampleArray.pop();
console.log("Array after pop:", sampleArray);
console.log("Array basics executed successfully.");
```



```
Sample array:  ▼ (5) [1, 2, 3, 4, 5] i
                   0: 1
                   1: 2
                   2: 3
                   3: 4
                   4: 5
                   length: 5
                 ▶ [[Prototype]]: Array(0)
Array length: 5
First element: 1
Last element: 5
Array after push:  ▶ (6) [1, 2, 3, 4, 5, 6]
Array after pop:  ▶ (5) [1, 2, 3, 4, 5]
Array basics executed successfully.
```

JavaScript Array

```
//array excercise find the maximum number in an array
let numArrayExcercise = [10, 5, 8, 20, 15];
let maxNum = numArrayExcercise[0];
for (let num of numArrayExcercise) {
    if (num > maxNum) {
        maxNum = num;
    }
}
console.log("Maximum number in the array is:", maxNum);
console.log("Find maximum number executed successfully.");


//array excercise remove duplicates from an array
let arrayWithDuplicates = [1, 2, 2, 3, 4, 4, 5];
let uniqueArray = [];
for (let num of arrayWithDuplicates) {
    if (!uniqueArray.includes(num)) {
        uniqueArray.push(num);
    }
}
console.log("Array with duplicates:", arrayWithDuplicates);
console.log("Array after removing duplicates:", uniqueArray);
console.log("Remove duplicates executed successfully.");
```



```
Maximum number in the array is: 20
Find maximum number executed successfully.
Array with duplicates:  ▶ (7) [1, 2, 2, 3, 4, 4, 5]
Array after removing duplicates:  ▶ (5) [1, 2, 3, 4, 5]
Remove duplicates executed successfully.
```

JavaScript error

```
//different type of errors
//syntax error
// Uncomment the code below to see the syntax error
// console.log("This is a syntax error example"
// console.log("This line will not be executed due to the syntax error above.");
// console.log("Syntax error example executed successfully.");
//reference error
// Uncomment the code below to see the reference error
// console.log("Value of undefined variable:", undefinedVariable);
// console.log("This line will not be executed due to the reference error above.");
// console.log("Reference error example executed successfully.");
//type error
// Uncomment the code below to see the type error
// let numType = 5;
// numType.toUpperCase(); // This will cause a type error
// console.log("This line will not be executed due to the type error above.");
// console.log("Type error example executed successfully.");
//range error
// Uncomment the code below to see the range error
// let arrRange = new Array(-1); // This will cause a range error
// console.log("This line will not be executed due to the range error above.");
// console.log("Range error example executed successfully.");
//eval error
// Uncomment the code below to see the eval error
// eval("alert('Hello')"); // Using alert in Node.js will cause an eval error
// console.log("This line will not be executed due to the eval error above.");
// console.log("Eval error example executed successfully.");
//URI error
// Uncomment the code below to see the URI error
// decodeURIComponent("%"); // This will cause a URI error
// console.log("This line will not be executed due to the URI error above.");
```

JavaScript error

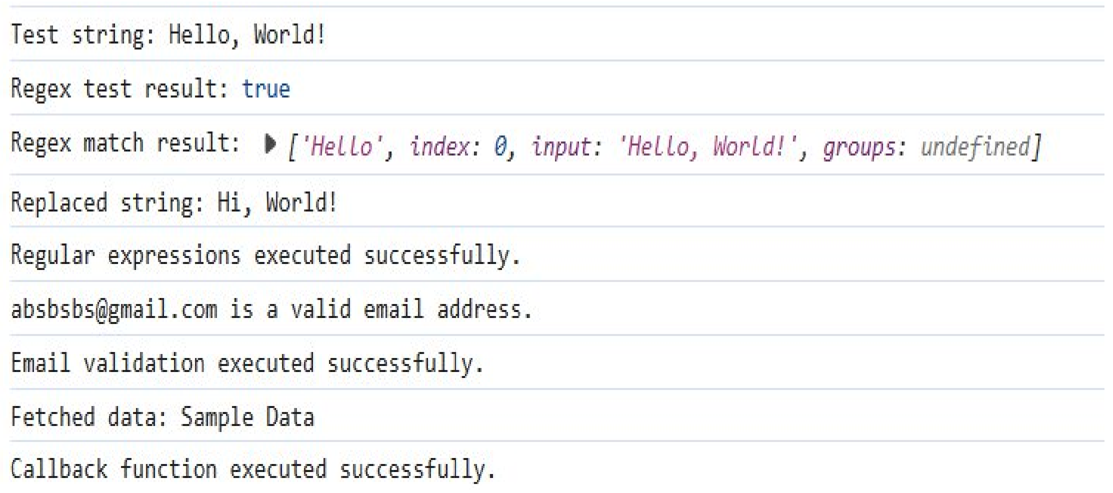
```
//error handling using try-catch and finally
try {
   let result = 10 / 0; // This will not throw an error in JavaScript
   if (!isFinite(result)) {
      throw new Error("Division by zero is not allowed.");
   }
   console.log("Result is:", result);
}
catch (error) {
   console.log("Error caught:", error.message);
}
finally {
   console.log("Finally block executed.");
}
console.log("Error handling executed successfully.");


//error handling excercise parse JSON safely
let invalidJsonString = '{"name": "Alice", "age": 25, "city": "New York"'; // Missing closing brace
try {
   let parsedData = JSON.parse(invalidJsonString);
   console.log("Parsed JSON data:", parsedData);
}
catch (error) {
   console.log("Error parsing JSON:", error.message);
}
finally {
   console.log("JSON parsing attempt finished.");
}
console.log("JSON parsing executed successfully.");
```

JavaScript error

```
//custom error
class ValidationError extends Error {
   constructor(message) {
      super(message);
      this.name = "ValidationError";
   }
}
function validateAge(age) {
   if (age < 0 || age > 120) {
      throw new ValidationError("Age must be between 0 and 120.");
   }
   return true;
}
try {
   validateAge(150);
}
catch (error) {
   if (error instanceof ValidationError) {
      console.log("Validation error caught:", error.message);
   }
   else {
      console.log("Other error caught:", error.message);
   }
}
finally {
   console.log("Age validation attempt finished.");
}
console.log("Custom error handling executed successfully.");
```

JavaScript regular expression

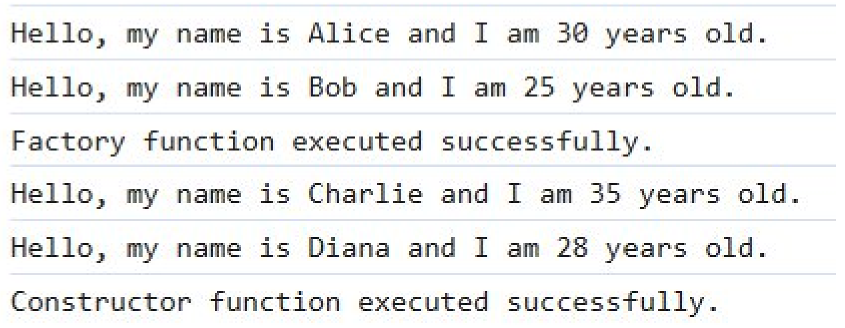
```javascript
//regular expressions
let regex = /hello/i;
let testString = "Hello, World!";
console.log("Test string:", testString);
console.log("Regex test result:", regex.test(testString));
let matchResult = testString.match(regex);
console.log("Regex match result:", matchResult);
let replacedString = testString.replace(regex, "Hi");
console.log("Replaced string:", replacedString);
console.log("Regular expressions executed successfully.");


//regular expressions excercise validate email format
let emailRegex = /^[^\s@]+@[^\s@]+\.[^\s@]+$/;
let emailToTest = "absbsbs@gmail.com";
if (emailRegex.test(emailToTest)) {
    console.log(emailToTest, "is a valid email address.");
}
else {
    console.log(emailToTest, "is not a valid email address.");
}
console.log("Email validation executed successfully.");
```



```
Test string: Hello, World!
Regex test result: true
Regex match result:  ▶ ['Hello', index: 0, input: 'Hello, World!', groups: undefined]
Replaced string: Hi, World!
Regular expressions executed successfully.
absbsbs@gmail.com is a valid email address.
Email validation executed successfully.
Fetched data: Sample Data
Callback function executed successfully.
```

JavaScript oops

```javascript
//factory function
function createPerson(name, age) {
    return {
        name: name,
        age: age,
        greet: function() {
            console.log(`Hello, my name is ${this.name} and I am ${this.age} years old.`);
        }
    };
}
let person1 = createPerson("Alice", 30);
let person2 = createPerson("Bob", 25);
person1.greet();
person2.greet();
console.log("Factory function executed successfully.");


//constructor function
function Person(name, age) {
    this.name = name;
    this.age = age;
    this.greet = function() {
        console.log(`Hello, my name is ${this.name} and I am ${this.age} years old.`);
    }
}
let personA = new Person("Charlie", 35);
let personB = new Person("Diana", 28);
personA.greet();
personB.greet();
console.log("Constructor function executed successfully.");
```



```
Hello, my name is Alice and I am 30 years old.
Hello, my name is Bob and I am 25 years old.
Factory function executed successfully.
Hello, my name is Charlie and I am 35 years old.
Hello, my name is Diana and I am 28 years old.
Constructor function executed successfully.
```

JavaScript asyncronous

```
//asynchronous non-blocking code
console.log("Start of asynchronous code.");
setTimeout(() => {
    console.log("This message is from setTimeout after 2 seconds.");
}
, 2000);
console.log("End of asynchronous code.");
console.log("Asynchronous code executed successfully.");
```

```
//what is callback function
function fetchDataCallback(callback) {
    setTimeout(() => {
        const data = "Sample Data from Callback";
        callback(data);
    }
    , 1000);
}
fetchDataCallback((data) => {
    console.log("Fetched data using callback:", data);
    console.log("Callback function executed successfully.");
});
```

JavaScript asynchronous

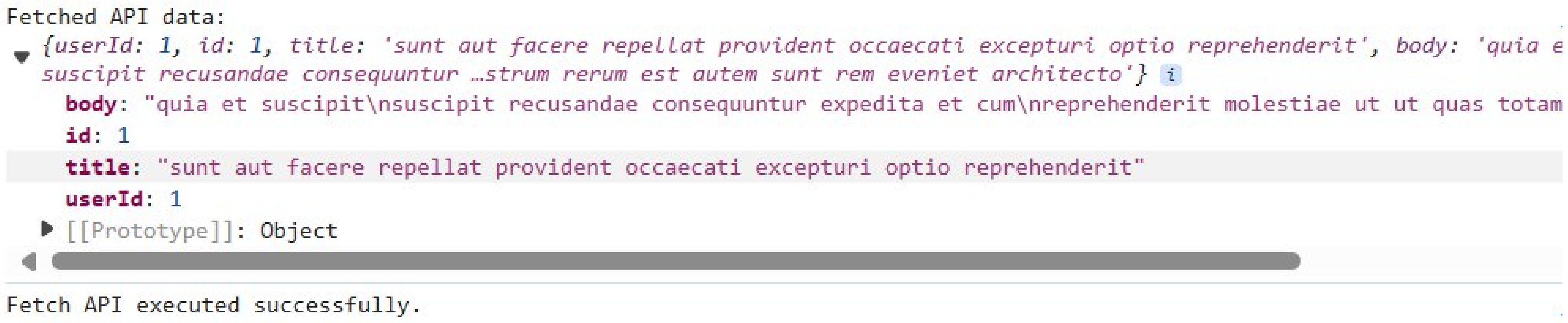
```
//what is promise
function fetchDataPromise() {
    return new Promise((resolve, reject) => {
        setTimeout(() => {
            const data = "Sample Data from Promise";
            resolve(data);
        }
        , 1000);
    });
}
fetchDataPromise()
    .then((data) => {
    console.log("Fetched data using promise:", data);
    console.log("Promise executed successfully.");
}
    )
    .catch((error) => {
    console.log("Error fetching data:", error);
}
    );
```

JavaScript asynchronous

```
//what is async await
async function fetchDataAsync() {
    try {
        const data = await fetchDataPromise();
        console.log("Fetched data using async/await:", data);
        console.log("Async/await executed successfully.");
    }
    catch (error) {
        console.log("Error fetching data:", error);
    }
}
fetchDataAsync();
```

JavaScript asynchronous

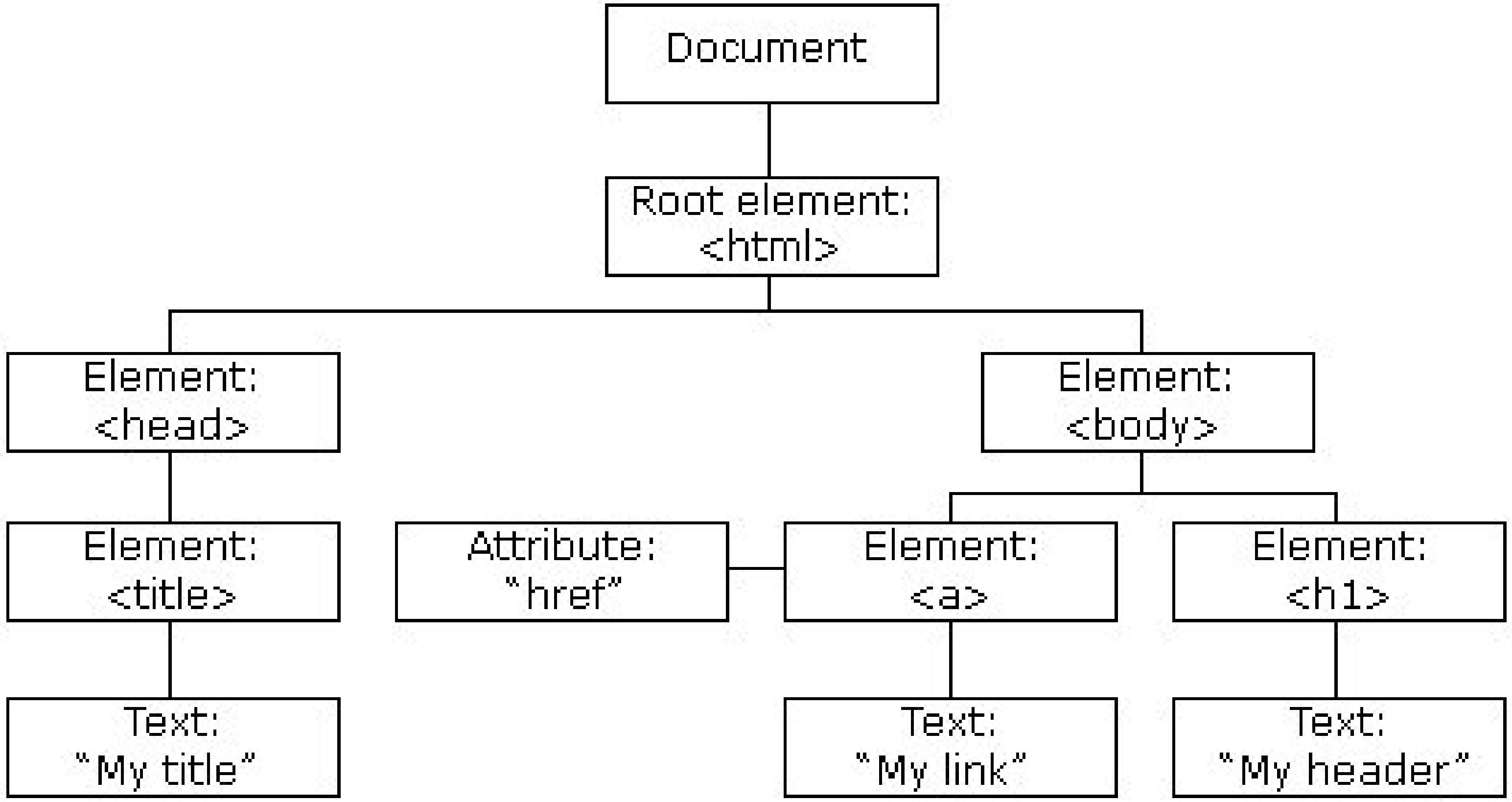
```
//ajax call with fetch api

async function fetchApiData() {
    try {
        const response = await fetch('https://jsonplaceholder.typicode.com/posts/1');
        const data = await response.json();
        console.log("Fetched API data:", data);
        console.log("Fetch API executed successfully.");
    }
    catch (error) {
        console.log("Error fetching API data:", error);
    }
}
fetchApiData();
```



```
Fetched API data:
▼ {userId: 1, id: 1, title: 'sunt aut facere repellat provident occaecati excepturi optio reprehenderit', body: 'quia e
  suscipit recusandae consequuntur …strum rerum est autem sunt rem eveniet architecto'} i
    body: "quia et suscipit\nsuscipit recusandae consequuntur expedita et cum\nreprehenderit molestiae ut ut quas totam
    id: 1
    title: "sunt aut facere repellat provident occaecati excepturi optio reprehenderit"
    userId: 1
  ▶ [[Prototype]]: Object

Fetch API executed successfully.
```

DOM

The Document Object Model (DOM) in JavaScript represents the structure of an HTML or XML document as a tree of objects. It's a programming interface that allows JavaScript to interact with and manipulate the content, structure, and style of a web page



Web storage API

```
//local storage
localStorage.setItem("username", "john_doe");
const storedUsername =
localStorage.getItem("username");
console.log("Stored username in local storage:",
storedUsername);
localStorage.removeItem("username");
console.log("Username removed from local storage.");
console.log("Local storage executed successfully.");
```

```
//session storage
sessionStorage.setItem("sessionID", "abc123");
const storedSessionID =
sessionStorage.getItem("sessionID");
console.log("Stored session ID in session storage:",
storedSessionID);
sessionStorage.removeItem("sessionID");
console.log("Session ID removed from session
storage.");
console.log("Session storage executed successfully.");
```

Web storage API

```
//cookies
document.cookie
    = "user=JaneDoe; expires=Fri, 31 Dec 2024 23:59:59 GMT; path=/";
console.log("Cookie set:", document.cookie);



// Function to get a specific cookie by name
function getCookie(name) {
    const value = `; ${document.cookie}`;
    const parts = value.split(`; ${name}=`);
    if (parts.length === 2) return parts.pop().split(';').shift();
}
const userCookie = getCookie("user");
console.log("Retrieved cookie 'user':", userCookie);
```

Web storage API

```
// Function to delete a specific cookie by name
function deleteCookie(name) {
    document.cookie = `${name}=; expires=Thu, 01 Jan 1970 00:00:00 GMT; path=/`;
}
deleteCookie("user");
console.log("Cookie 'user' deleted.");
console.log("Cookies executed successfully.");
```

Thank you!!