Chapter 2

JavaScript Syntax and Basics

let name = "Rupali";

console.log("Hello, " + name);

Example:

let age = 45;

console.log(age); // Correct

Example:

// This is a single-line comment

/\*

This is a multi-line comment

\*/

Example:

let result = 15 + 10;

Example:

var keyword = "var is a keyword example";

console.log(keyword); // Output: "var is a keyword example"

Examples of Valid Identifiers:

let name = "Rupali";

let $dollarValue = 100;

let \_private = "secret";

Examples of Invalid Identifiers:

// let 123name; // Starts with a digit

// let if; // Uses a reserved keyword

Example:

let sum = 10 +

20;

// Output: 30

Example:

function addValues(x, y) {

return x + y;

}

let result = addValues(15, 10);

console.log("The sum is: " + result);

Example of Issue with ASI:

return

{

key: "value"

};

Example:

console.log("Line1\nLine2");

**Example:**

// Hoisting with var

console.log(testVar); // Output: undefined (hoisted declaration)

var testVar = 10;

console.log(testVar); // Output: 10

// Scope issue with var

if (true) {

var count = 5;

}

console.log(count); // Output: 5 (accessible outside block)

// Block scope

if (true) {

let counter = 10;

console.log(counter); // Output: 10

}

// console.log(counter); // Error: counter is not defined

// TDZ in action

// console.log(message); // Error: Cannot access 'message' before initialization

let message = "Hello, Rupali!";

console.log(message); // Output: Hello, Rupali!

// Immutable reference

const pi = 3.14;

// pi = 3.14159; // Error: Assignment to constant variable

// Modifiable object properties

const config = { theme: "dark" };

config.theme = "light";

console.log(config.theme); // Output: light

1. **Scope Differences:**

function testScope() {

if (true) {

var x1 = 10;

let y1 = 20;

const z1 = 30;

}

console.log(x1); // Output: 10

// console.log(y1); // Error: y is not defined

// console.log(z1); // Error: z is not defined

}

* 1. **Hoisting Differences:**

console.log(a); // Output: undefined

var a = 5;

// console.log(b); // Error: Cannot access 'b' before initialization

let b = 10;

// console.log(c); // Error: Cannot access 'c' before initialization

const c = 15;

1. **Immutability with const:**

const myArray = [1, 2, 3];

myArray.push(4);

console.log(myArray); // Output: [1, 2, 3, 4]

// myArray = [5, 6]; // Error: Assignment to constant variable

**Example:**

*let greet = "Hello, Rupali!";*

*console.log(greet.length); // Output: 14*

*console.log(greet.toUpperCase()); // Output: HELLO, RUPALI!*

*console.log(greet.includes("Rupa")); // Output: true*

Example:

let numb1 = 40;

let numb2 = 3.14;

console.log(numb1 + numb2); // Output: 43.14

console.log((numb2).toFixed(1)); // Output: 3.1

console.log(parseInt("100px")); // Output: 100

console.log(Number.isNaN("hello" \* 2)); // Output: true

#### **Example:**

let isStudent = true;

let hasID = false;

if (isStudent && hasID) {

console.log("Eligible for student discount");

} else {

console.log("Not eligible for student discount");

}

// Output: Not eligible for student discount

Example:

if ("Hello") {

console.log("This is truthy!"); // Output: This is truthy!

}

if (0) {

console.log("This won't run."); // Won't execute since 0 is falsy

}

Example:

let x1 = 10, y1 = 4;

console.log(x1 + y1); // Output: 14

console.log(x1 – y1); // Output: 6

console.log(x1 \* y1); // Output: 40

console.log(x1 / y1); // Output: 2.5

console.log(x1 % y1); // Output: 2

let a = 5;

a++;

console.log(a); // Output: 6

let b = 10;

b--;

console.log(b); // Output: 9

Example:

let num1 = 10, num2 = 5, num3 = "10";

console.log(num1 == num3); // Output: true (loose equality)

console.log(num1 === num3); // Output: false (strict equality)

console.log(num1 > num2); // Output: true

console.log(num1 <= num2); // Output: false

console.log(num1 !== num3); // Output: true

Example:

let a = 10, b = 5, c = 15;

// Logical AND

if (a > b && c > a) {

console.log("Both conditions are true.");

} else {

console.log("At least one condition is false.");

}

// Output: Both conditions are true.

// Logical OR

if (a < b || c > a) {

console.log("At least one condition is true.");

}

// Output: At least one condition is true.

// Logical NOT

let isLoggedIn = false;

if (!isLoggedIn) {

console.log("User is not logged in.");

}

// Output: User is not logged in.

Precedence Example:

let result = 10 + 5 \* 2; // Multiplication has higher precedence than addition.

console.log(result); // Output: 20

To override precedence, use parentheses:

let result = (10 + 5) \* 2;

console.log(result); // Output: 30

**Syntax:**

if (condition) {

// Code to execute if the condition is true

}

**Example:**

let age = 18;

if (age >= 18) {

console.log("You are **adult now**.");

}

**With Multiple Conditions:**

let temperature = 30;

if (temperature > 35) {

console.log("It's a hot day!");

} else if (temperature > 25) {

console.log("It's a warm day.");

} else {

console.log("It's a cool day.");

}

**Syntax:**

if (condition) {

// **execute this** if the condition is true

} else {

// execute **this** if the condition is false

}

Example:

let age\_value = 22;

if (age\_value >= 18) {

console.log("You are an adult.");

} else {

console.log("You are a minor.");

}

Syntax:

switch (expression) {

case value1:

// Code to execute if expression === value1

break;

case value2:

// Code to execute if expression === value2

break;

default:

// Code to execute if no cases match

}

Example:

let fruit = "apple";

switch (fruit) {

case "banana":

console.log("Yellow fruit");

break;

case "apple":

console.log("Red or green fruit");

break;

case "orange":

console.log("Orange fruit");

break;

default:

console.log("Unknown fruit");

}

Nested if-else Example:

let age = 16;

if (age >= 18) {

console.log("You are eligible to vote.");

} else if (age >= 16) {

console.log("You can apply for a driving license.");

} else {

console.log("You are not eligible for voting or driving.");

}

Complex switch Example:

let command = "start";

switch (command) {

case "start":

console.log("The engine is starting.");

break;

case "stop":

console.log("The engine is stopping.");

break;

case "pause":

console.log("The engine is paused.");

break;

default:

console.log("Unknown command.");

}

Example:

let color = "red";

switch (color) {

case "red":

console.log("Stop");

case "yellow":

console.log("Caution");

case "green":

console.log("Go");

}

**Syntax**:

for (initialization; condition; increment/decrement) {

// Code block to execute

}

**Example 1: Iterating through a range of numbers**

for (let i = 1; i <= 5; i++) {

console.log(`Iteration number: ${i}`);

}

**Example 2: Iterating over an array**

const fruits = ['Apple', 'Banana', 'Cherry'];

for (let i = 0; i < fruits.length; i++) {

console.log(fruits[i]);

}

**Syntax:**

while (condition) {

// Code block to execute

}

**Example 1: Print numbers until a condition is met**

let counter = 1;

while (counter <= 3) {

console.log(`Counter is at: ${counter}`);

counter++;

}

**Example 2: Breaking out of a loop**

let sum = 0;

let number = 1;

while (true) {

sum += number;

if (sum > 10) break; // Exit the loop once sum exceeds 10

number++;

}

console.log(`Sum: ${sum}`);

**Syntax:**

do {

// Code block to execute

} while (condition);

**Example 1: Executing a block at least once**

let i = 0;

do {

console.log(`Value of i: ${i}`);

i++;

} while (i < 3);

**Example 2: Validating user input**

let password;

do {

password = prompt("Enter your password:");

} while (password !== "secure123");

console.log("Password accepted!");

**Structure of try-catch-finally**

try {

// Code that may throw an error

} catch (error) {

// Code to handle the error

} finally {

// Code that executes regardless of an error

}

**Examples**

**1. Handling a Basic Error**

try {

let num = 10;

let result = num / 0; // No error; division by zero in JavaScript is Infinity

console.log(result);

} catch (error) {

console.log("An error occurred:", error.message);

} finally {

console.log("Execution completed.");

}

**2. Handling a Reference Error**

try {

console.log(value); // ReferenceError: value is not defined

} catch (error) {

console.log("An error occurred:", error.message);

} finally {

console.log("Error handling done.");

}

**3. Throwing a Custom Error**

try {

let age = -5;

if (age < 0) {

throw new Error("Age cannot be negative.");

}

console.log("Age is valid.");

} catch (error) {

console.log("An error occurred:", error.message);

} finally {

console.log("Validation complete.");

}

**4. Using finally for Cleanup**

try {

console.log("Opening a file...");

throw new Error("File not found.");

} catch (error) {

console.log("Error:", error.message);

} finally {

console.log("Closing the file...");

}

**Example**:

try {

JSON.parse("{ invalid JSON }"); // SyntaxError

} catch (error) {

console.log("Error name:", error.name);

console.log("Error message:", error.message);

console.log("Error stack:", error.stack);

}

**Example**:

function connectToDatabase() {

console.log("Connecting to database...");

throw new Error("Connection failed!");

}

try {

connectToDatabase();

} catch (error) {

console.log("Caught an error:", error.message);

} finally {

console.log("Closing database connection...");

}

**Example:**

class ValidationError extends Error {

constructor(message) {

super(message);

this.name = "ValidationError";

}

}

try {

throw new ValidationError("Invalid input provided.");

} catch (error) {

console.log(`${error.name}: ${error.message}`);

}

**Example:**

function calculateSum(a, b) {

console.log("Value of a:", a); // Log the value of 'a'

console.log("Value of b:", b); // Log the value of 'b'

let sum = a + b;

console.log("Sum is:", sum); // Log the result

return sum;

}

calculateSum(5, 10);

**Example:**

function isEven(number) {

let result = number % 2 === 0;

return result;

}

console.log(isEven(4));

console.log(isEven(7));

**Example:**

function findMax(a, b) {

debugger; // Execution will pause here

return a > b ? a : b;

}

console.log(findMax(8, 12));

Example:

let array = [1, 2, 3];

console.log(array[5]); // Accessing an index that does not exist

Console Output:

undefined

Example:

function a() {

b();

}

function b() {

c();

}

function c() {

throw new Error("Something went wrong!");

}

a();

Console Output:

Error: Something went wrong!

at c (<anonymous>:8:11)

at b (<anonymous>:4:5)

at a (<anonymous>:2:5)

Example:

function multiply(a, b) {

let result = a \* b;

return result;

}

multiply(3, 4);

Example:

try {

let user = JSON.parse('{"name": "Alice"}'); // Valid JSON

console.log(user.age.toUpperCase()); // Throws error (age is undefined)

} catch (error) {

console.error("An error occurred:", error.message);

} finally {

console.log("Error handling complete.");

}

Example with async/await:

async function fetchData() {

try {

let response = await fetch("https://jsonplaceholder.typicode.com/posts");

let data = await response.json();

console.log(data);

} catch (error) {

console.error("Error fetching data:", error.message);

}

}

fetchData();

Example of ESLint Rule:

// Code with missing 'use strict'

let x = 10; // No error but considered bad practice

With ESLint:

Error: 'x' is defined without 'use strict' mode.