/ Convert °C -> °F Add commentMore actions

// Enter temperature in °C

// 30°C

// Formula: (0°C × 9/5) + 32

// + -> string -> number

// V1

var celsius = +window.prompt("Enter temperature in °C");

var fahrenheit = (celsius \* 9 / 5) + 32

console.log("The temperature in °F is " + fahrenheit)

// console.log("30°C is 86°F")

console.log(celsius + "°C is " + fahrenheit + "°F")

// V2 -> Readability -> DX // Template literal

console.log(`${celsius}°C is ${fahrenheit}°F`)

// Output

// The temperature in °F is 86°F

// 30°C is 86°F

=========================================================================

// Task 1

// Find area of circle given radius

// PI = 3.14

// Enter the radius in cm: 2

// camelCase

// CONSTANT\_CASE

var PI = 3.14;

var radius = window.prompt("Enter the radius in cm:");

// var area = PI \* radius \* radius;

var area = PI \* radius \*\* 2;

console.log(area);

console.log(`The Area for the given ${radius}cm radius is ${area}cm²`);

console.log(

`The Area for the given ${radius}cm radius is ${PI \* radius \*\* 2}cm²`

);

// The Area for the given 2cm radius is 12.56cm²

**Example 1: Variable Interpolation**

javascript

CopyEdit

let name = "Latha";

console.log(`Welcome, ${name}! 😊`); // Output: Welcome, Latha! 😊

✅ name is a **variable**, and it’s inserted into the string.

**🧪 Example 2: Expression Interpolation**

javascript

CopyEdit

let a = 5;

let b = 3;

console.log(`Sum is: ${a + b}`); // Output: Sum is: 8

**✅ 1. if Statement – Simple Decision**

javascript

CopyEdit

let age = 18;

if (age >= 18) {

console.log("You can vote ✅");

}

✔️ Executes only if condition is **true**.

**✅ 2. if...else – Two Choices**

javascript

CopyEdit

let isRainy = true;

if (isRainy) {

console.log("Take an umbrella ☔");

} else {

console.log("Enjoy the sunshine ☀️");

}

**✅ 3. if...else if...else – Multiple Choices**

javascript

CopyEdit

let marks = 85;

if (marks >= 90) {

console.log("Grade: A+");

} else if (marks >= 75) {

console.log("Grade: A");

} else {

console.log("Keep trying!");

}

**✅ 4. Ternary Operator – Short Decision**

javascript

CopyEdit

let age = 17;

let result = (age >= 18) ? "Adult" : "Minor";

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

| **Case Type** | **Example** | **Where it's used** |
| --- | --- | --- |
| **lowercase** | "hello world" | General text, messages, inputs |
| **UPPERCASE** | "HELLO WORLD" | Constants, alerts, emphasis |
| **camelCase** | firstName | ✅ JS variables & functions |
| **PascalCase** | StudentName | ✅ Class or constructor names |
| **snake\_case** | student\_name | Used in some languages (Python) |
| **kebab-case** | student-name | ✅ HTML/CSS or URLs |

**🧪 JavaScript String Case Methods**

| **Method** | **Usage** | **Example** | **Output** |
| --- | --- | --- | --- |
| .toLowerCase() | Converts to lowercase | "HELLO".toLowerCase() | "hello" |
| .toUpperCase() | Converts to uppercase | "hello".toUpperCase() | "HELLO" |
| Custom camelCase | You write it manually | let userName = "Latha"; | camelCase |

**✅ Example:**

javascript

CopyEdit

let name = "LaTHa";

console.log(name.toLowerCase()); // "latha"

console.log(name.toUpperCase()); // "LATHA"

A **bug** is:

❌ A **mistake or error** in your code that causes the program to behave unexpectedly or crash.

**🧠 Simple Examples:**

1️⃣ **Wrong result:**

javascript

CopyEdit

let total = 5 \* 2;

console.log(total); // ✅ 10

// Bug example:

let total = 5 + 2;

console.log(total); // ❌ 7 instead of 10 (wrong operator)

2️⃣ **Spelling mistake:**

javascript

CopyEdit

let name = "Latha";

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

3️⃣ **Missing bracket or syntax:**

javascript

CopyEdit

if (true

console.log("Hello"); // ❌ SyntaxError: missing )

let x = 10;

// 1000 lines

console.log(x);

// const x = 10;

// const y = [4, 5, 8];

// // 1000 lines

// y.push(30);

// console.log(x);

// // z = 0;

const books = [

{ title: "Infinite Jest", rating: 4.5, genre: "Fiction" },

{ title: "The Catcher in the Rye", rating: 3.9, genre: "Fiction" },

{ title: "Sapiens", rating: 4.9, genre: "History" },

{ title: "A Brief History of Time", rating: 4.8, genre: "Science" },

{ title: "Clean Code", rating: 4.7, genre: "Technology" },

];

// Array of objects -> Array of stings

// Case 1:

const getTile = (book) => book.title;

const titles = books.map(getTile);

// const titles = books.map(getTile()); // ❌

// Case 2:

const titles1 = books.map((book) => book.title); // HOF

// Case 3: `this`

function getTile(book) {

return book.title;

}

const titles3 = books.map(getTile);

// Case 4: `this`

const getTile = (book) => {

return book.title;

};

const titles4 = books.map(getTile);

const marks = [90, 84, 82, 95, 30];

const topMark = marks.reduce((max, curr) => {

if (max > curr) {

return max;

} else {

return curr;

}

});

const topMark1 = marks.reduce((max, curr) => {

return max > curr ? max : curr;

});

const topMark2 = marks.reduce((max, curr) => (max > curr ? max : curr));

// Task 1.1

// Use Reduce

// Find the top score

// Output

// 95

const scores = [

{

marks: 32,

name: "Yvette Merritt",

},

{

marks: 57,

name: "Lillian Ellis",

},

{

marks: 22,

name: "Mccall Carter",

},

{

marks: 21,

name: "Pate Collier",

},

{

marks: 91,

name: "Debra Beard",

},

{

marks: 75,

name: "Nettie Hancock",

},

{

marks: 20,

name: "Hatfield Hodge",

},

];

// Task 1.2

// Use Reduce

// Find the top scorer name

// Output

// Debra Beard

const topScorer = scores.reduce((topStudent, student) =>

topStudent.marks > student.marks ? topStudent : student

);

console.log(`${topScorer.name} scored ${topScorer.marks}`);

const marks = [90, 84, 82, 95, 30];

const topMark = marks.reduce((max, curr) => {

if (max > curr) {

return max;

} else {

return curr;

}

});

const topMark1 = marks.reduce((max, curr) => {

return max > curr ? max : curr;

});

const topMark2 = marks.reduce((max, curr) => (max > curr ? max : curr));

// Task 1.1

// Use Reduce

// Find the top score

// Output

// 95

const scores = [

{

marks: 32,

name: "Yvette Merritt",

},

{

marks: 57,

name: "Lillian Ellis",

},

{

marks: 22,

name: "Mccall Carter",

},

{

marks: 21,

name: "Pate Collier",

},

{

marks: 91,

name: "Debra Beard",

},

{

marks: 75,

name: "Nettie Hancock",

},

{

marks: 20,

name: "Hatfield Hodge",

},

];

// Task 1.2

// Use Reduce

// Find the top scorer name

// Output

// Debra Beard

const topScorer = scores.reduce((topStudent, student) =>

topStudent.marks > student.marks ? topStudent : student

);

console.log(`${topScorer.name} scored ${topScorer.marks}`);

const marks = [90, 84, 82, 95, 30];

const topMark = marks.reduce((max, curr) => {

if (max > curr) {

return max;

} else {

return curr;

}

});

const topMark1 = marks.reduce((max, curr) => {

return max > curr ? max : curr;

});

const topMark2 = marks.reduce((max, curr) => (max > curr ? max : curr));

// Task 1.1

// Use Reduce

// Find the top score

// Output

// 95

const scores = [

{

marks: 32,

name: "Yvette Merritt",

},

{

marks: 57,

name: "Lillian Ellis",

},

{

marks: 22,

name: "Mccall Carter",

},

{

marks: 21,

name: "Pate Collier",

},

{

marks: 91,

name: "Debra Beard",

},

{

marks: 75,

name: "Nettie Hancock",

},

{

marks: 20,

name: "Hatfield Hodge",

},

];

// Task 1.2

// Use Reduce

// Find the top scorer name

// Output

// Debra Beard

const topScorer = scores.reduce((topStudent, student) =>

topStudent.marks > student.marks ? topStudent : student

);

console.log(`${topScorer.name} scored ${topScorer.marks}`);

### Table

1. Method name

2. Mutation Yes/No? - No

3. Return Data type - Array

### Chaining

1. No mutation

2. Return Data type - Array

### Stops Chaining (join, includes, some,/)

1. No mutation

2. Data type other than array - boolean, eg.

const sum1 = (n1, n2) => {

return n1 + n2;

};

// 3. Multiple args

// Implicit return

const sum = (n1, n2) => n1 + n2;

function dbl1(n) {

return n \* 2;

}

// 2. One arg (bracket option)

const dbl = (n) => n \* 2;

function msg1() {

return "Hello, Murkesh";

}

// 1. No args

const msg = () => "Hello, Murkesh";

msg();

function getWinStat1(team) {

return team?.stats?.win ? team.stats.win : "Data not found";

}

const getWinStat2 = (team) =>

team?.stats?.win ? team.stats.win : "Data not found";

var x = "fun";

var result = x ? x : "no fun";

const getWinStat = (team) => team?.stats?.win ?? "Data not found";

// var x = sayHello();

sayHello()();

// If function can be treated as value

// function -> value

// function -> First class citizen

// 1. Value can be passed as arg

// 2. Value can be returned

// 3. Value can be assigned

const x = () => "Hi";

function sayHello1() {

return function () {

console.log("Hello!!");

};

}

// step 1

// const sayHello = () => {

// return function () {

// console.log("Hello!!");

// };

// };

const sayHello = () => () => console.log("Hello!!");

// F#

// Scala

// Lisp

// Small

// Haskell

// Function programming style (Paradigm)

// 1. Currying

// 2. Partial Application

// 3. Point-free style

// 4. compose or pipe

// const add = (x) => (y) => (z) => x + y + z;

// const add1 = (x, y, z) => x + y + z;

// console.log(add(8)(9)(10));

// console.log(add1(8, 9, 10));

const add = (x) => (y) => x + y;

const add5 = add(5);

const add2 = add(2);

console.log(add5(15)); // 20

// const add5 = (y) => 5 + y;

console.log(add5(10));

// function add(x) {

// return function (y) {

// return x + y;

// };

// }

// x:5 y:9

console.log(add(5)(9)); // 14

// console.log("Hi");

// console.log("Hi");

// console.log("Hi");

// console.log("Hi");

// console.log("Hi");

// console.log("Hi");

// console.log("Hi");

// console.log("Hi");

// console.log("Hi");

// console.log("Hi");

// Repeat -> Loops

// Initialization; Condition; Increment

for (let i = 1; i <= 3; i++) {

console.log("Hi");

}

for (let i = 1; i <= 3; i = i + 2) {

console.log("Hi");

}

for (let i = 1; i <= 3 / 2; i = i + 2) {

console.log("Hi");

}

const codes = ["alpha", "beta", "gamma", "delta"];

for (let i = 0; i < 4; i++) {

console.log(i, codes[i]);

}

// Task 1.1

// Output

// Hulk contains 4 letters

// Iron man contains 8 letters

// ...

// Thor contains 4 letters

const avengers = [

"Hulk",

"Iron man",

"Black widow",

"Captain america",

"Spider man",

"Thor",

];

// i < 6

console.log("Normal for loop");

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

console.log(`${avengers[i]} contains ${avengers[i].length} letters`);

}

console.log("for..in loop");

for (let i in avengers) {

console.log(`${avengers[i]} contains ${avengers[i].length} letters`);

}

console.log("for..of loop");

for (let avenger of avengers) {

console.log(`${avenger} contains ${avenger.length} letters`);

}

// Task 1.2 - Loop inside printCast

// printCast(avengers)

console.log("printCast");

function printCast(avengers) {

for (let avenger of avengers) {

console.log(`${avenger} contains ${avenger.length} letters`);

}

}

printCast(avengers);

// function printPattern(emoji = "❤️", rows = 5) {

// for (let i = 1; i <= rows; i++) {

// console.log(emoji.repeat(i));

// }

// }

const printPattern = (emoji = "❤️", rows = 5) => {

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

console.log(emoji.repeat(i));

}

};

// Task 1.3.1

printPattern("🐳", 5);

printPattern("🍨", 3);

// Clue: repeat()

// Task 1.3.2

printPattern();

// Output

// ❤️

// ❤️❤️

// ❤️❤️❤️

// ❤️❤️❤️❤️

// ❤️❤️❤️❤️❤️

// Task 1.4

const nums = [80, 43, 72, 47, 20, 21];

const getEvenNumbers = (nums) => {

const evenNums = [];

for (let num of nums) {

if (num % 2 == 0) {

evenNums.push(num);

}

}

return evenNums;

};

console.log(getEvenNumbers(nums));

// Output

// [80, 72, 20];

const books = [

{ title: "Infinite Jest", rating: 4.5, genre: "Fiction" },

{ title: "The Catcher in the Rye", rating: 3.9, genre: "Fiction" },

{ title: "Sapiens", rating: 4.9, genre: "History" },

{ title: "A Brief History of Time", rating: 4.8, genre: "Science" },

{ title: "Clean Code", rating: 4.7, genre: "Technology" },

];

const getTitles = (books) => {

const titles = [];

for (const book of books) {

titles.push(book.title);

}

return titles;

};

console.log(getTitles(books));

// 1. join() Array -> String

// 2. split() String -> Array

const fruits = ["🍎", "🍊", "🍓", "🍍"];

console.log(fruits.join(", "));

// "🍎, 🍊, 🍓, 🍍"

console.log(fruits.join());

const basket = "🍎,🍊|🍓,🍍";

console.log(basket.split(","));

const quote = "Do or Die";

console.log(quote.split(""));

console.log(quote.split(" ")); // ["Do", "or", "Die"]

console.log(quote.split());

## 🚀 Concatenate Arrays (Two Ways)

```js

// ✅ First array

var a1 = [100, 200];

// ✅ Second array

var a5 = [700, 800];

// ✅ Concatenate using spread operator ✨

var a6 = [...a1, ...a5];

console.log(a6);

// 👉 Output: [100, 200, 700, 800]

```

\*\*📝 Explanation:\*\*

\* The `...` (spread) operator \*\*expands\*\* each array's elements

\* You merge both arrays in a single statement, avoiding `concat()`

---

## 🌟 Spread Values into a New Array

```js

// ✅ Base array

var a1 = [100, 200];

// ✅ New array with elements plus spreading ✨

var a7 = [80, ...a1, 300, 400];

console.log(a7);

// 👉 Output: [80, 100, 200, 300, 400]

```

\*\*📝 Explanation:\*\*

\* Elements of `a1` are “spread” \*\*in the middle\*\* of a new array

\* This is extremely useful for mixing arrays with custom values

## day 2

two connection

tcp-email,no loss data

but lag

udp-live steam

## ip packet

sever send small packet

way connection

## HTTP

HTTP2 &3 what happend

HTTP error

1xx-informational

2xx -suscess

3xx-redrect

4xx-not found

5xx-client error

## CRUD-post,get,put,delete

insta like action -post

two engine

\*\*blink -render engine

\*\*v8 -v8

gecko

spidermonkey

HTML PARSER-dom

cssom tree

## shortcut

-- '! + tab'=html example

-- ctrl +/ =block command line

##html forgiving language

## datatype

##primitive datatype

1.java script

2.number

3.string

4.boolean

## Non primitive data type

overloaded operator + two fuction

console.log("cool"+"awesome"); //concatenation

// implicit corerion /conversion

console.log(5+"9") //"5"+"9"->"59"

console.log(5-"9") //5-9->-4

// expilict corerion /conversion

console.log(5+parseInt("9"))

console.log(5+parseint("9.7"))

console.log(5+parsefloat("9.7"))

// only value NaN is not equal in javascript