An **arrow function** is a shorter way to write functions in **JavaScript**. It was introduced in **ES6 (ECMAScript 2015)** and is often used for concise and cleaner code.

**🔹 Syntax**

const functionName = (parameter1, parameter2) => {

// function body

};

**🔹 Example**

// Traditional function

function add(a, b) {

return a + b;

}

// Arrow function

const add = (a, b) => a + b;

console.log(add(2, 3)); // Output: 5

**🔹 When to Use {} and return**

If the function has **one expression**, you can **omit {} and return**:

const square = x => x \* x; // return is implicit

But if you write a block {}, you must use return explicitly:

const square = x => {

return x \* x;

};

**🔹 No Parameters**

If there are **no parameters**, use empty parentheses:

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

**🔹 One Parameter**

If there's **only one parameter**, you can omit the parentheses:

const double = x => x \* 2;

**🔹 Key Feature: Lexical this**

Arrow functions do **not have their own this**. They **inherit this from their surrounding scope**, which is very useful in callbacks.

function Person() {

this.age = 0;

setInterval(() => {

this.age++; // `this` refers to Person

console.log(this.age);

}, 1000);

}

If you used a regular function inside setInterval, you'd need .bind(this).

**🔹 Not Suitable For**

Avoid arrow functions in these cases:

* As object methods (if you need this)
* As constructors (they can’t be used with new)

Here are **practical arrow function examples** for you to **practice**, along with **solutions and explanations**:

## 🧪 Example 1: Basic Addition

### ✅ Task:

Create an arrow function that adds two numbers.

// Write your code here:

### ✅ Solution:

const add = (a, b) => a + b;

console.log(add(4, 6)); // Output: 10

## 🧪 Example 2: Square of a Number

### ✅ Task:

Create an arrow function that returns the square of a number.

// Write your code here:

### ✅ Solution:

const square = x => x \* x;

console.log(square(5)); // Output: 25

## 🧪 Example 3: Return a Greeting

### ✅ Task:

Create an arrow function that returns "Hello, Sanjib!"

// Write your code here:

### ✅ Solution:

const greet = () => "Hello, Sanjib!";

console.log(greet()); // Output: Hello, Sanjib!

## 🧪 Example 4: Array Map with Arrow Function

### ✅ Task:

Use an arrow function to double all numbers in an array using .map().

let numbers = [1, 2, 3, 4, 5];

// Write your code here:

### ✅ Solution:

let numbers = [1, 2, 3, 4, 5];

let doubled = numbers.map(n => n \* 2);

console.log(doubled); // Output: [2, 4, 6, 8, 10]

## 🧪 Example 5: Filter Even Numbers

### ✅ Task:

Use an arrow function to filter even numbers from an array using .filter().

let nums = [1, 2, 3, 4, 5, 6];

// Write your code here:

### ✅ Solution:

let nums = [1, 2, 3, 4, 5, 6];

let evens = nums.filter(n => n % 2 === 0);

console.log(evens); // Output: [2, 4, 6]

## 🧪 Example 6: Lexical this Demo (Advanced)

### ✅ Task:

See how arrow functions inherit this.

function Timer() {

this.seconds = 0;

setInterval(() => {

this.seconds++;

console.log(this.seconds);

}, 1000);

}

new Timer();

This logs 1, 2, 3, ... every second because this refers to Timer's context.

Here are **arrow function examples** involving **callbacks** and **Promises**, with clear explanations and practice opportunities.

## ✅ CALLBACK Examples

### 🧪 Example 1: Arrow Function as a Callback (setTimeout)

#### 🔧 Task:

Use an arrow function inside setTimeout to print a message after 2 seconds.

// Write your code here

#### ✅ Solution:

setTimeout(() => {

console.log("⏳ 2 seconds passed!");

}, 2000);

### 🧪 Example 2: Custom Function with a Callback

#### 🔧 Task:

Write a function that accepts a name and a callback, and the callback prints a greeting.

// Write your code here

#### ✅ Solution:

javascript

CopyEdit

function greetUser(name, callback) {

callback(name);

}

greetUser("Sanjib", name => {

console.log(`Hello, ${name}! 👋`);

});

### 🧪 Example 3: Using Arrow Function in Array forEach()

let fruits = ["Apple", "Banana", "Mango"];

// Use arrow function to print each fruit

fruits.forEach(fruit => console.log(fruit));

## ✅ PROMISE Examples

### 🧪 Example 1: Basic Promise with Arrow Function

#### 🔧 Task:

Create a Promise that resolves after 1 second and logs "Task done!".

// Write your code here

#### ✅ Solution:

javascript

CopyEdit

const myPromise = new Promise((resolve, reject) => {

setTimeout(() => {

resolve("✅ Task done!");

}, 1000);

});

myPromise.then(result => console.log(result));

### 🧪 Example 2: Promise with .then() and Arrow Functions

const fetchData = () => {

return new Promise((resolve, reject) => {

setTimeout(() => {

resolve("📦 Data fetched successfully");

}, 1500);

});

};

fetchData()

.then(data => console.log(data))

.catch(err => console.log("❌ Error:", err));

### 🧪 Example 3: Chaining .then() with Arrow Functions

const getNumber = () => Promise.resolve(5);

getNumber()

.then(n => n \* 2)

.then(n => n + 3)

.then(result => console.log("🎯 Final result:", result)); // 13

Bottom of Form

Bottom of Form

Top of Form

Bottom of Form