**Assignment - 27**

// 1. Double using callback:

function doubleArray(arr, callback) {

const doubledArr = arr.map(num => callback(num));

return doubledArr;

}

const nums = [1, 2, 3, 4, 5];

function double(num) {

return num \* 2;

}

const doubledNums = doubleArray(nums, double);

console.log(doubledNums); // [2, 4, 6, 8, 10]

// 2. String Manipulation:

function manipulateString(str, callback) {

const manipulatedStr = str.toUpperCase();

callback(manipulatedStr);

}

function logstring(str) {

console.log(`The manipulated string is: ${str}`);

}

manipulateString("hello, world!", logstring); // The manipulated string is: HELLO, WORLD!

// 3. Age in Days:

function ageInDays(person, callback) {

const fullName = `${person.firstName} ${person.lastName}`;

const ageInDays = person.age \* 365;

callback(fullName, ageInDays);

}

function logResult(name, age) {

console.log(`The person's full name is ${name} and their age in days is ${age}.`);

}

const person = {

firstName: "Mithun",

lastName: "Sarkar",

age: 20

};

ageInDays(person, logResult); // The person's full name is Mithun Sarkar and their age in days is 7300.

// 4. Arrange in alphabetical order:

const books = [

{ title: "The Great Gatsby", author: "F. Scott Fitzgerald", year: 1925 },

{ title: "To Kill a Mockingbird", author: "Harper Lee", year: 1960 },

{ title: "1984", author: "George Orwell", year: 1949 },

{ title: "Pride and Prejudice", author: "Jane Austen", year: 1813 }

];

function getTitles(books) {

const titles = books.map(book => book.title);

return titles.sort();

}

function logTitles(titles) {

console.log(titles);

}

const titles = getTitles(books);

logTitles(titles); // ["1984", "Pride and Prejudice", "The Great Gatsby", "To Kill a Mockingbird"]

// 5. Greeting Promise:

function greet(name) {

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

if (name) {

resolve(`Hello, ${name}!`);

} else {

reject("Please provide a name.");

}

});

}

greet("Mithun")

.then(message => console.log(message)) // Hello, Mithun!

.catch(error => console.error(error));

// 6. Fetch results asynchronously:

async function fetchData() {

try {

const response = await fetch("https://jsonplaceholder.typicode.com/todos/1");

const data = await response.json();

console.log(data);

} catch (error) {

console.error(error);

}

}

fetchData();

// 7. Multiple requests:

async function fetchData() {

try {

const [todoResponse, postResponse] = await Promise.all([

fetch("https://jsonplaceholder.typicode.com/todos/1"),

fetch("https://jsonplaceholder.typicode.com/posts/1")

]);

const todoData = await todoResponse.json();

const postData = await postResponse.json();

const result = { todo: todoData, post: postData };

console.log(result);

} catch (error) {

console.error(error);

}

}

fetchData();

// 8. Get Data from API and Display it on the browser console:

fetch("https://jsonplaceholder.typicode.com/posts")

.then(response => response.json())

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

.catch(error => console.error(error));

// 9. Error Handling:

fetch("https://jsonplaceholder.typicode.com/posts/123456789")

.then(response => {

if (!response.ok) {

throw new Error("Network response was not ok");

}

return response.json();

})

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

.catch(error => console.error(error.message)); // Network response was not ok