**NAME:SHANTANU K. KHOPE**

**PRN:23070521134**

**Task 1: Dynamic Student Report Card Using Destructuring and Template Literals**

**Problem Statement:**

You are given a student object that contains the details of a student including name, roll number, and marks in 5 subjects.

Using object and array destructuring and template literals, write a program to generate a formatted report card in the console.

**Your program should:**

* Calculate the total and percentage.
* Use destructuring to extract the marks.
* Use template literals to display a multi-line summary like:

Report Card for Riya (Roll No: 101)

-----------------------------------

Subjects: Math: 85, Science: 90, English: 78, History: 88, Art: 92

Total Marks: 433

Percentage: 86.6%

Hint: You must decide the structure of the object and choose how to destructure it effectively.



**Task 2: Build a Custom Greeting Generator with Default Parameters and Arrow Functions**

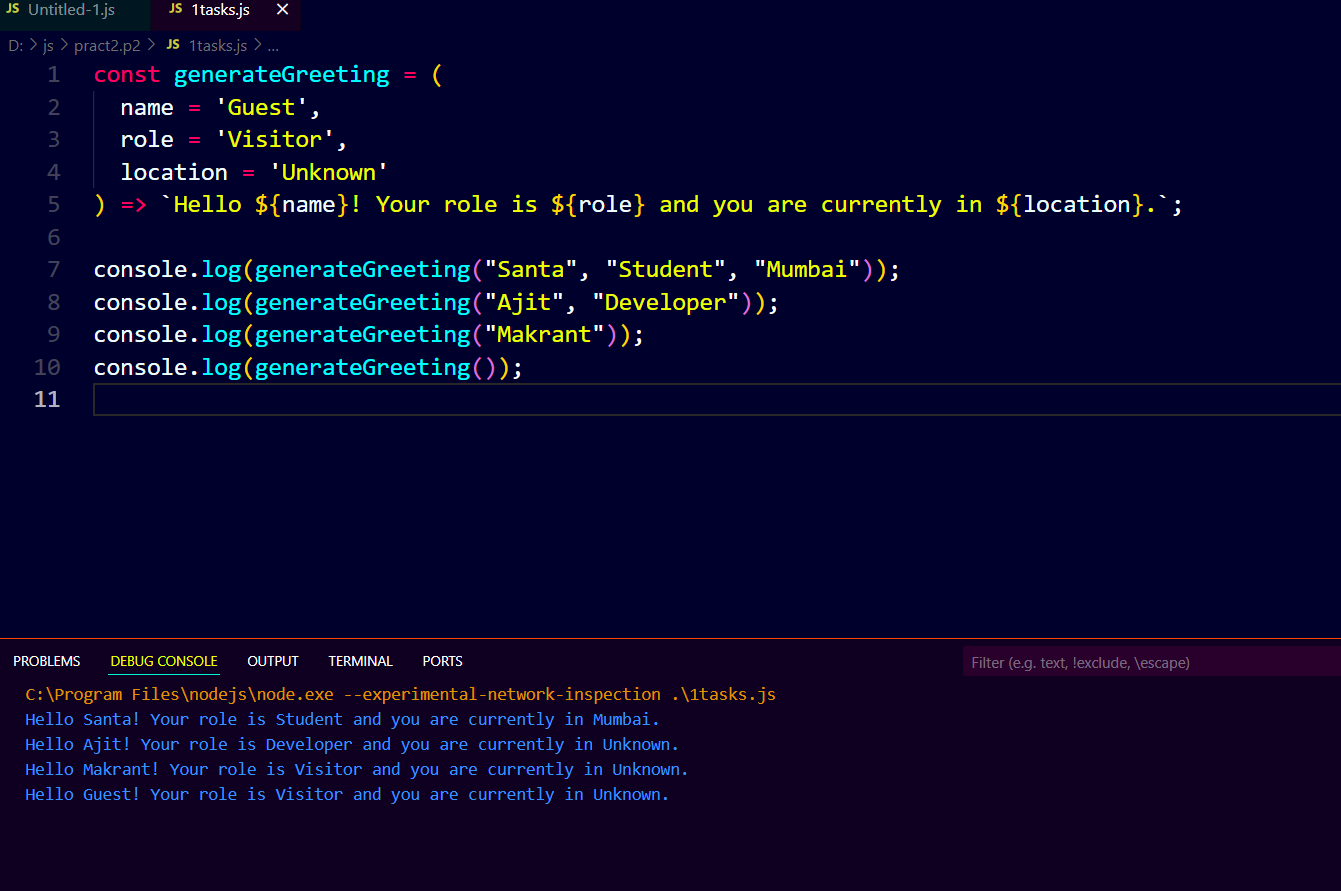
Problem Statement:

* Write a function named generateGreeting that:
* Takes in name, role, and location as parameters (all with default values).
* Uses an arrow function to return a greeting like:

"Hello [name]! Your role is [role] and you are currently in [location]."

**Requirements:**

* Do not use function keyword.
* Use template literals for message construction.
* Call the function with different combinations of arguments and no arguments.
* Only one function should handle all variations without breaking.



**Task 3: Implement a Data Type Analyzer (without typeof)**

**Problem Statement:**

Create a function analyzeData that takes any input and prints:

* What kind of primitive or reference type it is (without using typeof).
* Whether it’s mutable or immutable.
* A summary message.

You cannot use typeof, but must still determine if it's:

* string, number, boolean, undefined, null, symbol, bigint
* array, object, function

You must figure out creative ways to determine the type, such as using constructor checks, Array.isArray, etc.

function analyzeData(input) {

let type = '';

let mutability = '';

let summary = '';

if (input === null) {

type = "null";

mutability = "immutable";

} else if (Array.isArray(input)) {

type = "array";

mutability = "mutable";

} else if (input !== undefined && input.constructor === String) {

type = "string";

mutability = "immutable";

} else if (input !== undefined && input.constructor === Number) {

type = "number";

mutability = "immutable";

} else if (input !== undefined && input.constructor === Boolean) {

type = "boolean";

mutability = "immutable";

} else if (input !== undefined && input.constructor === Symbol) {

type = "symbol";

mutability = "immutable";

} else if (input !== undefined && input.constructor === BigInt) {

type = "bigint";

mutability = "immutable";

} else if (input === undefined) {

type = "undefined";

mutability = "immutable";

} else if (typeof input === "function" || input instanceof Function) { // Avoids typeof, but Function constructor needed

type = "function";

mutability = "immutable";

} else if (input !== null && typeof input === "object") {

type = "object";

mutability = "mutable";

}

summary = `Type: ${type} | Mutability: ${mutability}`;

console.log(summary);

}

// Test cases

analyzeData("hello"); // string, immutable

analyzeData(100); // number, immutable

analyzeData(false); // boolean, immutable

analyzeData(undefined); // undefined, immutable

analyzeData(null); // null, immutable

analyzeData(Symbol("id")); // symbol, immutable

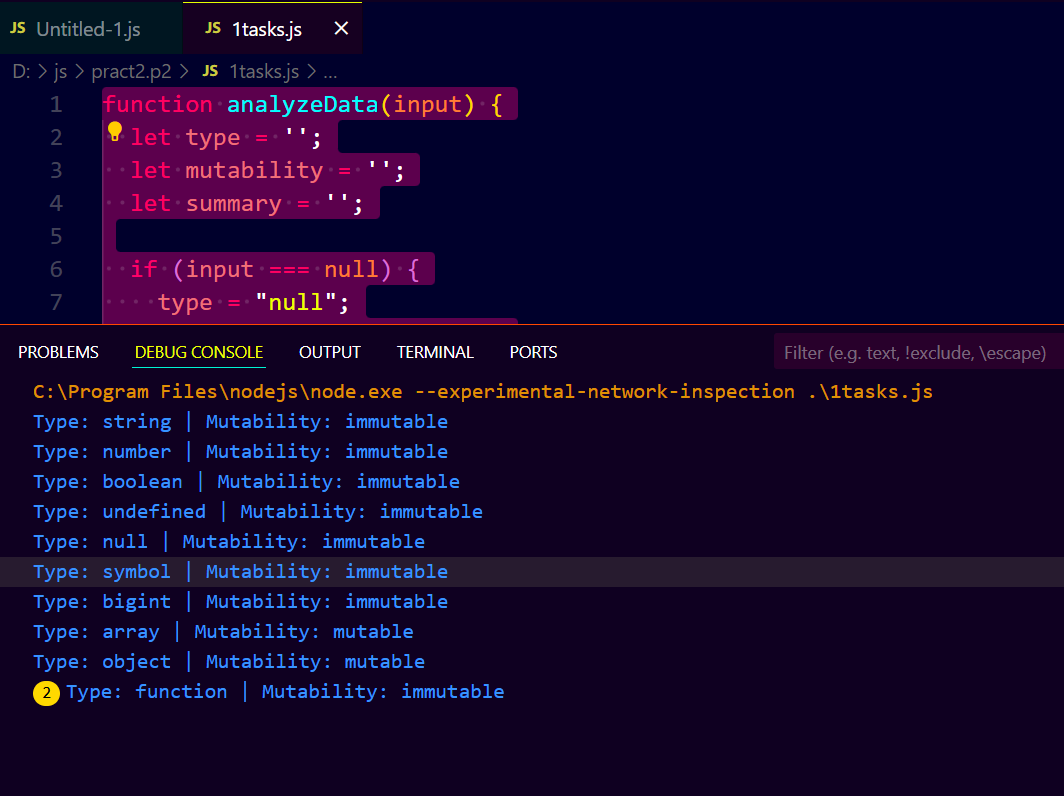
analyzeData(9007199254740991n); // bigint, immutable

analyzeData([1,2,3]); // array, mutable

analyzeData({a:1}); // object, mutable

analyzeData(function(){}); // function, immutable

analyzeData(()=>{}); // function, immutable



**Task 4: Color Mixer using Destructuring and Spread/Rest Operators**

**Problem Statement:**

Write a program that:

* Takes two arrays of colors from the user (e.g., ["red", "green"] and ["blue", "yellow"]).
* Merges the arrays using the spread operator into a new array called palette.
* Extract the first and last colors from the palette using array destructuring.
* Collect the middle colors using the rest operator.
* Display a message like:

"Main colors: Red and Yellow. Others in the palette: Green, Blue."

You are not allowed to use loops or array indexing manually.

