SVU JS Assignment (Day-11)
25/02/2025

Part 1: Objects & Methods

Problem 1: Create a Student Management System (Using Objects)

Task:
Create an object student that has the following properties:

- name (string)
- rollNumber (number)
- marks (object containing subject-wise marks)
- getAverageMarks (method that calculates and returns the average of all subjects)
- checkPassOrFail (method that checks if the student has passed. A student is considered **passed** if their average marks are above **40**.)

Example:

```
const student = {
  name: "Aryan Kumar",
  rollNumber: 101,
  marks: {
    Math: 80,
    Science: 65,
    English: 50,
    History: 70
  },
  getAverageMarks: function() {
    // Calculate and return average marks
  },
  checkPassOrFail: function() {
    // Check if passed or failed
  }
};


console.log(student.getAverageMarks());  // Output: 66.25
```

```
console.log(student.checkPassOrFail());  // Output: "Passed"
```

Problem 2: Library System (Nested Objects & Methods)

Task:
Create an object library that contains a books object, where each book has the following details:

- title (string)
- author (string)
- availableCopies (number)
- borrowBook (method that reduces the availableCopies by 1 if copies are available)
- returnBook (method that increases the availableCopies by 1)

Example:

```
const library = {
  books: {
    "Atomic Habits": { author: "James Clear", availableCopies: 3 },
    "The Alchemist": { author: "Paulo Coelho", availableCopies: 5 },
  },
  borrowBook: function(bookName) {
    // Logic for borrowing a book
  },
  returnBook: function(bookName) {
    // Logic for returning a book
  }
};


library.borrowBook("Atomic Habits");
console.log(library.books["Atomic Habits"].availableCopies);  // Output: 2
```

Part 2: Conditions & Loops

Problem 3: Generate Multiplication Table (For Loop)

Task:
Write a function generateTable(num, limit) that prints the multiplication table of a given num up to limit.

Example:

generateTable(5, 10);

Output:

5 x 1 = 5

5 x 2 = 10

5 x 3 = 15

...

5 x 10 = 50

Problem 4: FizzBuzz (If-Else Conditions)

Task:
Write a function fizzBuzz(n) that prints numbers from **1 to n**, but:

- Print "Fizz" if the number is a multiple of 3
- Print "Buzz" if the number is a multiple of 5
- Print "FizzBuzz" if the number is a multiple of **both** 3 and 5
- Otherwise, print the number itself

Example:

fizzBuzz(15);

Output:

1

2

Fizz

4

Buzz

Fizz

7

8

Fizz

Buzz

11

Fizz

13

14

FizzBuzz

Part 3: String & Array Manipulations

Problem 5: Reverse a String Without Using .reverse()

Task:
Write a function reverseString(str) that takes a string and **reverses** it **without using .reverse()** method.

Example:

console.log(reverseString("JavaScript"));

Output:

tpircSavaJ

Problem 6: Remove Duplicates from an Array

Task:
Write a function removeDuplicates(arr) that removes duplicate elements from an array **without using Set()**.

Example:

console.log(removeDuplicates([1, 2, 3, 2, 4, 5, 1, 6]));

Output:

[1, 2, 3, 4, 5, 6]

Problem 7: Find the Longest Word in a Sentence

Task:
Write a function longestWord(sentence) that finds the **longest word** in a given sentence.

Example:

console.log(longestWord("Coding is amazing and challenging"));

Output:

"challenging"

Part 4: Higher-Level Thinking

Problem 8: Custom Array Method (Creating Your Own .map())

Task:
JavaScript has a built-in .map() method, but can you **create your own** version of it? Write a function myMap(arr, callback) that takes an array and a callback function and applies the callback to each element.

Example:

```
function myCallback(x) {
   return x * 2;
}


console.log(myMap([1, 2, 3, 4], myCallback));
```

Output:

[2, 4, 6, 8]

Problem 9: Find the First Non-Repeating Character in a String

Task:
Write a function firstUniqueCharacter(str) that returns the first non-repeating character in a given string.

Example:

```
console.log(firstUniqueCharacter("aabbcddce"));
```

Output:

"e"

Problem 10: Nested Loop Challenge – Find Pairs that Sum to a Target

Task:
Write a function findPairs(arr, target) that finds all pairs of numbers in an array that sum to a given target.

Example:

```
console.log(findPairs([2, 4, 3, 5, 7, 8, 9], 10));
```

Output:

[[3, 7], [2, 8], [5, 5]]

Bonus Problem (For Extra Challenge)

Problem 11: Implement a Stack in JavaScript
A **stack** follows the **LIFO (Last In, First Out)** principle. Implement a Stack class with the following methods:

- push(value) – Adds a value to the stack
- pop() – Removes and returns the last added value
- peek() – Returns the last added value without removing it
- isEmpty() – Returns true if the stack is empty, false otherwise

Example:

```
let myStack = new Stack();

myStack.push(10);

myStack.push(20);

console.log(myStack.pop()); // 20

console.log(myStack.peek()); // 10

console.log(myStack.isEmpty()); // false
```

Question: - String Manipulation Challenge
You are given a string containing alphabets and numbers. Your task is to extract all numbers from the string, sum them up, and return the new modified string where all numbers are replaced with their sum.
Example:

Input: "abc123xyz45pq7"

Output: "abc175pq"

Question: - Find Most Frequent Element in an Array
Given an array of numbers, find the element that appears the most times. If multiple elements have the same frequency, return any one of them.
Example:

Input: [1, 3, 2, 3, 4, 1, 3, 2, 3, 5]

Output: 3 (since 3 appears the most)