**Q7: Graph Coloring Problem (Greedy)**: Solve the **graph coloring problem** using a **greedy algorithm** to minimize the number of colors needed to color a graph such that no two adjacent nodes share the same color.

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Graph Coloring (Greedy)</title>

<style>

body {

font-family: Arial, sans-serif;

margin: 20px;

text-align: center;

}

#graph-container {

display: flex;

justify-content: center;

margin-top: 20px;

}

.node {

width: 50px;

height: 50px;

border-radius: 50%;

display: flex;

align-items: center;

justify-content: center;

color: white;

font-weight: bold;

margin: 10px;

}

.edge {

position: absolute;

background-color: black;

height: 2px;

}

</style>

</head>

<body>

<h1>Graph Coloring (Greedy Algorithm)</h1>

<button onclick="colorGraph()">Color the Graph</button>

<div id="graph-container"></div>

<script>

// Define the graph as an adjacency list

const graph = {

0: [1, 2],

1: [0, 2, 3],

2: [0, 1],

3: [1]

};

const colors = ["red", "blue", "green", "yellow", "purple"];

// Function to color the graph using the greedy algorithm

function colorGraph() {

const result = {};

for (const node in graph) {

const usedColors = new Set(

graph[node].map(neighbor => result[neighbor])

);

result[node] = colors.find(color => !usedColors.has(color));

}

renderGraph(result);

}

// Function to render the graph with nodes and their colors

function renderGraph(result) {

const container = document.getElementById("graph-container");

container.innerHTML = ""; // Clear previous graph

Object.entries(result).forEach(([node, color]) => {

const nodeElement = document.createElement("div");

nodeElement.className = "node";

nodeElement.style.backgroundColor = color;

nodeElement.textContent = node;

container.appendChild(nodeElement);

});

}

</script>

</body>

</html>