PROGRAM TITLE 11

MAP COLORING CSP

AIM:

To Write the python program for Map Coloring to implement CSP.

PROCEDURE:

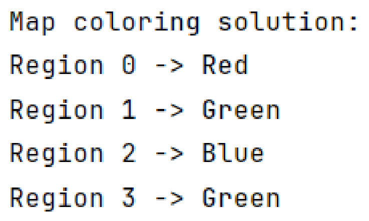
1. Define the Problem: Define the map and the colors available for coloring.
2. Represent the Problem: Represent the map as a graph, where each region is a vertex, and adjacent regions are connected by edges. Also, define the colors available.
3. Implement the Constraints: Implement the is_safe method to check if coloring a region with a specific color violates any constraints.
4. Backtracking Search: Implement the solve method using a backtracking algorithm to find a solution that satisfies all constraints.
5. Solution: Print the solution, if found, showing each region and its corresponding color. Otherwise, indicate that no solution exists.

CODING:

```
class MapColoring:
    def __init__(self, graph, colors):
        self.graph = graph
        self.colors = colors
        self.solution = {}

    def is_safe(self, vertex, color):
        for neighbor in self.graph[vertex]:
            if neighbor in self.solution and self.solution[neighbor] == color:
                return False
        return True
    def solve(self, vertex):
        if vertex not in self.graph:
            return True

        for color in self.colors:
            if self.is_safe(vertex, color):
                self.solution[vertex] = color
                if self.solve(next_vertex(vertex)):
                    return True
                self.solution.pop(vertex)

        return False

def next_vertex(vertex):
    return vertex + 1

if __name__ == "__main__":
    graph = {
        0: [1, 2, 3],
        1: [0, 2],
        2: [0, 1, 3],
        3: [0, 2]
    }
    colors = ['Red', 'Green', 'Blue', 'Yellow']

    map_coloring = MapColoring(graph, colors)
    if map_coloring.solve(0):
        print("Map coloring solution:")
        for region, color in map_coloring.solution.items():
            print(f"Region {region} -> {color}")
    else:
```

```
    print("No solution found.")
```

OUTPUT:



```
Map coloring solution:
Region 0 -> Red
Region 1 -> Green
Region 2 -> Blue
Region 3 -> Green
```

RESULT:

Hence the program been successfully executed and verified.