```python
def is_safe(node, color, assignment, neighbors):
    for neighbor in neighbors[node]:
        if neighbor in assignment and assignment[neighbor] == color:
            return False
    return True

def map_coloring(nodes, colors, neighbors, assignment={}):
    # If all nodes are assigned, print result
    if len(assignment) == len(nodes):
        print("Color Assignment:", assignment)
        return True

    # Select the next node to color
    unassigned = [n for n in nodes if n not in assignment][0]

    # Try each color for the selected node
    for color in colors:
        if is_safe(unassigned, color, assignment, neighbors):
            assignment[unassigned] = color

            # Recurse to assign colors to remaining nodes
            if map_coloring(nodes, colors, neighbors, assignment):
                return True

            # Backtrack if color assignment fails
            del assignment[unassigned]

    return False
nodes = ['WA', 'NT', 'SA', 'Q', 'NSW', 'V', 'T']
neighbors = {
    'WA': ['NT', 'SA'],
    'NT': ['WA', 'SA', 'Q'],
    'SA': ['WA', 'NT', 'Q', 'NSW', 'V'],
    'Q': ['NT', 'SA', 'NSW'],
    'NSW': ['SA', 'Q', 'V'],
    'V': ['SA', 'NSW'],
    'T': []
}

# Define available colors
colors = ['Red', 'Green', 'Blue']

# Run the Map Coloring CSP
if not map_coloring(nodes, colors, neighbors):
    print("No solution found.")
```

```
Color Assignment: {'WA': 'Red', 'NT': 'Green', 'SA': 'Blue', 'Q': 'Red', 'NSW':
'Green', 'V': 'Red', 'T': 'Red'}
```