EXPERIMENT 5

Implement Backtracking algorithms for CSP

AIM:

To implement backtracking algorithm in map coloring problem

ALGORITHM

1. Initialize Data:

- Define the `domain_colors` as a list of available colors.

- Define a list of `variable_states` representing the states that need to be colored.

- Create a `neighbors` dictionary to specify the neighboring relationships between states.

- Create an empty dictionary called `finalstateswithcolor` to store the assigned colors for each state.

2. Color Assignment Functions:

- Define two functions:

- `getthecolor(state)`: This function finds and returns a suitable color for a given state. It iterates through the available colors and calls the `assigncolor` function to check if each color is valid for the state.

- `assigncolor(state, color)`: This function checks if it's valid to assign a specific color to a state. It ensures that none of the neighboring states has the same color.

3. Main Function:

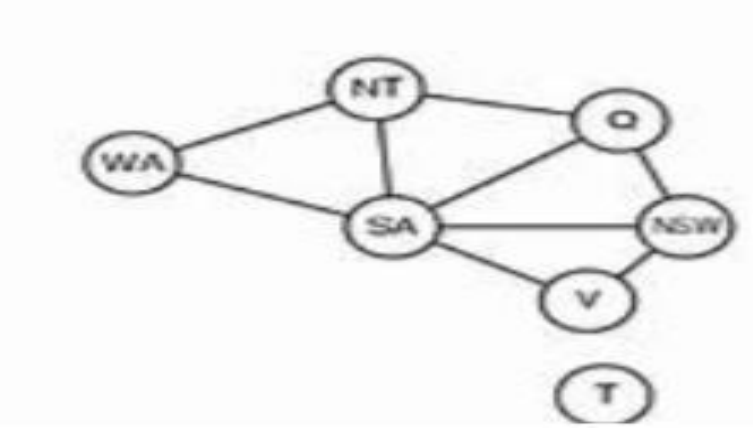
- In the `main` function:

- Use the degree heuristic to sort the states by the number of neighbors they have in descending order. This heuristic prioritizes states with more neighbors to be colored first.

- Iterate through the sorted list of states.

- For each state, call the `getthecolor` function to find a suitable color for the state, and store the assigned color in the `finalstateswithcolor` dictionary.

- Print the dictionary `finalstateswithcolor` to display the states with their assigned colors while ensuring that neighboring states have different colors.



4..PROGRAM

```
domain_colors = ['Red', 'Blue', 'Green']

#wa-west australia,nt-northern territory,sa-south australia,q-queensland,nsw-new south wales,v-victoria,t-tansmania

variable_states = ['wa', 'nt', 'sa', 'q', 'nsw', 'v','t']

 neighbors = {}

neighbors['wa'] = ['nt', 'sa']

neighbors['nt'] = ['wa', 'sa', 'q']

neighbors['sa'] = ['wa', 'nt', 'q', 'nsw', 'v']

neighbors['q'] = ['nt', 'sa', 'snw']

neighbors['nsw'] = ['q', 'sa', 'v']

neighbors['v'] = ['sa', 'nsw']

neighbors['t'] = []


finalstateswithcolor = {}


def getthecolor(state):
    for color in colors:
        if assigncolor(state, color):
            return color


def assigncolor(state, color):
```

```
    for neighbor in neighbors.get(state):
        color_of_neighbor = finalstateswithcolor.get(neighbor)
        if color_of_neighbor == color:
            return False
     return True
 def main():
   #use degree heuristics to find the state with largest number first
    sorted_states = sorted(neighbors.keys(), key=lambda state: len(neighbors[state]),
reverse=True)


    for state in sorted_states:
        finalstateswithcolor[state] = getthecolor(state)
     print("The states with colors are",finalstateswithcolor)
 main()
```

OUTPUT

```
The states with colors are {'sa': 'Red', 'nt': 'Blue', 'q': 'Green', 'nsw': 'Blue', 'wa': 'Green', 'v': 'Green', 't': 'Red'}
```

RESULT:

The backtracking algorithm in map coloring problem is implemented and the output is verified.