**Name – Armaan Asif Shaikh**

**PRN- 22311412**

**Roll No - 382041**

**AI ASSIGNMENTS**

**Assignment -2**

#include <bits/stdc++.h>

using namespace std;

vector<string> regions = {"WA", "NT", "SA", "Q", "NSW", "V", "T"};

vector<string> colors = {"Red", "Green", "Blue"};

map<string, vector<string>> 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", {}} // Tasmania isolated

};

map<string, string> assignment;

map<string, vector<string>> domains;

void initDomains() {

for (auto &r : regions) domains[r] = colors;

}

bool isConsistent(string var, string color) {

for (auto &n : neighbors[var]) {

if (assignment.find(n) != assignment.end() && assignment[n] == color)

return false; // conflict

}

return true;

}

bool forwardCheck(string var, string color, map<string, vector<string>> &localDomains) {

for (auto &n : neighbors[var]) {

if (assignment.find(n) == assignment.end()) { // not assigned yet

auto &nd = localDomains[n];

nd.erase(remove(nd.begin(), nd.end(), color), nd.end());

if (nd.empty()) return false; // domain wiped out

}

}

return true;

}

bool backtrack() {

if (assignment.size() == regions.size()) return true;

string var;

for (auto &r : regions) {

if (assignment.find(r) == assignment.end()) {

var = r;

break;

}

}

for (auto &color : domains[var]) {

if (isConsistent(var, color)) {

assignment[var] = color;

auto savedDomains = domains;

if (forwardCheck(var, color, domains)) {

if (backtrack()) return true;

}

domains = savedDomains;

assignment.erase(var);

}

}

return false;

}

int main() {

initDomains();

if (backtrack()) {

cout << "Solution Found:\n";

for (auto &r : regions) {

cout << r << " = " << assignment[r] << "\n";

}

} else {

cout << "No solution exists.\n";

}

return 0;

}

