```cpp
#include <stdio.h>
#include <iostream>
#include <fstream>
#include <string>
#include <vector>

using namespace std;

const int COLORES = 3;

typedef struct {
    int r; //Azul representa 0
    int a; //Rojo representa 1
    int v; //Verde representa 2
} tColores;

bool esValida(vector<int> sol, int k, vector<int>& usd, int dis[], int color) {

    bool valida = true;

    if (usd[color] > dis[color]) {
        usd[color]--;
        valida = false;
    }
    else if (usd[2] > usd[0]) {
        usd[color]--;
        valida = false;
    }
    else if (sol[k] == 2 && sol[k - 1] == sol[k]) {
        usd[color]--;
        valida = false;
    }

    return valida;
}

bool esSolucion(int k, int m) {
    return k == (m - 1);
}

void tratarSolucion(vector<int> sol, int m, bool& exito, vector<int>& usd, int color) {

    if (usd[0] + usd[2] < usd[1]) {
        string color;

        for (int i = 0; i < m; i++) {

            if (sol[i] == 0) color = "azul";
            else if (sol[i] == 1) color = "rojo";
            else if (sol[i] == 2) color = "verde";

            cout << color << " ";
        }
```

```
        exito = true;

        cout << endl;
    }

    usd[color]--;
}

void torreColor(vector<int> sol, int k, int n, int m, int dis[], vector<int>& usd, bool&
exito) {
    for (int color = 0; color < n; color++) {
        sol[k] = color;
        usd[color]++;
        if (esValida(sol, k, usd, dis, color)) {
            if (esSolucion(k, m)) {
                tratarSolucion(sol, m, exito, usd, color);
            }
            else {
                torreColor(sol, k + 1, n, m, dis, usd, exito);
                usd[color]--;
            }
        }
    }
}

bool resuelveCaso() {

    // Lectura de los datos
    int m;

    cin >> m;

    if (m == 0) return false;

    int disponibles[COLORES];

    vector<int> usados(COLORES);

    usados[0] = 0;
    usados[1] = 1;
    usados[2] = 0;

    cin >> disponibles[0] >> disponibles[1] >> disponibles[2];

    vector<int> sol(m);

    sol[0] = 1;

    bool exito = false;

    torreColor(sol, 1, COLORES, m, disponibles, usados, exito);

    if (!exito) cout << "SIN SOLUCION" << endl;
```

```
    cout << endl;

    return true;
}


int main() {

    // Para la entrada por fichero.
#ifndef DOMJUDGE
    std::ifstream in("casos.txt");
    auto cinbuf = std::cin.rdbuf(in.rdbuf());
#endif

    // Resolvemos
    while (resuelveCaso()) {}

#ifndef DOMJUDGE // para dejar todo como estaba al principio
    std::cin.rdbuf(cinbuf);
    system("PAUSE");
#endif

    return 0;
}
```