Code

```c
#include <stdio.h>
#include <stdbool.h>

#define V 5

bool isSafe(int v, int graph[V][V], int path[], int pos) {
    if (graph[path[pos - 1]][v] == 0)
        return false;
    for (int i = 0; i < pos; i++) {
        if (path[i] == v)
            return false;
    }
    return true;
}

bool hamCycleUtil(int graph[V][V], int path[], int pos) {
    if (pos == V) {
        if (graph[path[pos - 1]][path[0]] == 1)
            return true;
        return false;
    }

    for (int v = 1; v < V; v++) {
        if (isSafe(v, graph, path, pos)) {
            path[pos] = v;
            if (hamCycleUtil(graph, path, pos + 1))
                return true;
            path[pos] = -1;
        }
    }
    return false;
}

bool hamCycle(int graph[V][V]) {
    int path[V];
    for (int i = 0; i < V; i++) {
        path[i] = -1;
    }
    path[0] = 0;

    if (hamCycleUtil(graph, path, 1) == false) {
        printf("Solution does not exist\n");
        return false;
    }

    printf("Solution Exists: Following is the Hamiltonian Cycle\n");
    for (int i = 0; i < V; i++) {
        printf("%d ", path[i]);
    }
    printf("%d\n", path[0]);
    return true;
}

int main() {
    int graph[V][V] = {
        {0, 1, 0, 1, 0},
        {1, 0, 1, 1, 0},
        {0, 1, 0, 1, 1},
```

```
        {1, 1, 1, 0, 1},
        {0, 0, 1, 1, 0}
    };

    hamCycle(graph);
    return 0;
}
```

Output

```
Solution Exists: Following is the Hamiltonian Cycle
0 1 2 4 3 0
```

Code

```
#include <stdio.h>
#define V 4

int isSafe(int v, int graph[V][V], int color[], int c);
int graphColorUtil(int graph[V][V], int m, int color[], int v);
int graphColoring(int graph[V][V], int m);
void display(int color[]);

int main(){
    int graph[V][V] = {{0, 1, 1, 1},
                       {1, 0, 1, 0},
                       {1, 1, 0, 1},
                       {1, 0, 1, 0}};

    int m = 3;
    graphColoring(graph, m);
    return 0;
}

int isSafe(int v, int graph[V][V], int color[], int c){
    for(int i=0; i<V; i++){
        if(graph[v][i] && c == color[i]) return 0;
    }
    return 1;
}

int graphColorUtil(int graph[V][V], int m, int color[], int v){
    if(v == V) return 1;

    for(int c=1; c<=m; c++){
        if(isSafe(v, graph, color, c)){
            color[v] = c;
            if(graphColorUtil(graph, m, color, v+1)){
                return 1;
            }
            color[v] = 0;
        }
    }
    return 0;
}

int graphColoring(int graph[V][V], int m){
    int color[V];
    for(int i=0; i<V; i++){
        color[i] = 0;
    }

    if(!graphColorUtil(graph, m, color, 0)){
        printf("Solution does not exist");
        return 0;
    }
    display(color);
    return 1;
}

void display(int color[]){
    printf("Solution: \n");
    for(int i=0; i<V; i++){
```

```
        printf(" %d ", color[i]);
    }
    printf("\n");
}
```

Output

```
Solution:
 1  2  3  2
```