**Graph algorithms - practical work no. 5**

Andreas R. Toadere **C++** | Group 917

Problem 3. Given an undirected graph, find a vertex coloring with minimum number of colors.

**Solution A:**

Graph coloring is widely used. Unfortunately, there is no efficient algorithm available for coloring a graph with minimum number of colors as the problem is a known [NP Complete problem](http://www.geeksforgeeks.org/np-completeness-set-1/). There are approximate algorithms to solve the problem though. Following is the basic Greedy Algorithm to assign colors. It doesn’t guarantee to use minimum colors, but it guarantees an upper bound on the number of colors. The basic algorithm never uses more than d+1 colors where d is the maximum degree of a vertex in the given graph.

**Time Complexity: O(V^2 + E)**

**Analysis of Basic Algorithm**

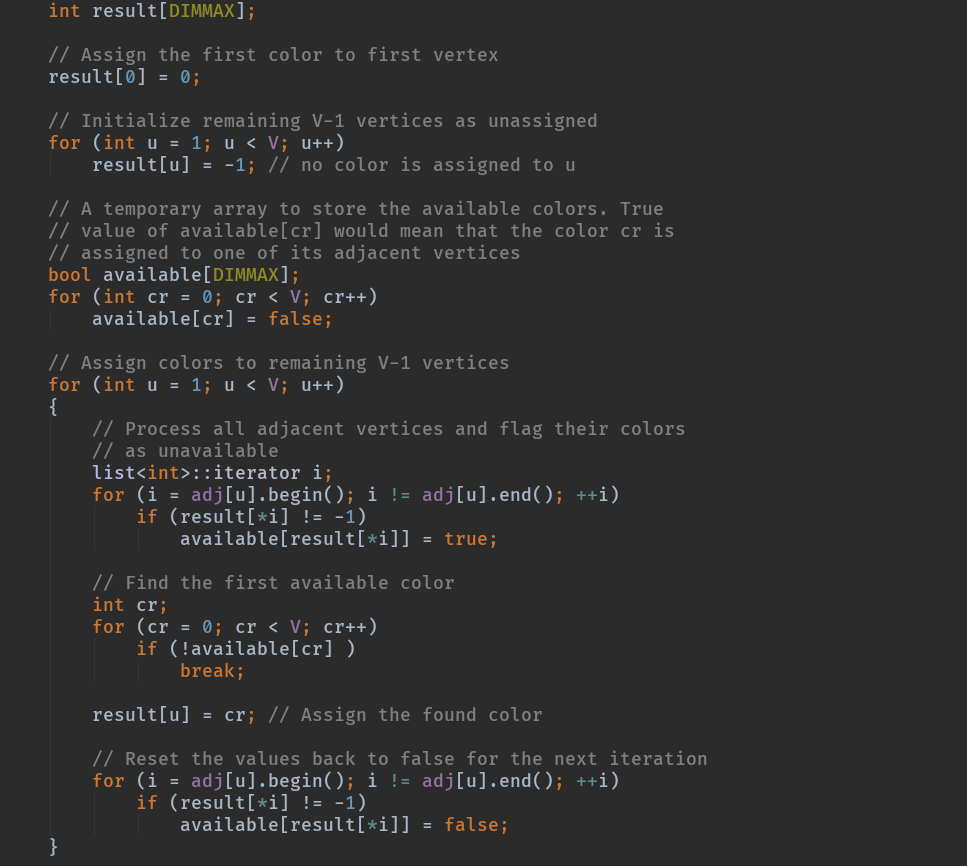
The algorithm doesn’t always use minimum number of colors. Also, the number of colors used sometime depend on the order in which vertices are processed. For example, consider the following two graphs. Note that in graph on right side, vertices 3 and 4 are swapped. If we consider the vertices 0, 1, 2, 3, 4 in left graph, we can color the graph using 3 colors. But if we consider the vertices 0, 1, 2, 3, 4 in right graph, we need 4 colors.



So the order in which the vertices are picked is important. Many people have suggested different ways to find an ordering that work better than the basic algorithm on average. The most common is [**Welsh–Powell Algorithm**](http://mrsleblancsmath.pbworks.com/w/file/fetch/46119304/vertex%20coloring%20algorithm.pdf) which considers vertices in descending order of degrees.

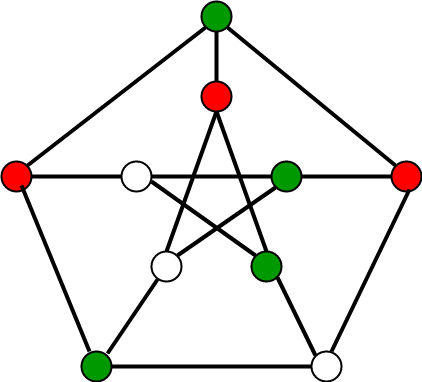
**How does the basic algorithm guarantee an upper bound of d+1?**

Here d is the maximum degree in the given graph. Since d is maximum degree, a vertex cannot be attached to more than d vertices. When we color a vertex, at most d colors could have already been used by its adjacent. To color this vertex, we need to pick the smallest numbered color that is not used by the adjacent vertices. If colors are numbered like 1, 2, …., then the value of such smallest number must be between 1 to d+1 (Note that d numbers are already picked by adjacent vertices).  
This can also be proved using induction.



**Solution B:**

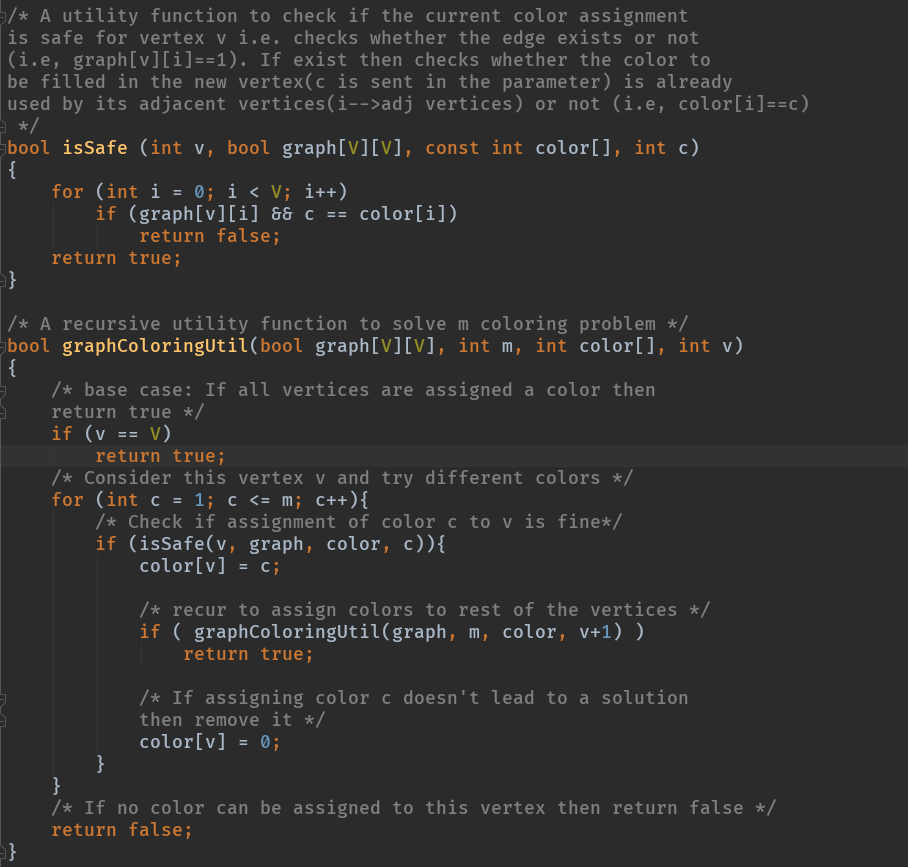
The second solution uses backtracking. Following is an example of graph that can be colored with 3 different colors.

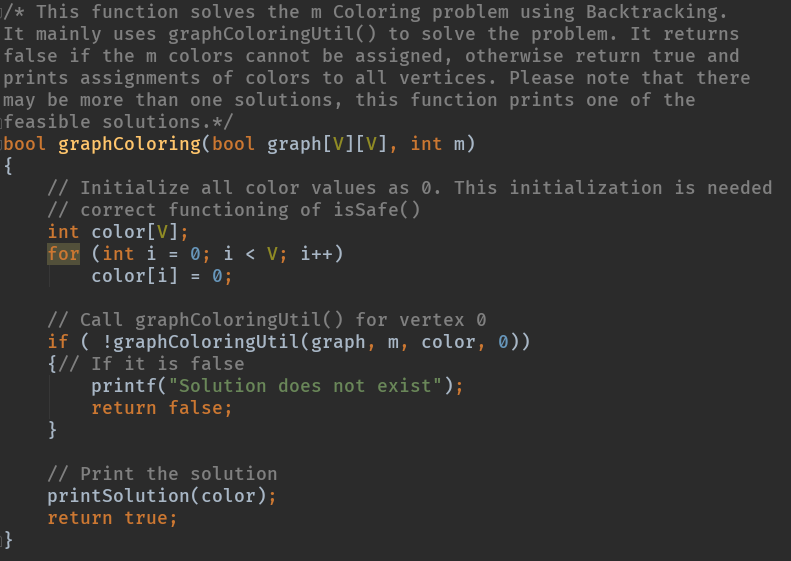


**Naive Algorithm**  
Generate all possible configurations of colors and print a configuration that satisfies the given constraints.

**Time Complexity: O(V^m + E)**

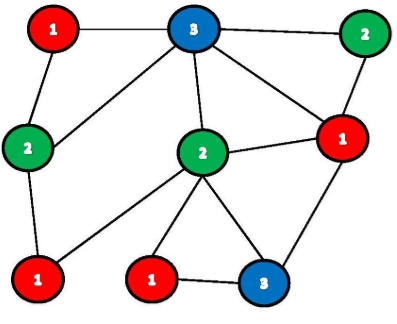
**Backtracking Algorithm**  
The idea is to assign colors one by one to different vertices, starting from the vertex 0. Before assigning a color, we check for safety by considering already assigned colors to the adjacent vertices. If we find a color assignment which is safe, we mark the color assignment as part of solution. If we do not a find color due to clashes then we backtrack and return false.



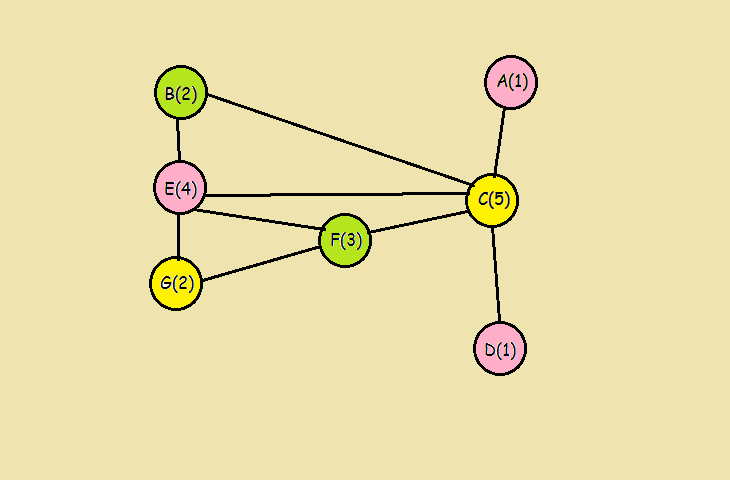
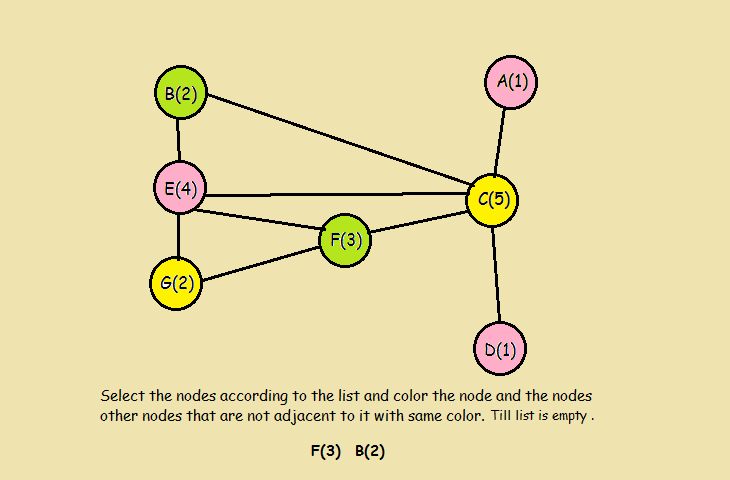
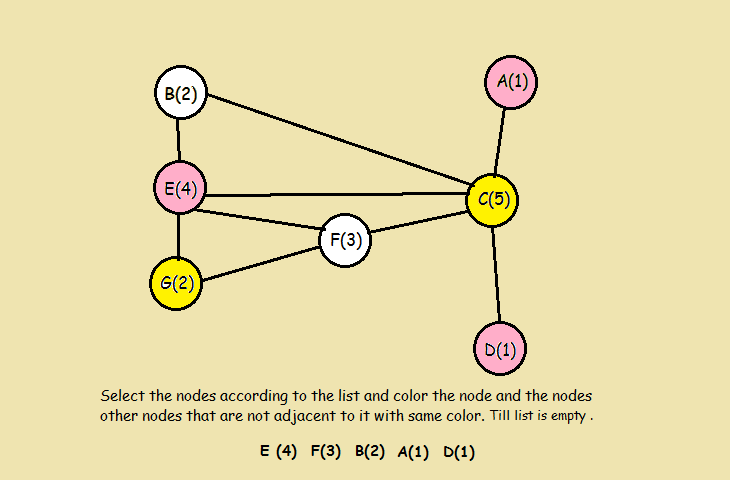
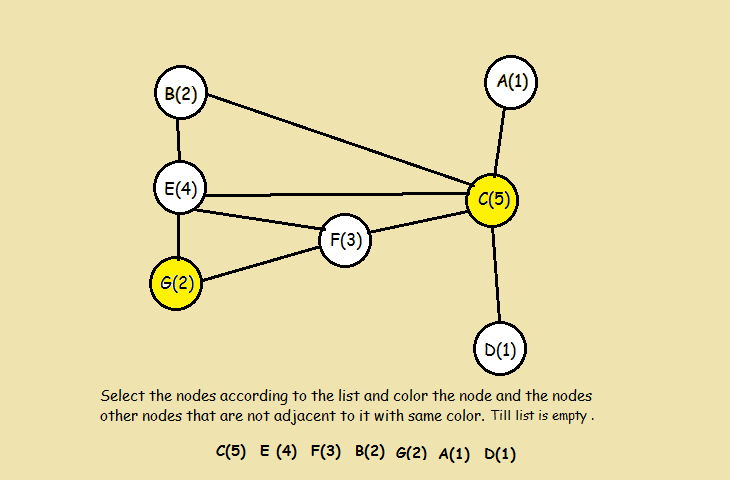
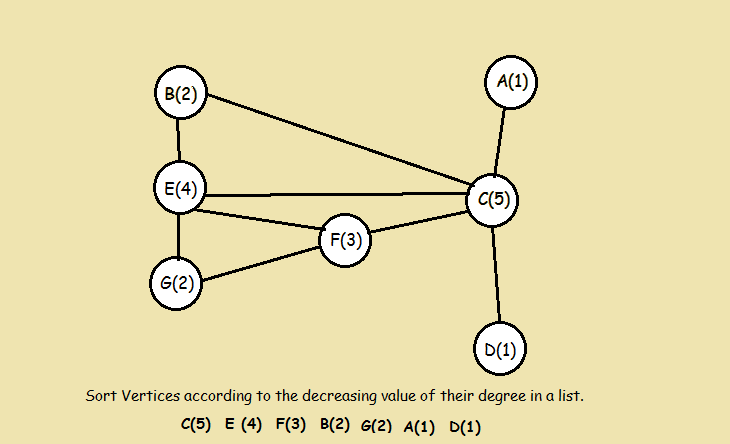
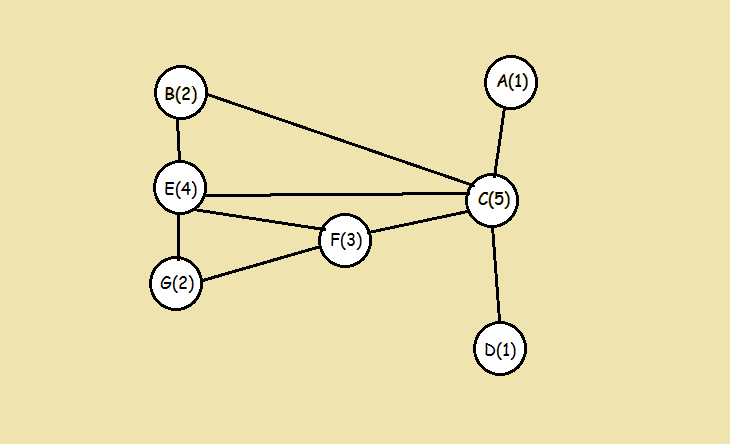
****

**Solution C:**

In graph theory, Welsh Powell is used to implement graph labeling; it is an assignment of labels traditionally called "colors" to elements of a graph subject to certain constraints.  
 In 1967 Welsh and Powell introduced in an upper bound to the chromatic number of a graph . It provides a greedy algorithm that runs on a static graph.  
The vertices are ordered according to their degrees, the resulting greedy coloring uses at most maximind(xi)+1,imaximind(xi)+1,i colors, at most one more than the graph’s maximum degree. This heuristic is called the **Welsh–Powell algorithm**.



**ALGORITHM IMPLEMENTATION USING IMAGES**



**ALGORITHM IMPLEMENTATION IN C++**

