**Q5.2) Create a simple greedy algorithm for coloring the vertices of any graph G, ideally using as few colors as possible. Explain how your algorithm works, i.e., the order in which your algorithm chooses the vertices of a given graph, and how a color is assigned to each vertex.**

Graph Coloring is a process of assigning colors to the vertices of a graph. It ensures that no two adjacent vertices of the graph are colored with the same color.

The chromatic number of the graph is the minimum number of colors needed to produce a proper coloring of the graph. A ***proper coloring*** is an assignment of colors to the vertices of a graph so that no two adjacent vertices have the same color. A ***k-coloring*** of a graph is a proper coloring involving a total of k colors. A graph that has a k-coloring is said to be ***k-colorable***.

So, what we seek is a ***k-coloring*** of our graph with k as small as possible.
Now, we know that a vertex is assigned one of the k colors and every pair u,v of adjacent vertices, u and v are assigned different colors.
A simple greedy algorithm (procedure) for properly coloring vertices with as few colors as possible is defined as below.

**Algorithm/Psuedocode**

```
function GreedyColoring(Graph G(V,E)):
    for all v ∈ V (G) do
        color(v) ← −1
    end for
    for all v ∈ V (G) in order do
        isCol(1 . . . Δ(G) + 1) ← false
        for all u ∈ N(v) where color(u) ≠ −1 do
            isCol(color(u)) ← true
        end for
        for k = 1 . . . Δ(G) + 1 do
            if isCol(k) = false then
                color(v) ← k
                break
```

```
                end if
            end for
        end for
        k ← max(color(1 . . . n))
        return k
end function
```

**Explanation - Working of Algorithm**

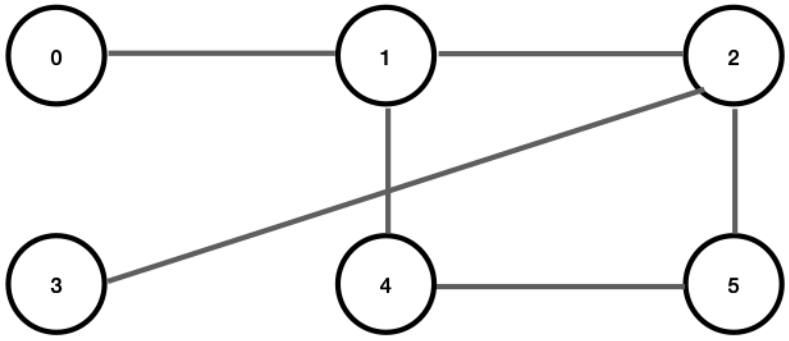
A simple greedy algorithm for creating a proper coloring is shown below. The basic idea is do a single pass through all vertices of the graph in some order and label each one with a numeric identifier. The procedure requires us to number consecutively the colors that we use, so each time we introduce a new color, we number it also. A vertex will labeled/colored with the lowest value that doesn't appear among previously colored neighbors.

1. Color the vertex with color 1.
2. Pick an uncolored vertex v. Color it with the lowest-numbered color that has not been used on any previously-colored vertices adjacent to v. If all previously-used colors appear on vertices adjacent to v, this means that we must introduce a new color and number it.
3. Repeat the step-2 until all vertices are colored.

**Runtime Analysis of the Algorithm**

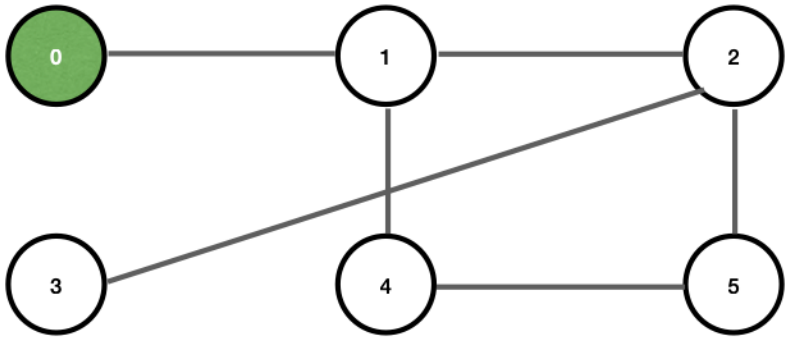
The time complexity of the above algorithm is O(VxE) where V is the total number of vertices and E is the total number of edges in any graph G.

**Example - the order in which the vertices are chosen using Greedy Algorithm**

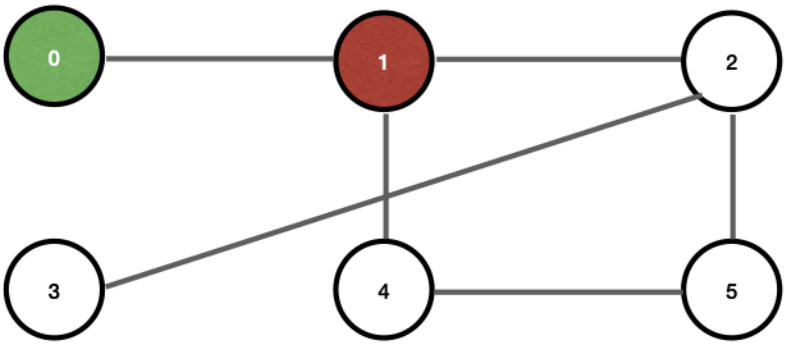


This is the example whose source code is provided in the separate file which uses greedy algorithm to color the graph in k-colors.

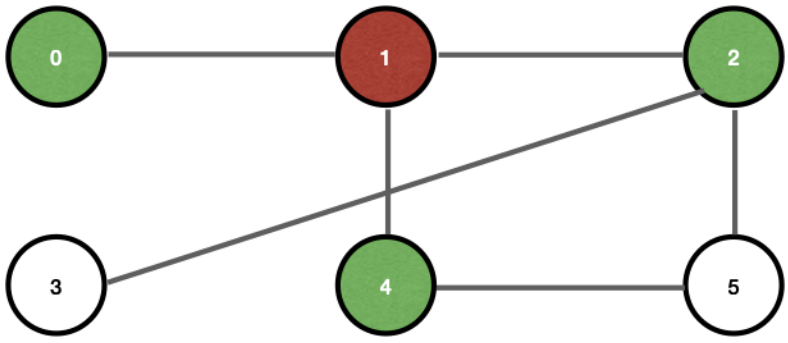
So, first vertex which vertex 0 will be colored using the first available color which is color 1 (green).



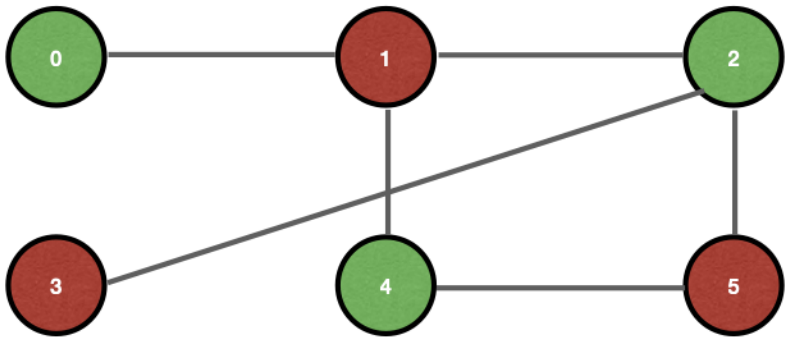

Next, adjacent of vertex 0 is vertex 1 so it will assign a different color which is color 2 (red).




Next, we check for adjacent vertices of vertex 1 which vertex 2 and 4, so it will assign color 1 (green) as both the vertices do not share border with the vertex which has already been assigned green color.



Next, we check for adjacent vertices of vertex 2 which vertex 3 and 5, so it will assign color 2 (red) as both the vertices do not share border with the vertex which has already been assigned red color.




As, we can see the graph is colored with 2 colors and adjacent vertices do not share same color. So, the chromatic number of the graph is 2.