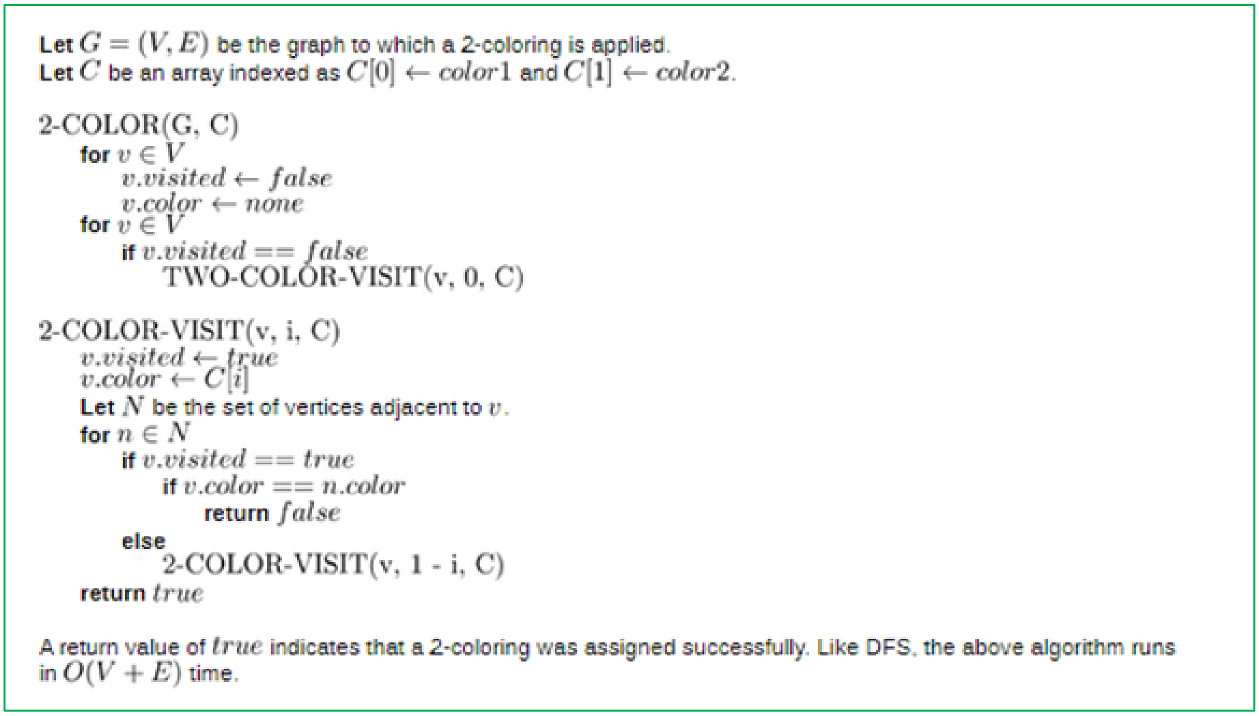
1. Graph-Coloring. Mapmakers try to use as few colors as possible when coloring countries on a map, as long as no two countries that share a border have the same color. We can model this problem with an undirected graph G = (V,E) in which each vertex represents a country and vertices whose respective countries share a border are adjacent. A k-coloring is a function c: V -> {1, 2, … , k} such that c(u) ≠ c(v) for every edge (u,v) ∈ E. In other words the number 1, 2, .., k represent the k colors and adjacent vertices must have different colors. The graph-coloring problem is to determine the minimum number of colors needed to color a given graph.

a. Give an efficient algorithm to determine a 2-coloring of a graph, if one exists.
Several correct solutions



Let $G = (V, E)$ be the graph to which a 2-coloring is applied.
Let C be an array indexed as $C[0] \leftarrow color1$ and $C[1] \leftarrow color2$.

```
2-COLOR(G, C)
    for $v \in V$
        $v.visited \leftarrow false$
        $v.color \leftarrow none$
    for $v \in V$
        if $v.visited == false$
            TWO-COLOR-VISIT(v, 0, C)

2-COLOR-VISIT(v, i, C)
    $v.visited \leftarrow true$
    $v.color \leftarrow C[i]$
    Let $N$ be the set of vertices adjacent to $v$.
    for $n \in N$
        if $v.visited == true$
            if $v.color == n.color$
                return $false$
        else
            2-COLOR-VISIT(v, 1 - i, C)
    return $true$
```

A return value of $true$ indicates that a 2-coloring was assigned successfully. Like DFS, the above algorithm runs in $O(V + E)$ time.

b. State the graph-coloring problem as a decision problem K-COLOR.

Decision Problem: **K-COLOR** : Given a graph G= (V,E) is there a way to color the vertices with exactly k colors such that adjacent vertices are colored differently?

c. Show that your decision problem is solvable in polynomial time if and only of the graph-coloring problem is solvable in polynomial time.

If we can solve K-COLOR for a given graph G in polynomial time, then the graph-coloring problem can be solved in polynomial time by assigning K from 1 to |V |and checking if the answer to K-COLOR is YES. As soon as we get the answer YES for K-COLOR we can stop since K is the minimum of any coloring. K is the solution to the graph-coloring that asks for the minimum number of colors. At most we have to try |V| colors. With K-=|V| you can color any graph G=(V,E) by assigning each vertex a different color.

If we can solve the graph-coloring problem for a given graph G=(V,E) in polynomial time then we can solve K-COLOR in polynomial time. Solve the graph-coloring problem and let K* be the minimum colors needed. The answer to K-COLOR is no if K < K* or K >|V|. The answer to K-COLOR is yes if K ≥ K*.

c. Use the fact that 3-COLOR is NP-complete to show that 4-COLOR is NP-complete

Step 1: **Show that 4-COLOR is in NP. Give a polynomial time algorithm to verify solution. This is the same as in part c) above but with k=4. Must Show polynomial to receive full credit**

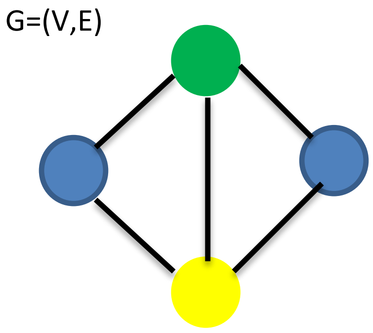
Given a Graph G=(V,E) and a 4-coloring function c: V -> {1, 2, 3, 4} we can verify if c is a "legal" coloring function in polynomial time. To verify the solution, for each vertex u in V we must check the colors of the adjacent vertices. All colors of adjacent vertices must be different. If for any (u, w) ∈ E, c(u) = c(w) then c is not a 4-COLORING of G.. The verification of the 4-coloring is polynomial in n (the number of vertices) since 4 ≤ n and the time required to look at all edges in G is O(n^2).

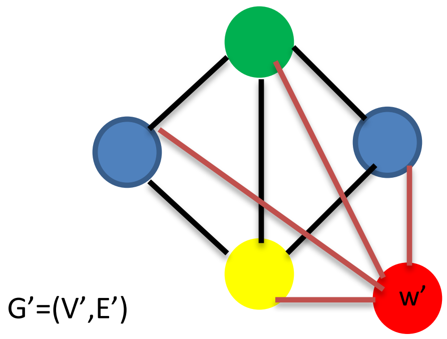
Step 2: Show that there is a polynomial reduction from 3-COLOR to 4-COLOR.

Reduce an instance G of 3-COLOR to an instance G' of 4-COLOR in polynomial time, and show that there is a 3-COLOR in G iff there is a 4-COLOR in G'. Let G=(V,E) be an instance of 3-COLOR transform G into G' by adding a new vertex w' that is connect t every other vertex. That is

G'=(V', E') where V'= V ∪{w'} and E' = E ∪ { (w', u) for all u ∈ G}

This reduction can be done in polynomial time since we are adding one vertex and at most n edges





blue = 1, yellow = 2, green = 3, red = 4.

If G has a 3-COLORing then G' has a 4-COLORing. Assume G has a 3-COLORing then there exists a function c: V -> {1, 2, 3} such that for all u, w ∈ V if (u,w) ∈E then c(u) ≠c(w). Now define the 4-coloring function c' for G'

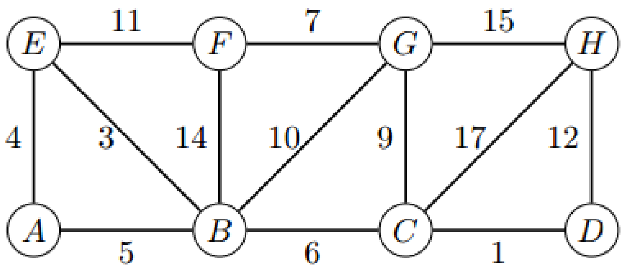
$$c'(u) = \begin{cases} c(u), & if\ u\ \in V \\ \quad 4, & if\ u\ \notin\ V\ \ (u = w') \end{cases}$$

Therefore, if there is a 3-COLORing in G then there is a 4-COLORing in G'

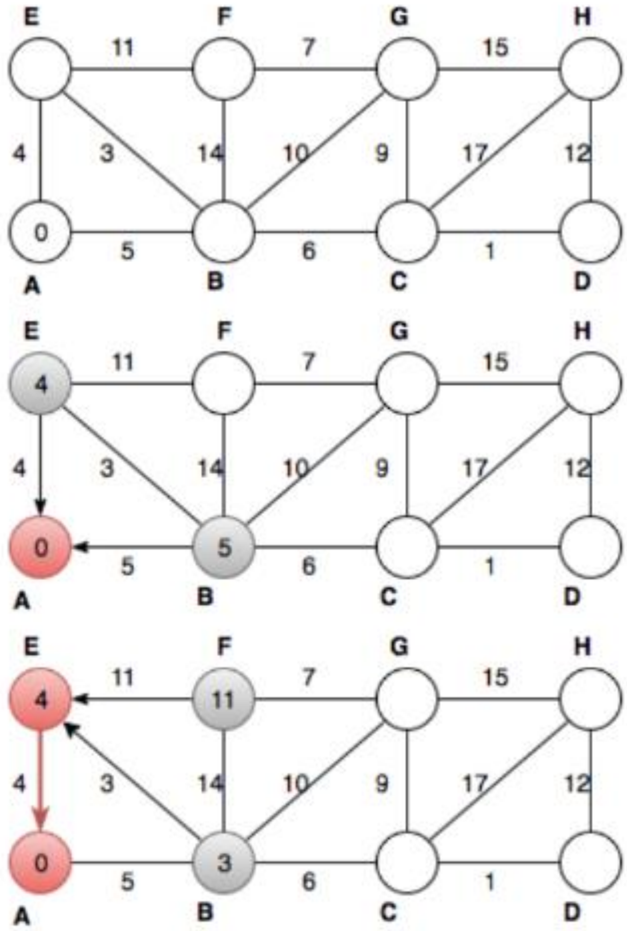
If G' has a 4-COLORing then G has a 3-COLORing. Assume G' has a 4-COLORing, since w' is adjacent to all other vertices in G' then w' must be a different color. Let c' be the coloring function for G', without loss of generality we can say that c'(w')= 4 and c(u) ≠ 4 for all u ∈(V' –{w'}). However, (V' –{w'}) = V. So we have colored all of the original vertices in V using only colors 1, 2 and 3 proving that G is 3-COLORable. Thus, the 4-Color problem is NP-Hard

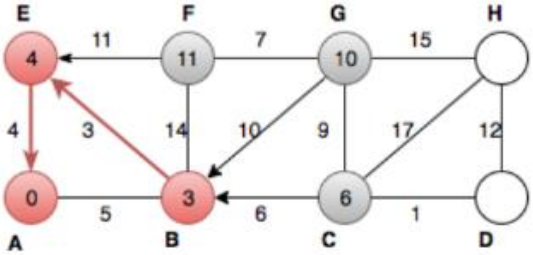
Since it was shown in Part 1 that 4-COLOR is in NP, and by Step 2 NP-Hard, 4-COLOR is NP-Complete. This statement must be included.

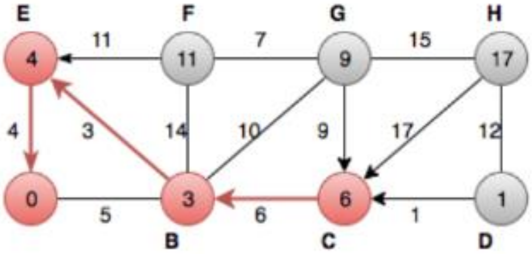
2. Consider the weighted graph below:

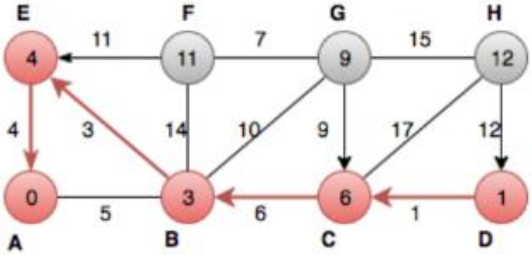


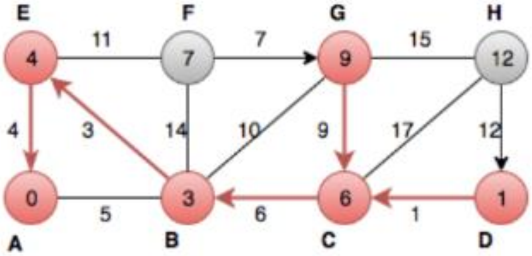
(a) Demonstrate Prim's algorithm starting from vertex A.

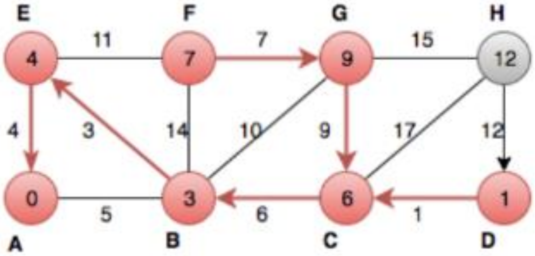


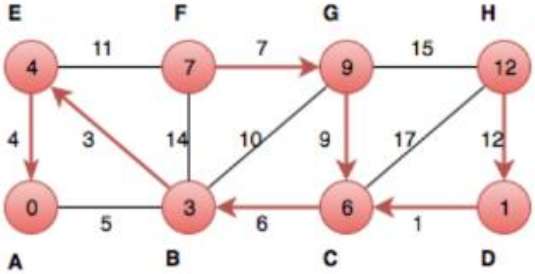

E
F
G
H
11
7
15
4
11
10
4
3
14
10
9
17
12
0
3
6
5
6
1
A
B
C
D


E
F
G
H
11
7
15
4
11
9
17
4
3
14
10
9
17
12
0
3
6
1
5
6
1
B
C
D

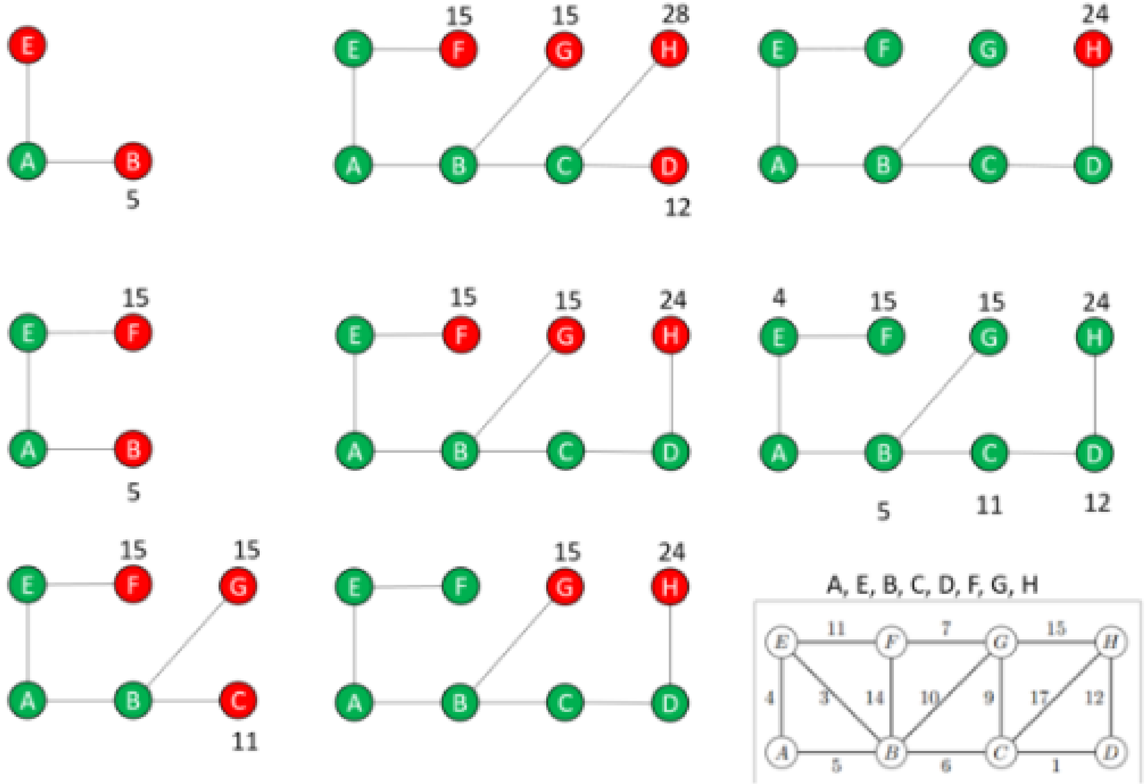

E
F
G
H
11
7
15
4
11
9
12
4
3
14
10
9
17
12
0
3
6
1
5
6
1
A
B
C
D


E
F
G
H
11
7
15
4
7
9
12
4
3
14
10
9
17
12
0
3
6
1
5
6
1
A
B
C
D


E
F
G
H
11
7
15
4
7
9
12
4
3
14
10
9
17
12
0
3
6
1
5
6
1
A
B
C
D


E
F
G
H
11
7
15
4
7
9
12
4
3
14
10
9
17
12
0
3
6
1
5
6
1
A
B
C
D

(b) Demonstrate Dijkstra’s algorithm on the graph, using vertex A as the source.



3. Suppose you have four production plants for making cars. Each works a little differently in terms of labor needed, materials, and energy used per car:

Location	labor hrs	materials	energy
plant1	2	3	15
plant2	3	4	10
plant3	4	5	9
plant4	5	6	7

Suppose we need to produce at least 400 cars at plant 3 according to a labor agreement. We have 3300 hours of labor and 4000 units of material available. We are allowed to use at most 12000 units of energy, and we want to maximize the number of cars produced. Formulate this as a linear programming problem: (1) what are the variables, (2) what is the objective in terms of these variables, and (3) what are the constraints.

Solution:

1. What are the variables? $x1,x2,x3,x4$, where xi denotes the number of cars at plant i.

2. What is our objective? maximize $x1 + x2 + x3 + x4$.

3. What are the constraints?

$$xi \geq 0 \text{ (for all i)}$$

$$x3 \geq 400$$

$$2x1 + 3x2 + 4x3 + 5x4 \leq 3300$$

$$3x1 + 4x2 + 5x3 + 6x4 \leq 4000$$

$$15x1 + 10x2 + 9x3 + 7x4 \leq 12000$$