Chris Grimes
CS 46101
HW 6

1. (10 points) A company named RT&T has a network of n switching stations connected by m high-speed communication links. Each customer's phone is directly connected to one station in his or her area. The engineers of RT&T have developed a prototype video-phone system that allows two customers to see eachother during a phone call. In order to have acceptable image quality, however, the number of links used to transmit video signals between the two parties cannot exceed 4. Suppose that RT&T network is represented by a graph. Design and give the pseudo-code for an efficient algorithm that computes, for each station, the set of stations it can reach using no more than 4 links. Analyze its running time.

```
Algorithm span4Video(a vertex A, distance from source):

    input: a vertex A and an int I which is the distance from the source of the video

    // I should begin at 0 the first time this method is called

    output: a set of vertices within 3 edges of vertex A

    set rtnval=null

    for each(edge in the graph that contains vertex A)//runs O(n+m)

        edge.label()=-1

    ++I

    if(I <5)

        for each(incident edge of A)// runs a max of O(n+m)

            if(edge.label()==-1)

                edge.label()=I

                rtnval.add(other vertex connected to edge)

                span4video(other vertex connected to edge, I)

    return rtnval
```

Assuming that links means edges the above algorithm is correct, if links means vertices the "if( I<5)" should be "if(I<4)"

Do to the two for loops in the above algorithm, the span4video algorithm will have a worst case running time of O(n+m).

Chris Grimes
CS 46101
HW 6

2. (10 points) An Eulerian circuit of a directed graph G with n vertices and m edges is a cycle that traverses each edge of G exactly once according to its direction. Such a cycle always exists if the in-degree is equal to the out-degree for each vertex in G. Describe a O(n + m) time algorithm for finding an Eulerian circuit of such a graph G. Analyze its running time.

```
Algorithm eulerianCircuit(a graph G):

    input: a graph G

    output: a list containing the vertices of a eulerian circuit in G

    an empty list rtnval

    for each(edge in graph g)

        edge.color()=red

    v= an edge contained in graph G from which we'll start our search

    rtnval.add(v)

    while(there is a red incident edge of V)

        edge.color()=white

        rtnval.add(other vertex connected to edge)

        v= other vertex connected to edge

    return rtnval
```

Do to the for loop and the while loop in the above algorithm, the eulerianCircuit algorithm will have a worst case running time of O(n+m).

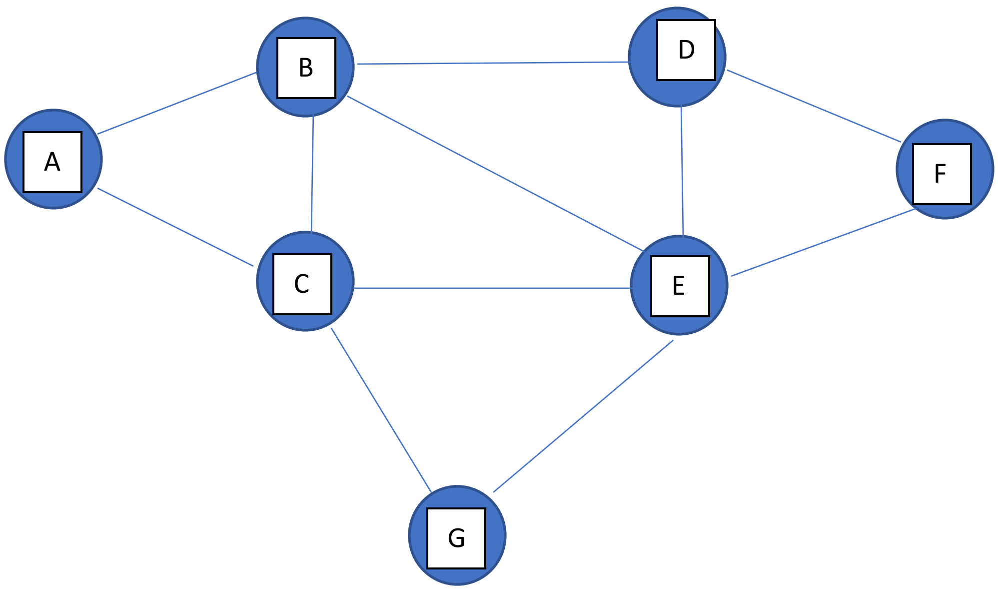
Chris Grimes
CS 46101
HW 6

3. (10 points) Consider the graph G depicted below, and perform the following graph search algorithms. Whenever faced with a decision of which vertex to pick from a set of vertices, pick the vertex whose label occurs earliest in the alphabet.

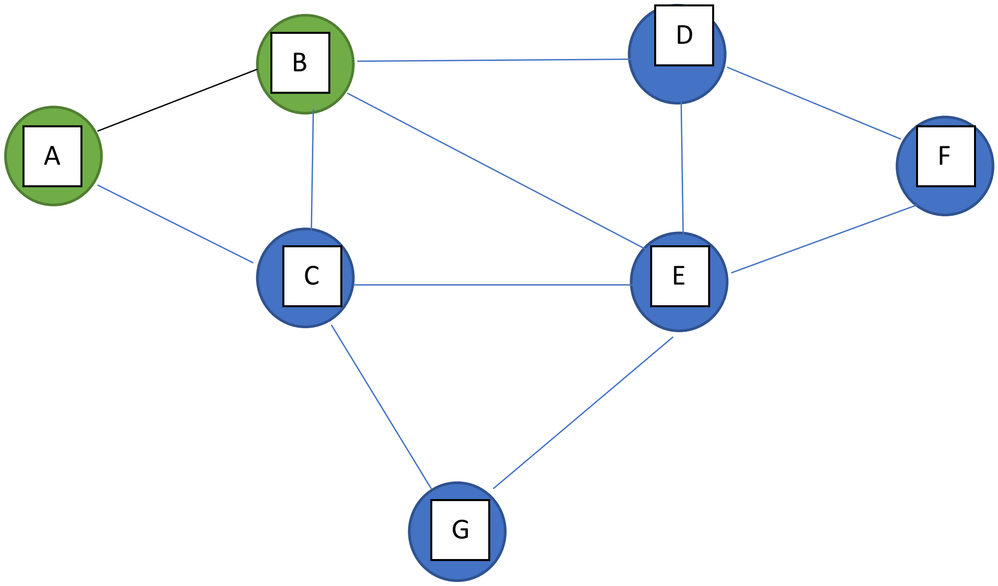
(a) Trace the execution of BFS beginning at vertex A, labeling each edge as a discovery or cross edge.

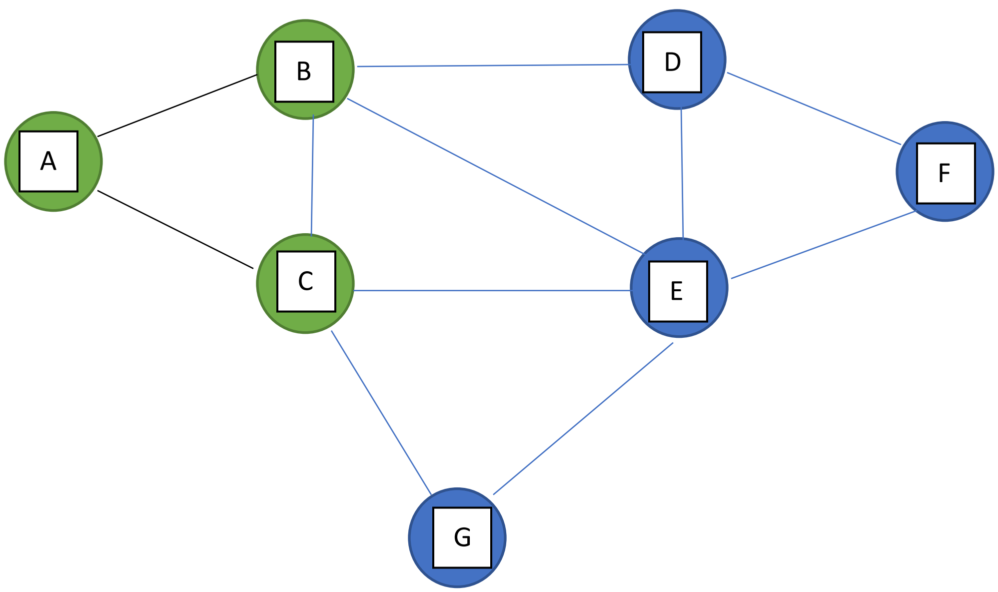
Blue circle= unexplored vertex blue line= unexplored edge

green circle= visited vertex black line= discovery edge red line= cross edge

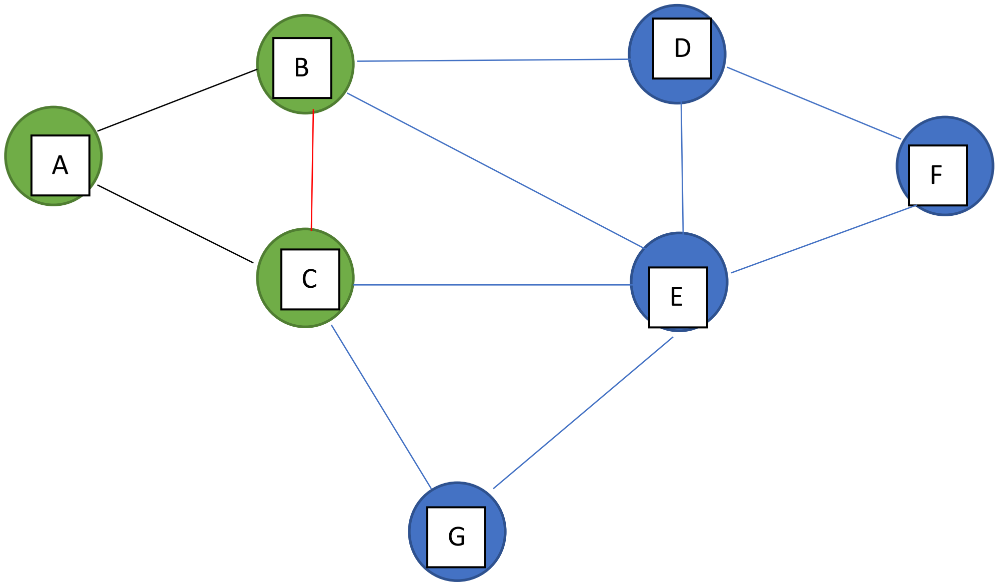


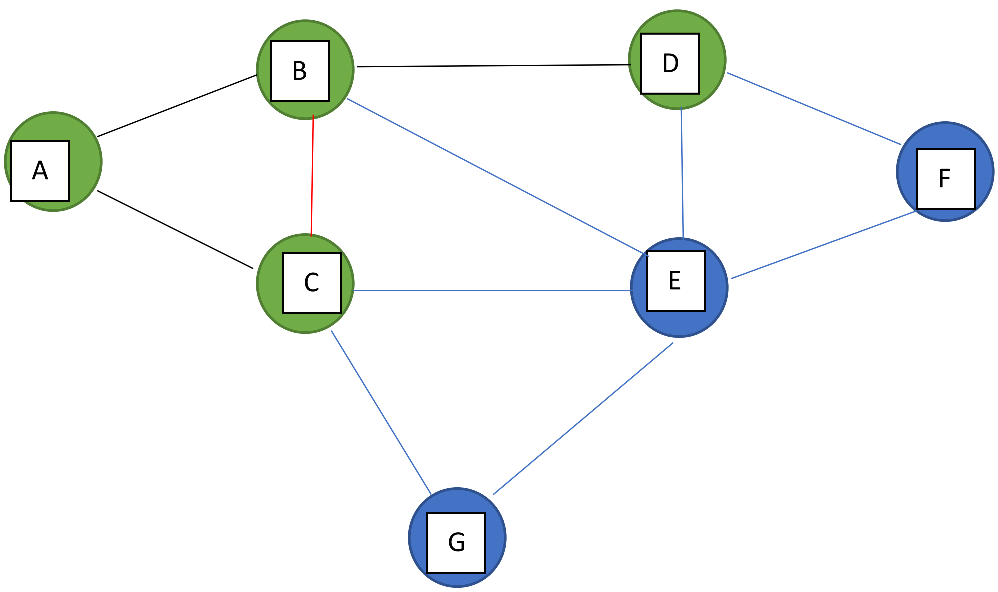
Chris Grimes
CS 46101
HW 6



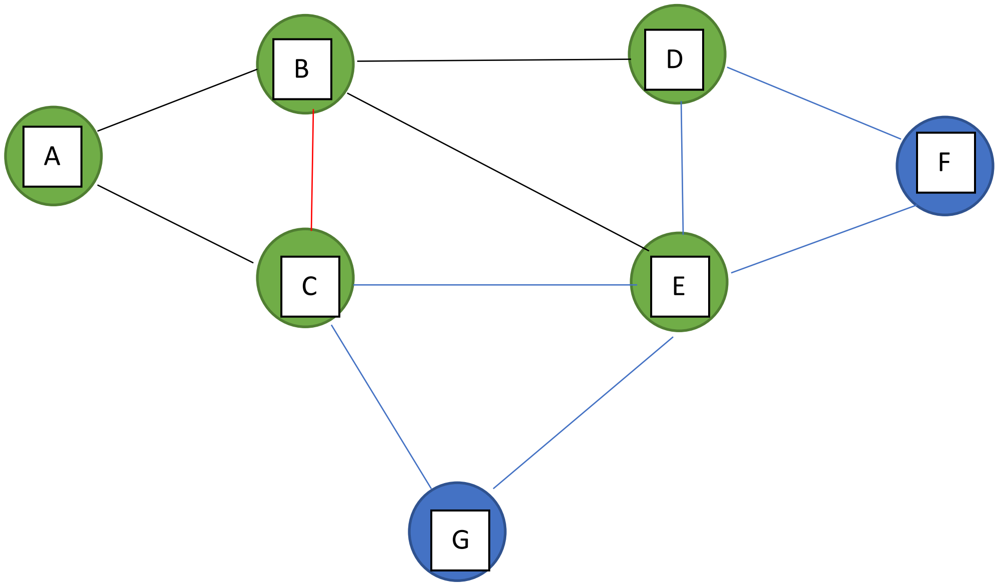



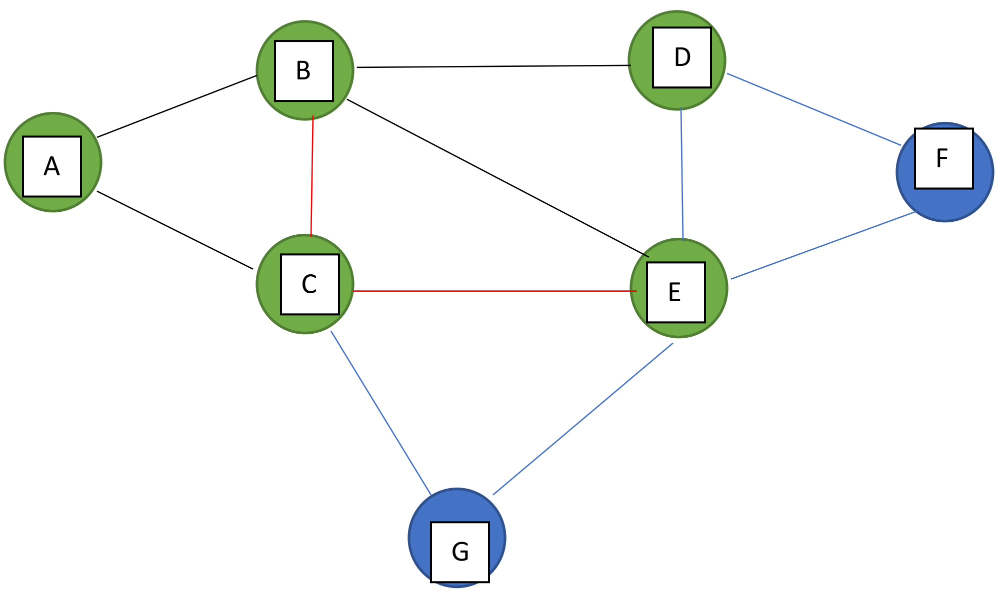
Chris Grimes
CS 46101
HW 6



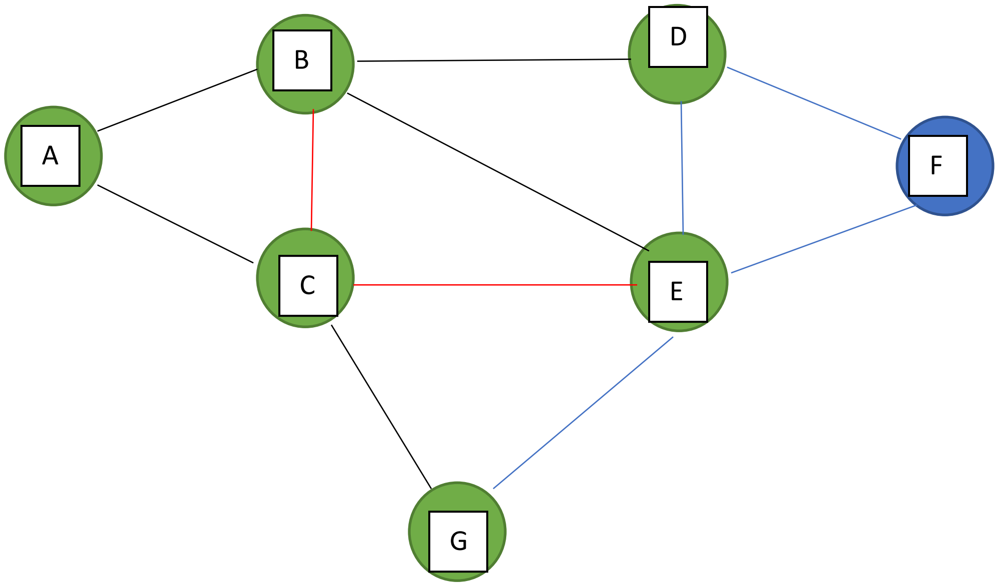



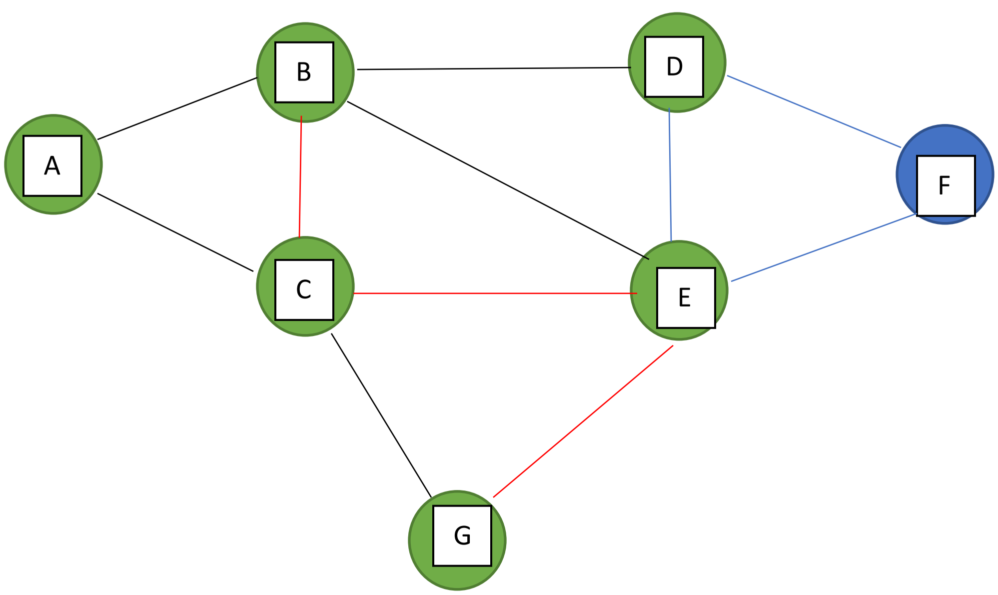
Chris Grimes
CS 46101
HW 6



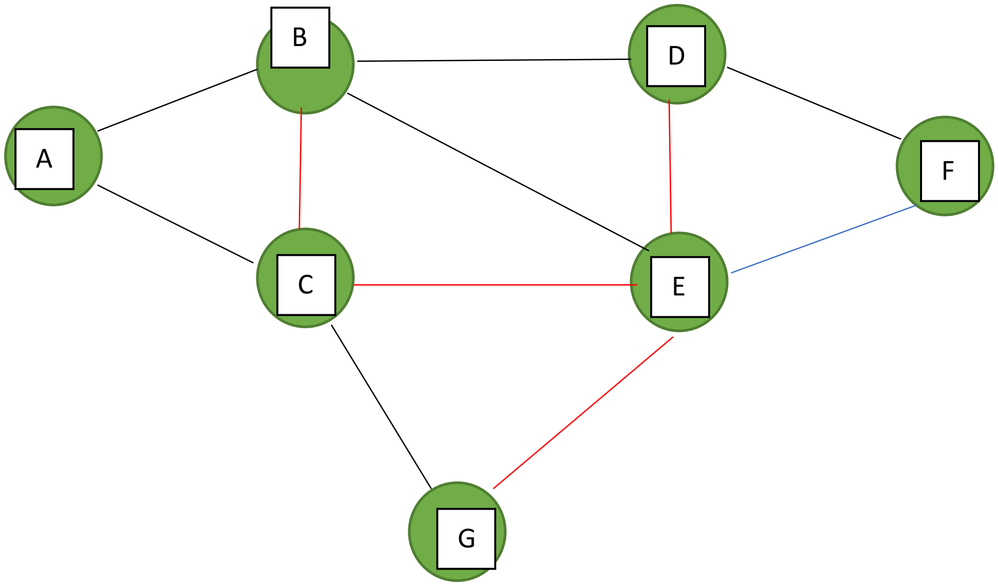



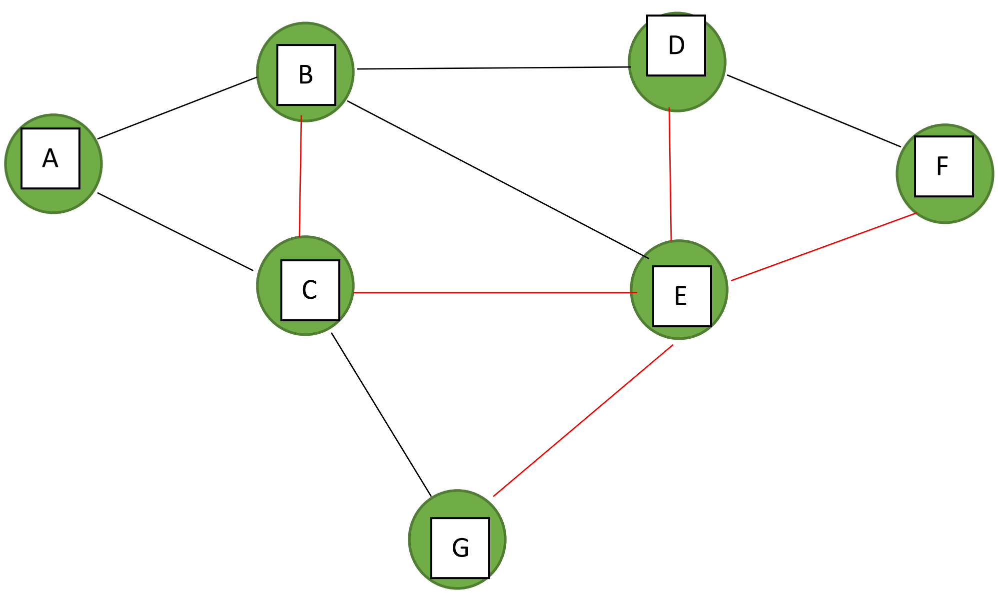
Chris Grimes
CS 46101
HW 6



Chris Grimes
CS 46101
HW 6



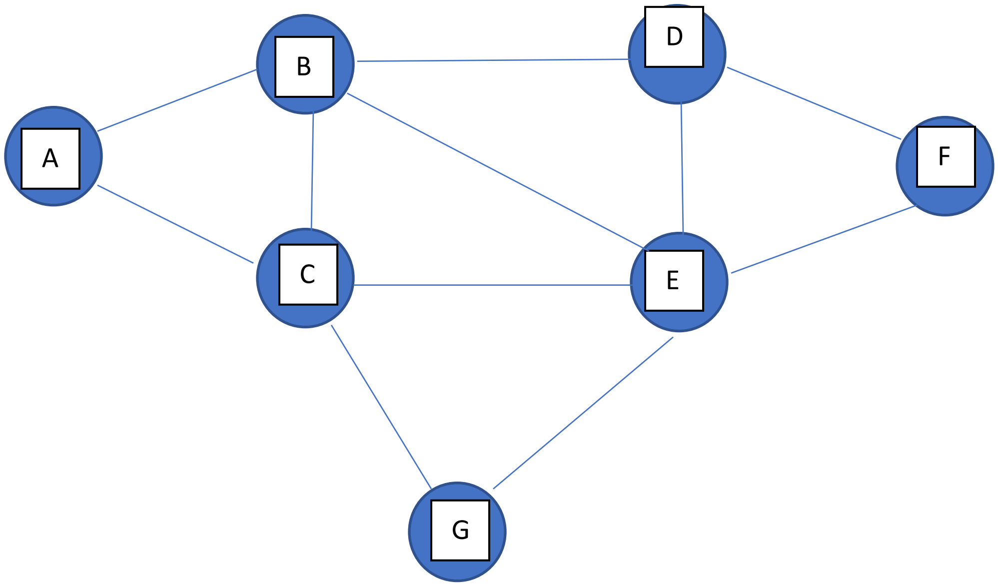



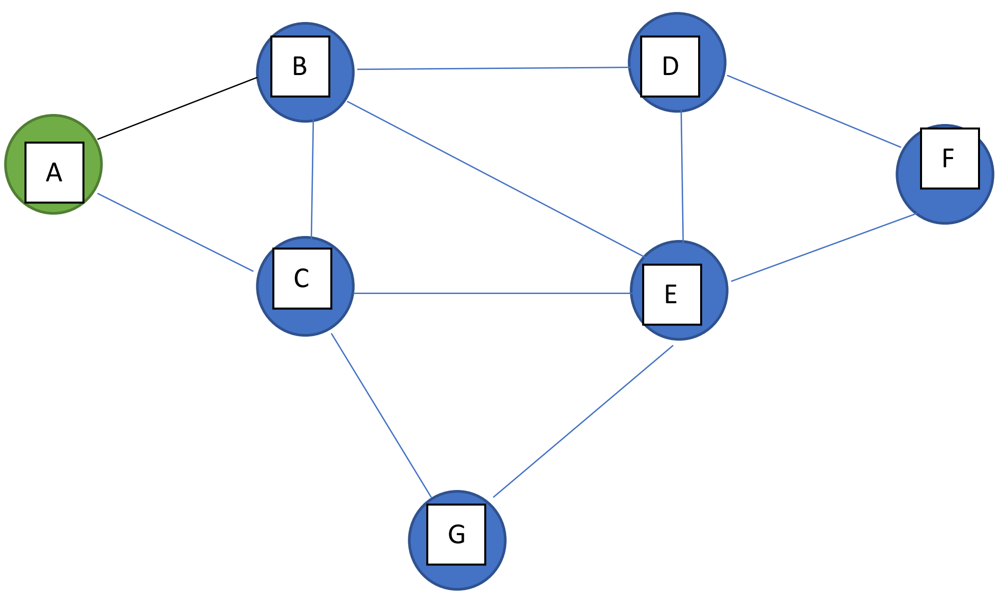
Chris Grimes
CS 46101
HW 6

(b) Trace the execution of DFS beginning at vertex A, labeling each edge as a discovery or back edge.

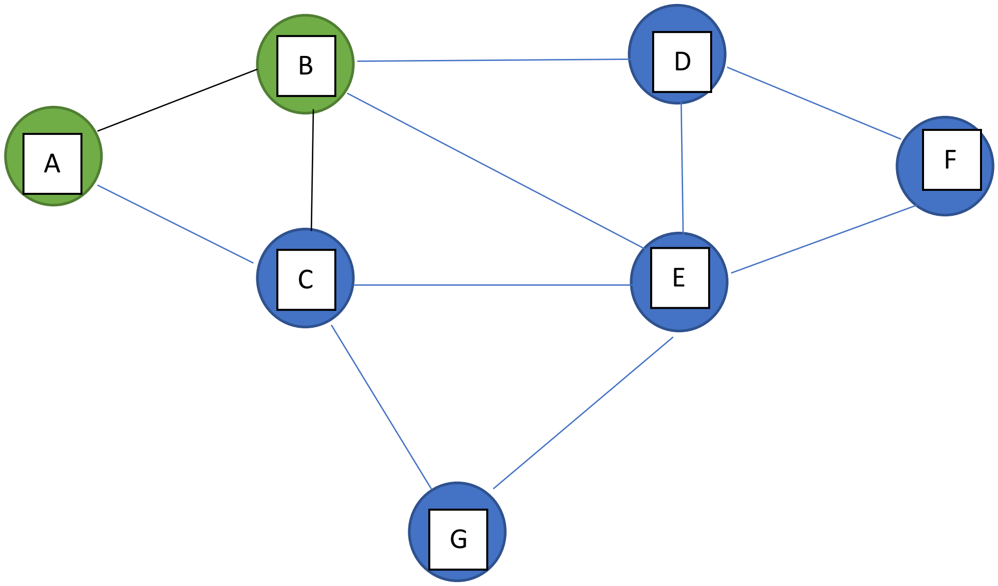
Blue circle= unexplored vertex blue line= unexplored edge

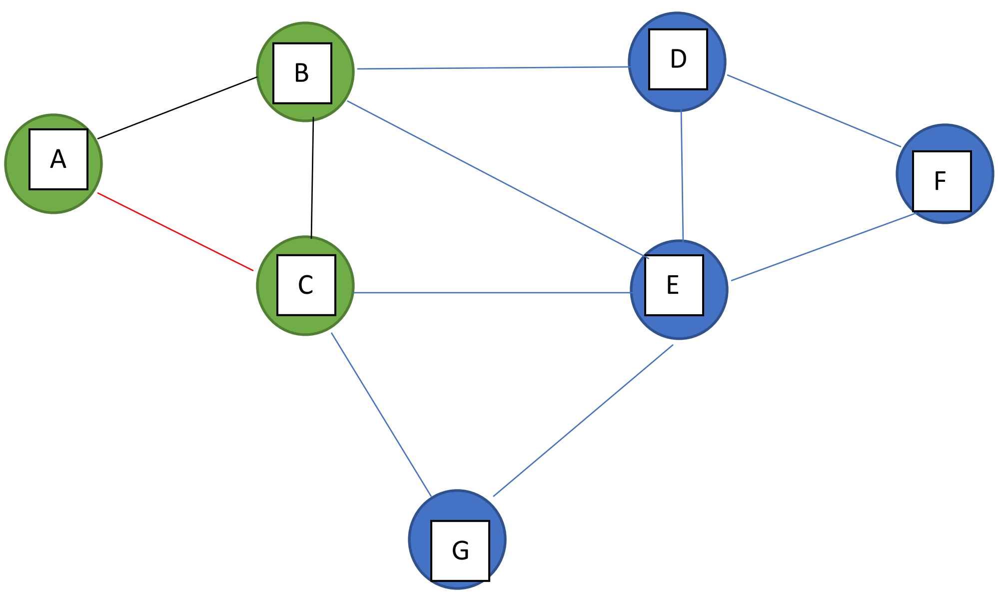
green circle= visited vertex black line= discovery edge red line= back edge



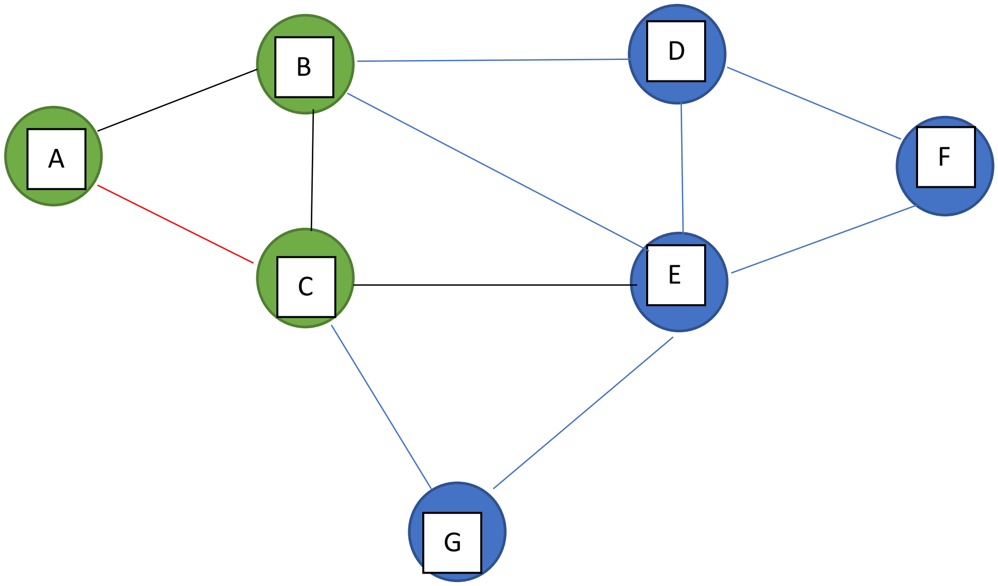



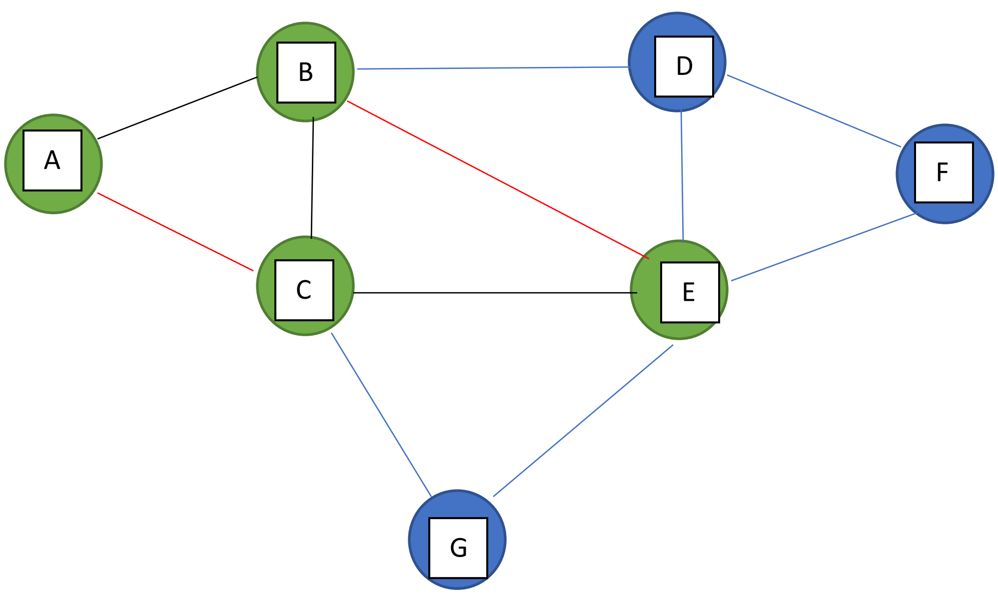
Chris Grimes
CS 46101
HW 6



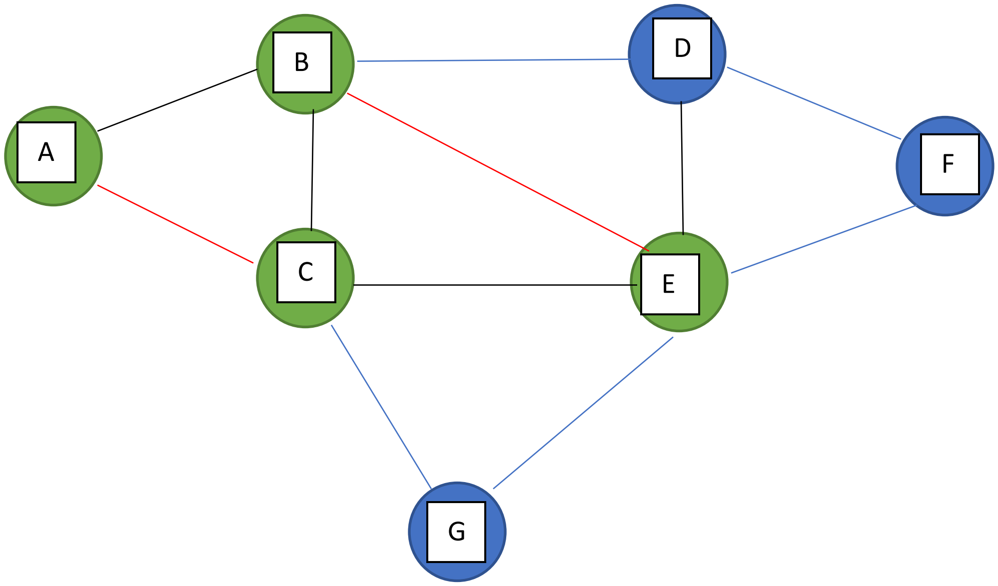



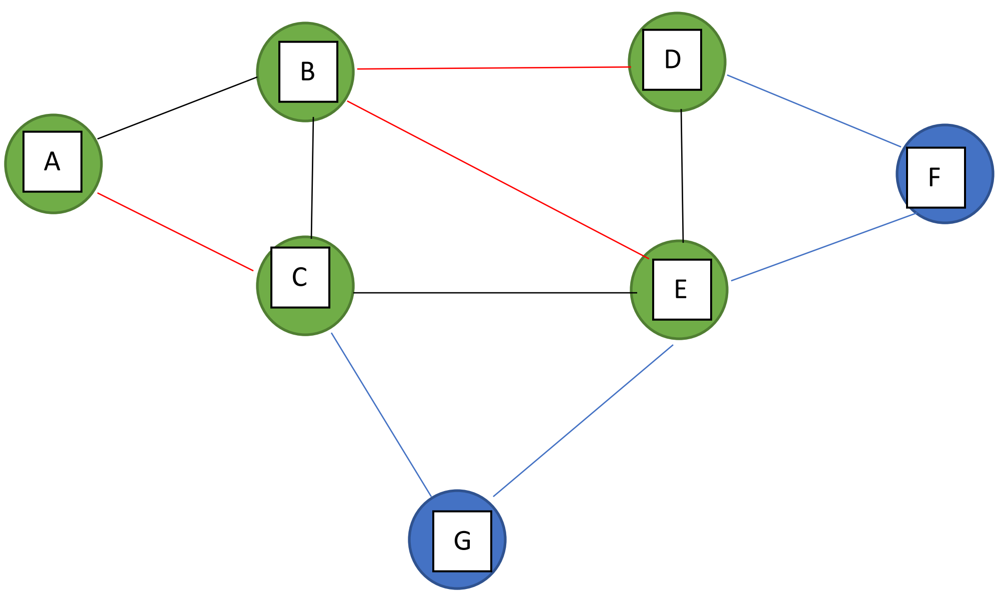
Chris Grimes
CS 46101
HW 6



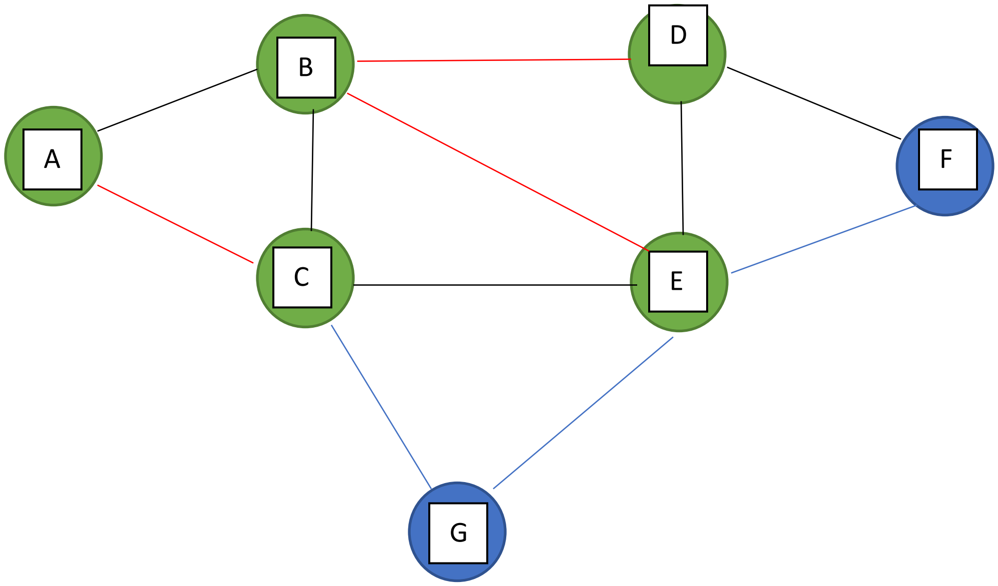



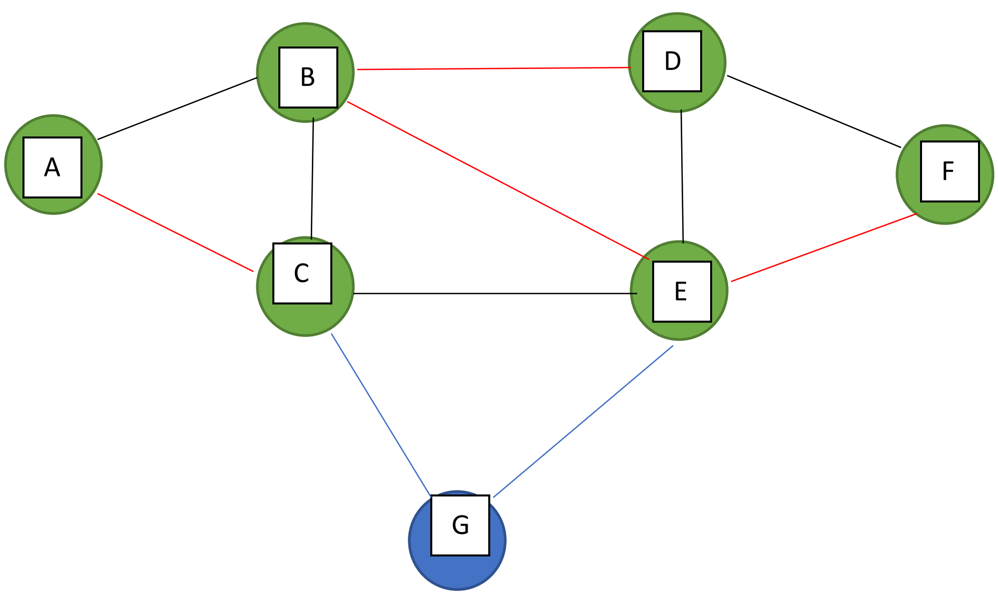
Chris Grimes
CS 46101
HW 6



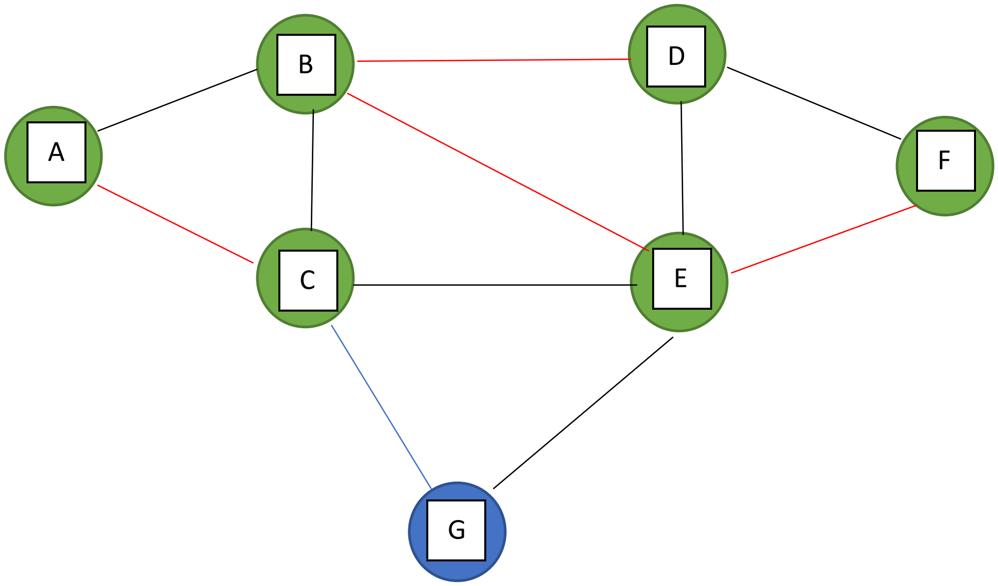



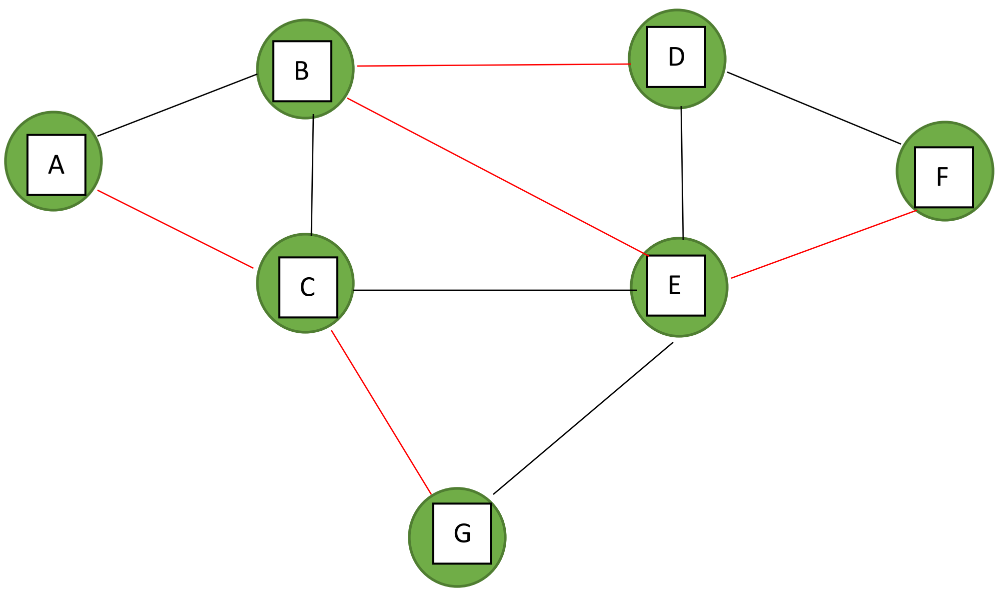
Chris Grimes
CS 46101
HW 6



Chris Grimes
CS 46101
HW 6



Chris Grimes
CS 46101
HW 6

4. (10 points) In each of the following problems, describe in a few sentences how the task is accomplished via DFS or BFS using only the discovery-edges or back/cross-edges (depending on the searching algorithm).

(a) Find a spanning tree of G, assuming G is connected.

Use dfs on G and remove any back edges from the tree that results. After removing the back edges you should be left with a spanning tree.

(b) Determine if G is acyclic.

Use dfs on G, if there exists no back edges, then there are no cycles and G is acyclic.

(c) Find a path from a vertex u to a vertex v.

Use dfs on (G,u), find the cycle that includes both u and v. Remove any and all back edges from said cycle, the resulting path should lead you from u to v.

(d) Find a shortest path from a vertex u to a vertex v.

Use bfs on (G,u), remove to cross edges from the resulting tree. After removing the cross edges you should be left with the shortest path from u to v.

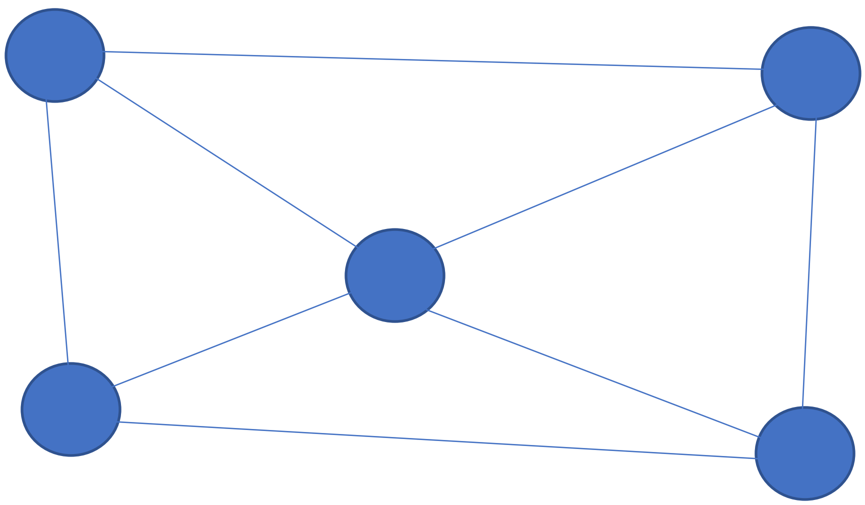
(e) Find all connected components of G.

Use dfs on G, remove any and all back edges from the resulting tree. After removing any and all back edges you should be left with a tree with edges connecting all of the connected components.

Chris Grimes
CS 46101
HW 6

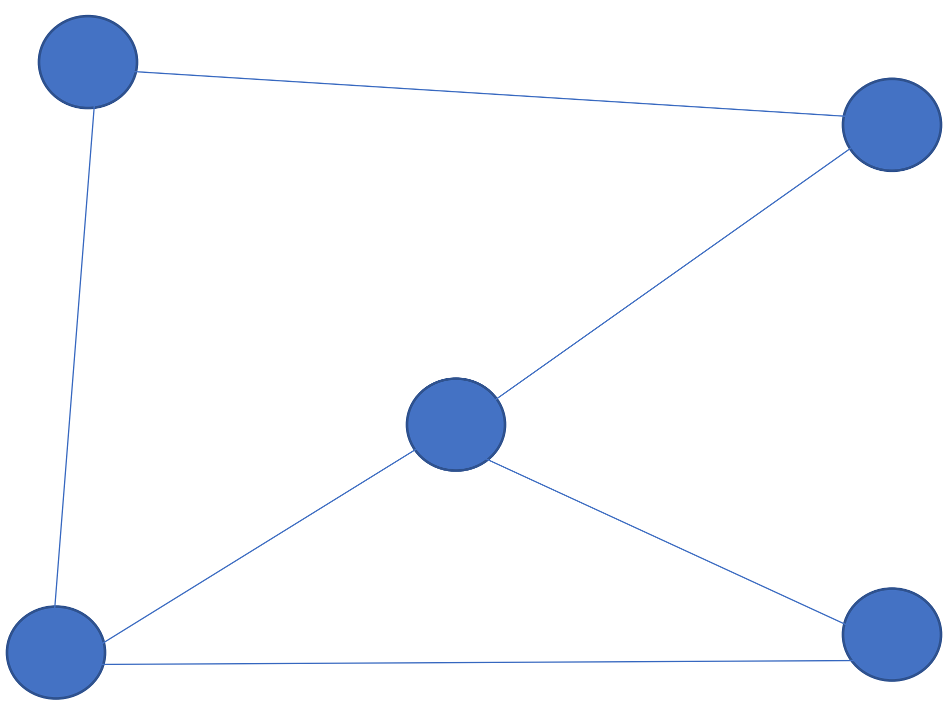
5. (10 points) A graph is triconnected if one has to remove at least 3 vertices from the graph to disconnect it. Construct examples of the following graphs or explain why it cannot be done. Assume the graph is undirected.

(a) A triconnected graph with exactly 5 vertices and 8 edges.



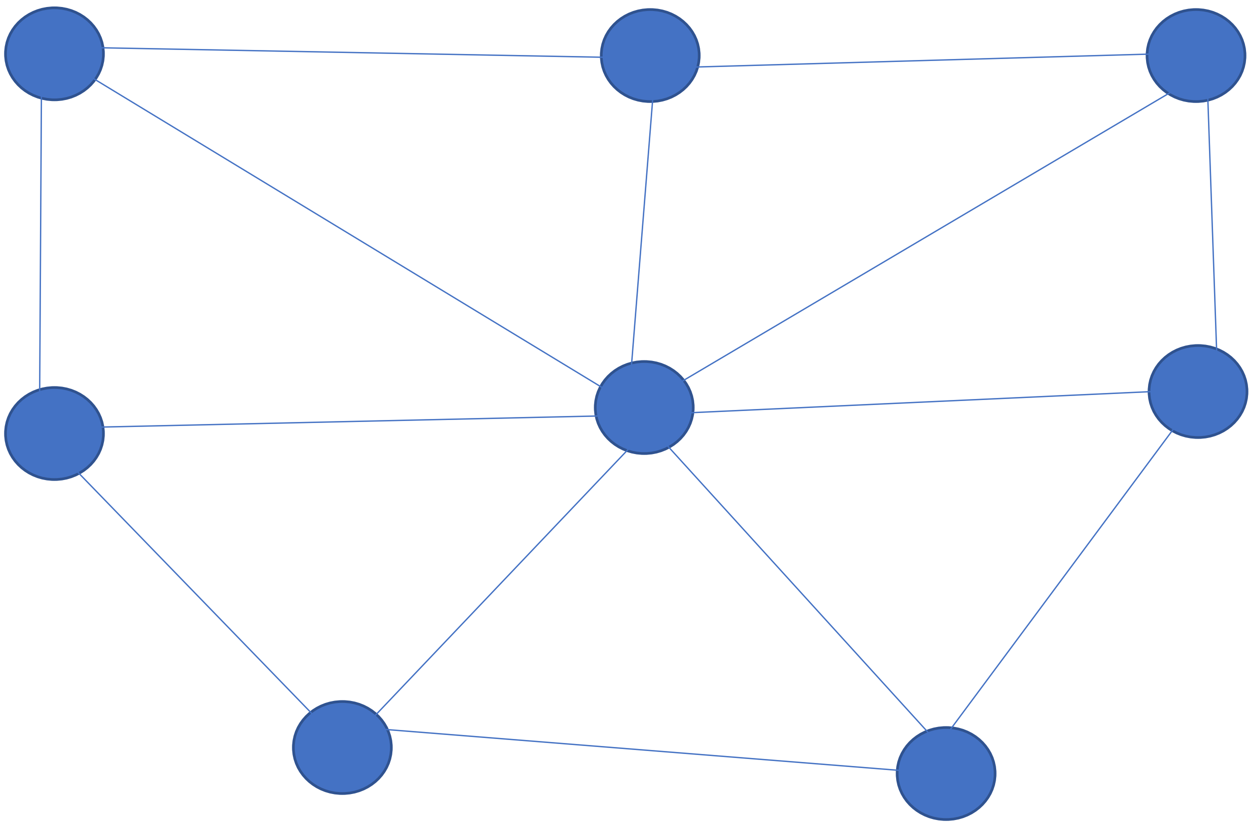
(b) A triconnected graph with exactly 5 vertices and 6 edges.

Such a graph does not exist as the removal of only two vertices would disconnect said graph.



Chris Grimes
CS 46101
HW 6

(c) A triconnected graph with exactly 8 vertices and 14 edges.



Chris Grimes
CS 46101
HW 6

6. (5 points) Bob loves foreign languages and wants to plan his course schedule to take the following nine language courses: LA15, LA16, LA22, LA31, LA32, LA126, LA127, LA141, and LA169. The course prerequisites are:

• LA15: (none)

• LA16: LA15

• LA22: (none)

• LA31: LA15

• LA32: LA16, LA31

• LA126: LA22, LA32

• LA127: LA16

• LA141: LA22, LA16

• LA169: LA32

Find a sequence of courses that allows Bob to satisfy all the prerequesites

LA15-> LA22-> LA16-> LA31-> LA32-> LA126->LA169->LA141->LA127