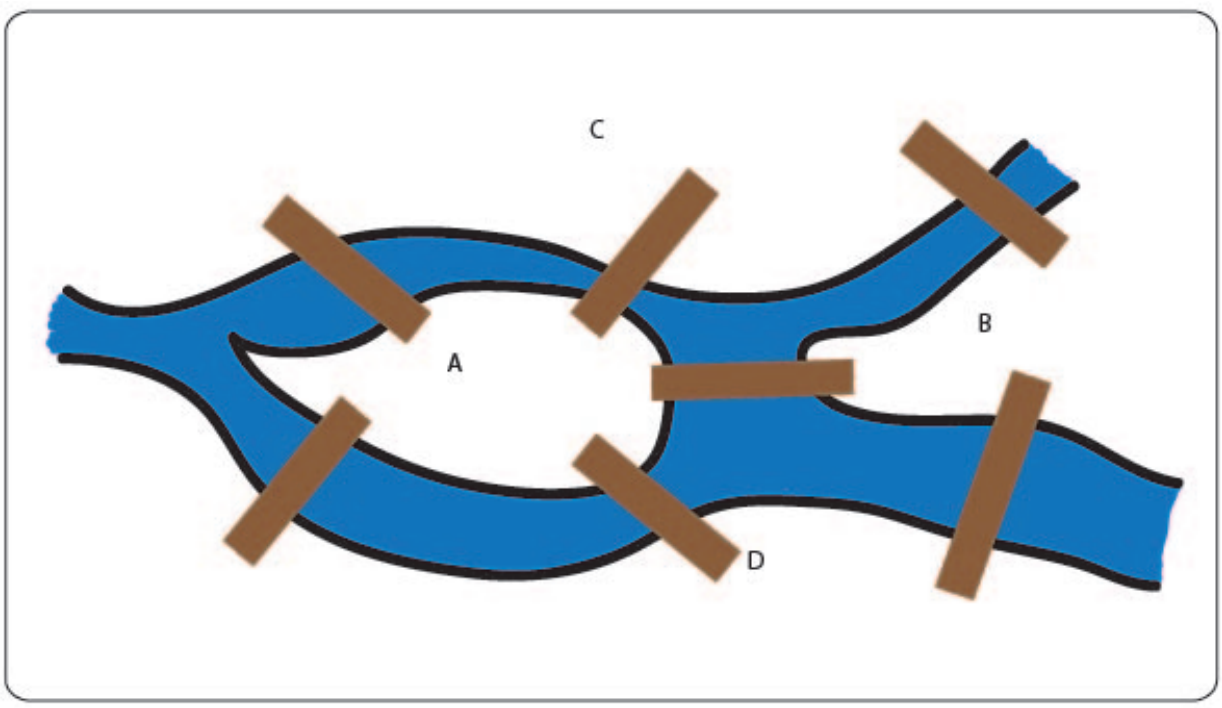
UNIVERSITY OF WATERLOO
Faculty of Mathematics
Waterloo, Ontario N2L 3G1

Centre for Education in
Mathematics and Computing

Grade 7/8 Math Circles
Graph Theory - Solutions
October 13/14, 2015

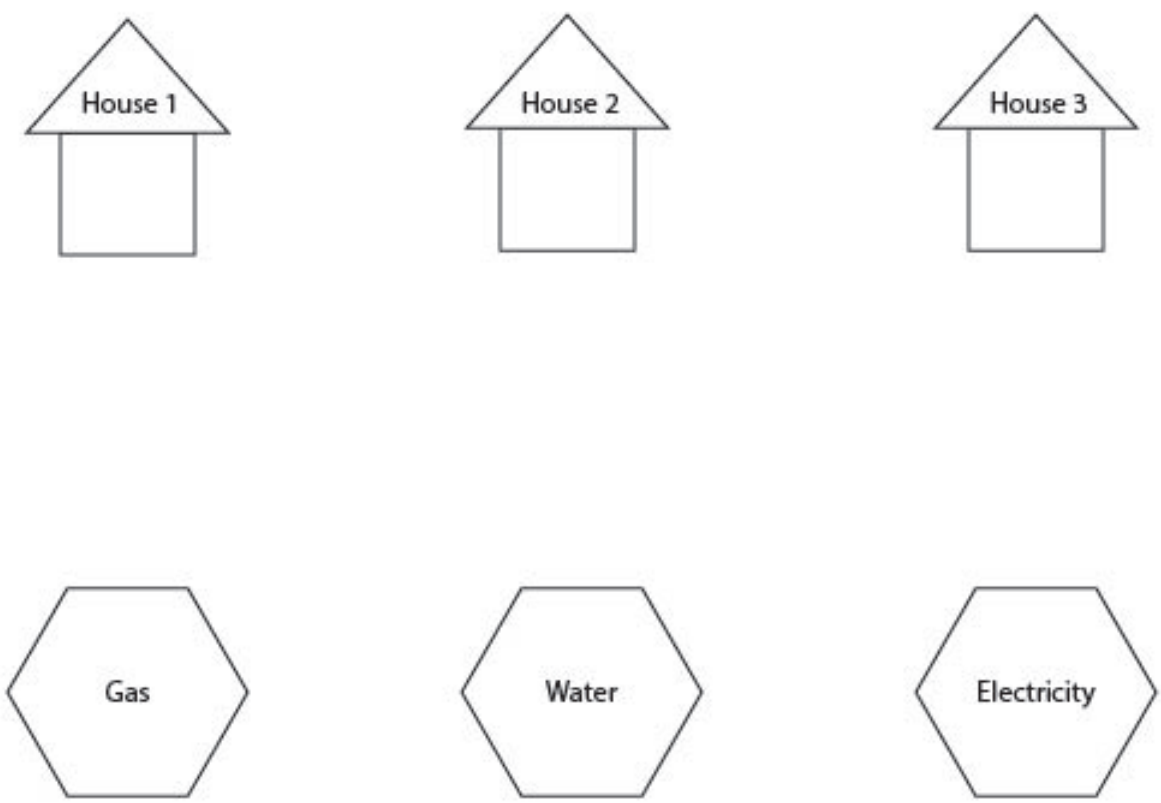
The Seven Bridges of Königsberg

In the mid-1700s the was a city named Königsberg. Today, the city is called Kaliningrad and is in modern day Russia. However, in the 1700s the city was a part of Prussia and had many Germanic influences. The city sits on the Pregel River. This divides the city into two main areas with the river running between them. In the river there are also two islands that are a part of the city. In 1736, a mathematician by the name of Leonhard Euler visited the city and was fascinated by the bridges. Euler wondered whether or not you could walk through the city and cross each bridge exactly once. Take a few minutes to see if you can find a way on the map of Königsberg below. It has been show that this problem has no solution since each land mass $(A, B, C, \& D)$ have an odd number of bridges connecting them.



The Three Utilities Problem

Suppose you have a neighbourhood with only three houses. Now, each house needs to be connected to a set of three utilities (gas, water, and electricity) in order for a family to live there. Your challenge is to connect all three houses to each of the utilities. Below is a diagram for you to draw out where each utility line/pipe should go. The catch is that the lines you draw to connect each house to a utility can't cross. Also, you can't draw lines through another house or utility plant. In other words, you can't draw House 1's water line through House 2 or through the electricity plant. This problem also has no solution when done on paper. In reality, you could dig holes under the ground at different depths, however this is not possible on a sheet of paper. Connecting each house and utility results in a $K_{3,3}$ graph, which is non-planar (see section on planarity below).



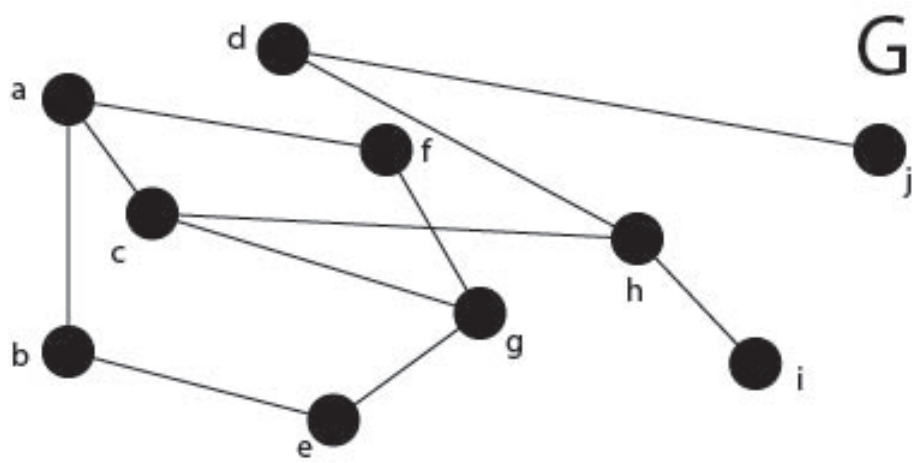
Introduction to Graph Theory

Both of these problems are examples of Graph Theory. Graph Theory is a relatively young branch of mathematics, and it was Euler's solution to the 7 Bridges problem in 1736 that represented the first formal piece of Graph Theory. But what is Graph Theory? In order to answer that question and to explore some of the applications of Graph Theory, we first need some definitions.

Definitions

- A **vertex** (plural: **vertices**) is a point. In drawings, we usually represent these as circles.
- An **edge** e is an unordered pair of vertices. For example, $e = \{a, b\}$. For simplicity, we often write the names of vertices as ab rather than $\{a, b\}$ In drawings, edges are represented by lines between the two vertices. An edge ab can also be written as ba - they are the same edge and you only need to list one.
- For an edge $e = \{a, b\}$, we call a and b the **endpoints** of edge e
- A vertex a is **incident** with an edge e if a is an endpoint of e
- Two vertices a and b are **adjacent** if there is an edge e with a and b as its endpoints
- The vertices adjacent to a vertex a are called the **neighbours** of a
- A **graph** $G = (V, E)$ is comprised of a set V of vertices, and a set of edges E.
- A **walk** is a sequence of vertices where each vertex is adjacent to the vertex before it and after it
- A **path** is a walk that doesn't repeat vertices
- A **cycle** is a path that begins and ends at the same vertex

These definitions can be difficult to understand as abstract concepts. To make these definitions more concrete, consider the graph G below:



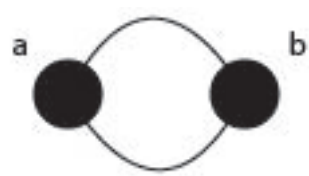
Example 1

1. List the vertices in G $\quad V = \{a, b, c, d, e, f, g, h, i, j\}$
2. List the edges in G $\quad E = \{ab, ac, af, be, cg, ch, dj, dh, eg, fg, hi\}$
3. List the edges incident with vertex c $\quad ac, cg, ch$
4. List the vertices adjacent to g $\quad c, e, f$
5. Find a path from j to b $\quad$ Multiple paths exist, one is j, d, h, c, a, b
6. Find a cycle in G $\quad$ There are 3 cycles in G. They are: a, b, e, g, f, a or a, b, e, g, c, a or a, c, g, f, a. Note that you can start and end the cycle at any of the vertices in it.

Graph Rules

In different applications, graphs can have a wide variety of rules associated with them. For example, in some applications graphs can be *directed* so they only go from a to b but not from b to a (think of it like a one-way street). For the graphs in this lesson we're going to focus one graphs that satisfy these rules:

- Our graphs will be undirected (i.e. like a two-way street)
- Graphs will not contain duplicate edges. What this means is there will only be one edge from a to b. We will not have situations like the one below:



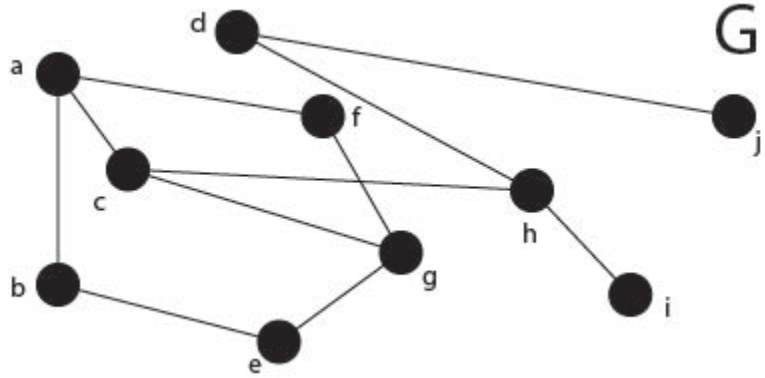
- Graphs will not have self-loops (edges that start and end at the same vertex). Below is an example of a self-loop:

Planarity

A **planar graph** is a graph that we can draw without any of the edges crossing. We call the drawing of the graph (without edges crossing) a **planar embedding**. But just because a drawing has edges that cross, is the graph non-planar?

While this would make our lives as mathematicians much easier, it isn't the case. Remember, a graph is just defined by its set of vertices V, and its set of edges E. If we can find a drawing where the edges do not cross, then the graph is planar.

To see exactly what this means, let's consider the graph G we used when we went through the graph definitions.



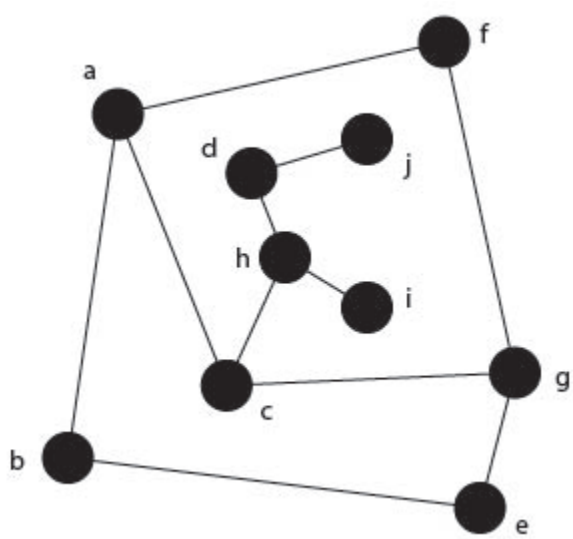
Clearly the edges $\{c, h\}$ and $\{f, g\}$ cross. However I claim that this graph is planar. Recall from our first exercise that the vertices of G are:

$$V = \{a, b, c, d, e, f, g, h, i, j\}$$

and

$$E = \{\{a, b\}, \{a, c\}, \{a, f\}, \{b, e\}, \{c, g\}, \{c, h\}, \{d, h\}, \{d, j\}, \{e, g\}, \{f, g\}, \{h, i\}, \}$$

Now, consider the graph G':



If you look at the vertices and edges of G', what do you notice?

The vertices and edges of G' and G are the same. Since we said a graph is only determined by its edges and vertices, we can say that G and G' are the same graph. What's more is that

since our drawing G' doesn't have any edges crossing, we can say that it is a planar graph. This means that G is also a planar graph, even though our first drawing had edges crossing each other.

So how can this help us determine if a graph is planar?

The answer to that is we can do essentially move around the vertices and edges of a graph until non of the edges across. This method works well if a graph is planar, but what if a graph isn't planar? Are we simply stuck rearranging edges and vertices forever?

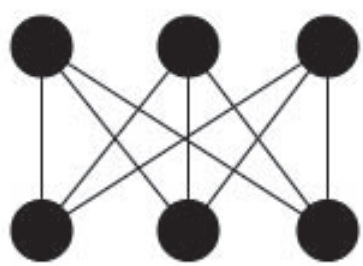
Thankfully we aren't, and there are a few ways we can tell if a graph is non-planar.

Identifying non-planar graphs

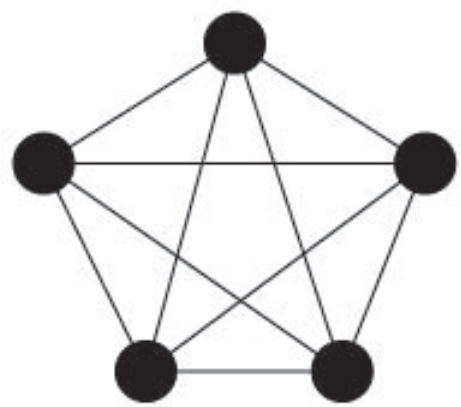
Theorem 1 A graph with v vertices, and e edges, is non-planar if $e > 3v - 6$

This formula gives us a quick way to see if something may be non-planar. Keep in mind that this formula doesn't guarantee that a graph is planar. To definitively say whether a graph is non-planar, we need to introduce two special graphs first.

These graphs, $K_{3,3}$ and K_5 are depicted below. On your own, check whether or not these graphs are planar.



$K_{3,3}$



K_5

Using Theorem 1, we can see that both of these graphs are non-planar. In fact, it can be shown that any non-planar graph contains a structure that is similar to either $K_{3,3}$ or K_5. This is beyond the scope of this Math Circle

Colourings

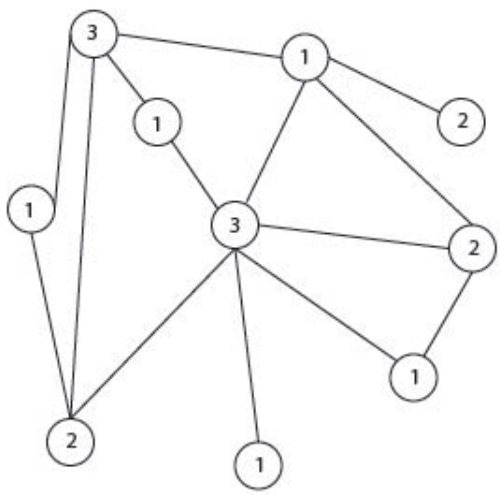
How many colours does it take to colour a map of the United States? What about a map of Africa?

While it may not be obvious, this problem (a more general form of it) has been one of the central problems in Graph Theory for a very long time. It turns out that we can colour any

planar graph in just *four* colours. In fact, this problem was initially posed in 1852, and a correct proof was not submitted until 1976, over 100 years later.

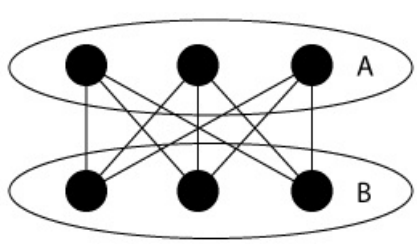
But what does it mean to colour a graph? It's easy to understand how to colour a map, we simply colour each region a different colour than it's neighbours. It turns out that colouring a graph is essentially the same thing. We assign each vertex a different colour than the vertices adjacent to it. Even though we talk about colours like red or green, we usually just label the vertex with a number, and each different number represents a new colour.

Below is an example of a colouring. Notice that this graph can be coloured using just three colours.



Colouring Bipartite Graphs

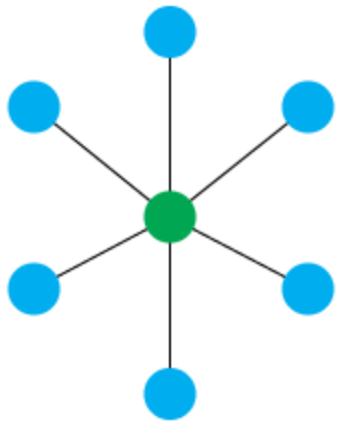
A **bipartite graph** is a graph where we can split the vertices into two groups, A and B, and all of the edges have one endpoint in A and one in B. This means that for any vertex in A, all of the vertices it is adjacent to are in B. One example of a bipartite graph is $K_{3,3}$. We can see that if we call the top row of vertices A and the bottom row of vertices B, then all of the edges go from A to B, and none from A to A or B to B.

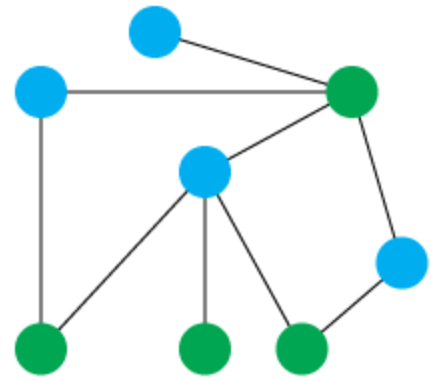


But bipartite graphs don't have to have their groups of vertices spaced as nicely as our drawing of $K_{3,3}$.

Exercise 2

While they might not look like it at first glance, both of the following graphs are bipartite - can you find the groups of vertices A and B? The green vertices are A and the blue vertices are B. You may have also switched which group you called A and B





Think about colouring these graphs. Is there anything special that you notice?

We already know that we can colour any planar graph with four colours, but bipartite graphs are different. In fact, any planar bipartite graph can be coloured with just *two* colours.

Let's think about why this is the case. Remember that in a colouring, we need to make sure that for any vertex, none of its neighbours can be coloured with the same colour. Now look at any vertex - call it v - in a bipartite graph - for simplicity, let's say it is in group A.

Since the graph is bipartite, what do we know about v's neighbours?

Well, since v is in A, then all of v's neighbours are in B. Let's suppose that we colour v red (our first colour). We need to colour all of v's neighbours with a different colour. Can we just colour them all with the same colour?

It turns out that we can. Since all of the neighbours of v are in B, we know that there are no edges between them. This means that none of these vertices are adjacent to each other, and so we can colour all of these vertices the same colour. In fact, we can colour all of the vertices in B (not just the neighbour's of v) the same colour.

Now, the only vertices we haven't coloured are the ones in AS (except v of course). Since the neighbours of all these vertices are in B, we can't colour them blue. But could we colour them red like v?

We can! The reason being that none of these vertices are adjacent to anything else in A. This means we can colour everything in A red, and everything in B blue, meaning the whole graph can be coloured using only two colours.

Weighted Graphs & Minimum Spanning Trees

So far, we've only dealt with unweighted graphs. Now we are going to look breifly at weighted graphs. In a weighted graph, each edge is given a numerical **weight**. These weights can represent many different things depending on the problem.

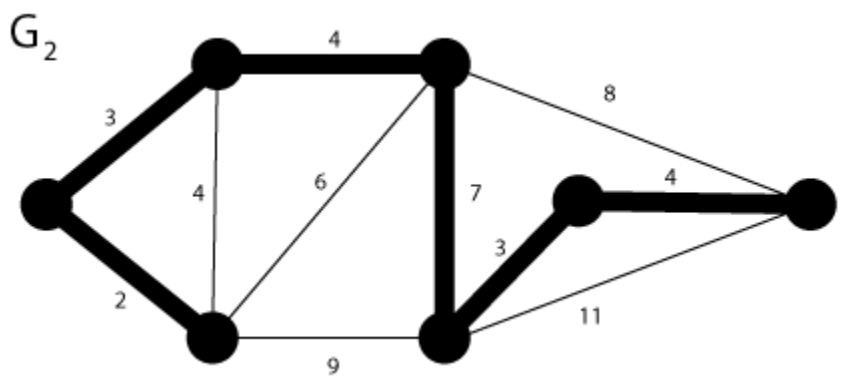
For example, if the graph was a map of the roads between cities in Ontario, then the weights could represent distances between cities. If the edges represented flights between countries then the weights could represent the cost of a flight.

Weighted graphs are extremely useful in the real world, and many every day applications rely on them. Google uses weighted graphs when it gives you directions, and airlines use weighted graphs to help determine prices for plane tickets.

Today, we will be looking at **minimum spanning trees** and how we can construct these trees. A **tree** is simply a graph with no cycles. A **spanning tree** is a tree that contains all of the vertices of the original graph. A **minimum spanning tree** is a spanning tree with the smallest total weight of *all* spanning trees. We can find the weight of a spanning tree by adding up the weights on each of its edges.

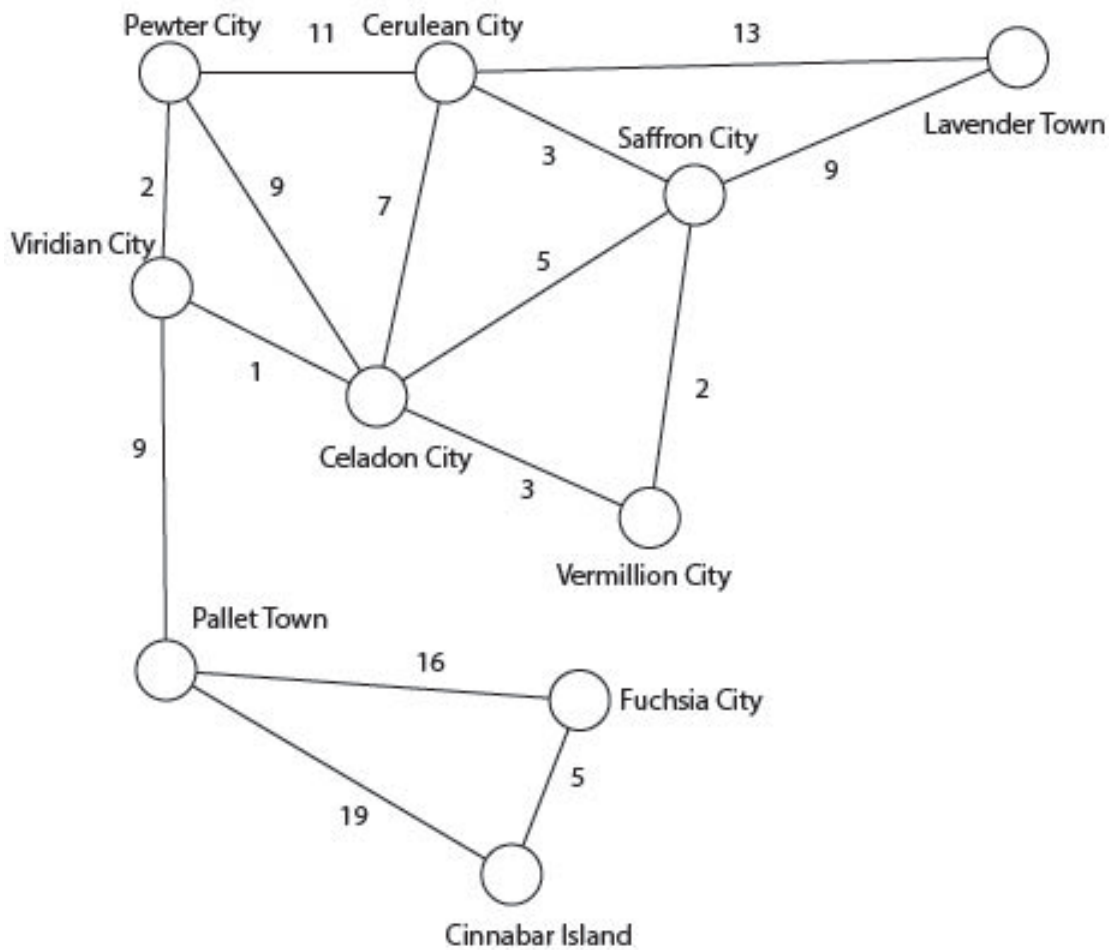
Exercise 3

Consider the graph G below.



1. True or False: G is a tree False - G has cycles in it
2. Find any tree in G Solutions will vary. Any set of edges from G that does not have a cycle is a valid answer
3. Draw any spanning tree of G. (Highlight or circle the edges of G that you are including in your spanning tree) See edges highlighted in above diagram
4. Calculate the weight of your spanning tree. The weight of this spanning tree is 23
5. Try and find another spanning tree with a lower weight This is a minimum spanning tree. The minimum weight is 23

One particular application that we will be looking at is building a train system to connect a group of cities. Our goal is to do this for as low a cost as possible. The graph below is a map of the region. The vertices represent cities, and the edges represent potential railroad tracks. The weights on each edge represent the cost to build a railroad between the two cities.



Exercise 4

On the graph above, try and connect all of the cities with railroads for the lowest total cost.
Solutions to this exercise will vary greatly from student to student

Think about what edges you had in your railroad system, then answer the following questions about your approach to this problem.

1. Were there any edges that you definitely wanted to include?
2. What about ones you wanted to avoid?
3. How did you pick which edges to include?

Prim's Algorithm Instead of simply trying every possible combination, mathematicians has developed an *algorithm* or series of steps to follow to determine the minimum spanning tree. One such algorithm is known as Prim's Algorithm. We will follow the steps of this algorithm to find the railway system with the lowest cost. The edges in red make the minimum spanning tree that results from Prim's Algorithm

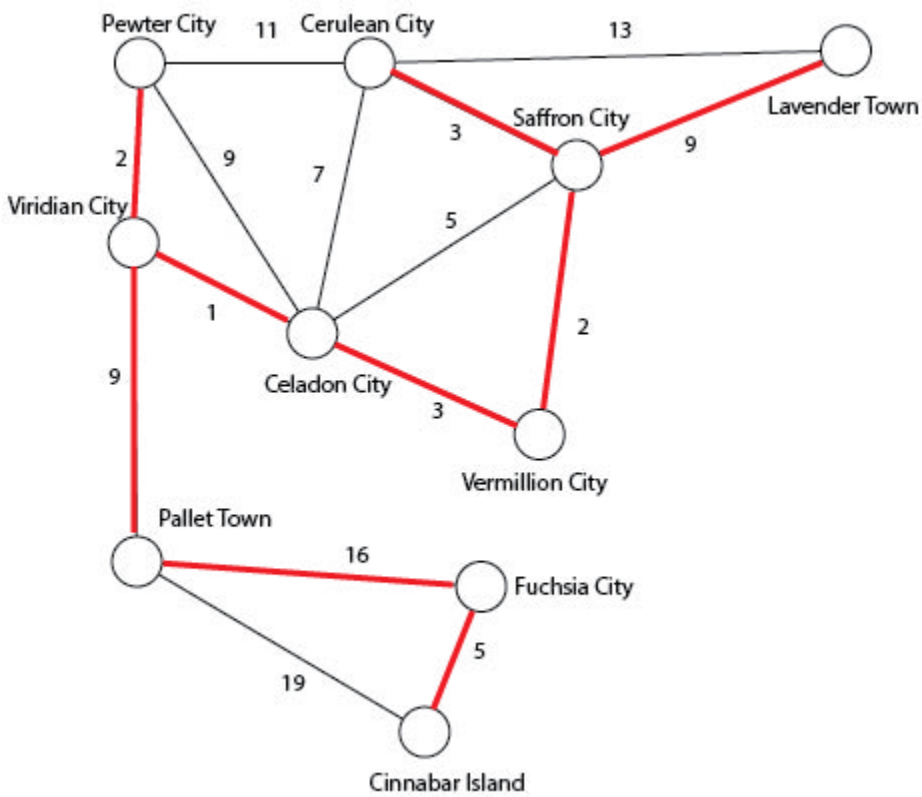
Step 1. Select a starting vertex (In our example, we will choose Pallet Town)

Step 2. Call S the group of all the vertices currently connected to your tree. Call V all the other vertices. (Note that the first time you reach this step, S will just be your starting vertex)

Step 3. If S contains all of the vertices in the graph, stop. You have your minimum spanning tree!

Step 4. Find all of the edges with one endpoint in S and one endpoint in V. Select the edge with the smallest weight and add it to your tree. If there is more than one edge with the smallest weight, you can pick any of them to add. (For now, to add an edge to your tree, just circle or highlight the edge)

Step 5. Return to step 2



Problem Set

1. For each graph below, answer the following:

   (a) List the edges and vertices of the graph.
   $G_1 : V = \{r, s, t, u, v, w, x, y, z\}$, $E = \{sv, su, sw, wx, vy, vr, yz, rt\}$
   $G_2 : V = \{r, s, t, u, v, w, x, y, z, a, b\}$,
   $E = \{su, sv, sr, uw, vz, vr, rb, wz, wx, xy, xt, at, ab\}$

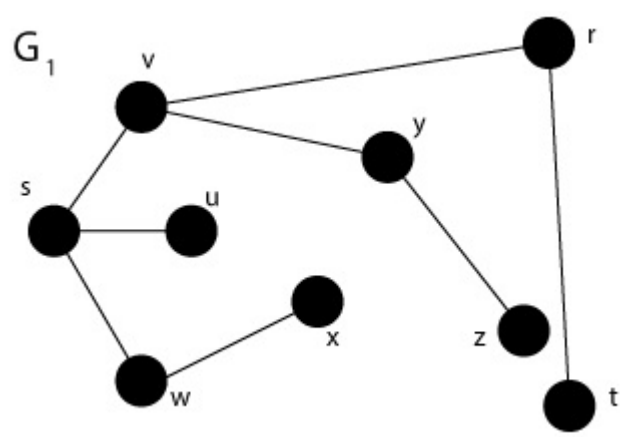
   (b) How many edges and vertices are there?
   G_1 : 9 vertices, 8 edges
   G_2 : 11 vertices, 13 edges

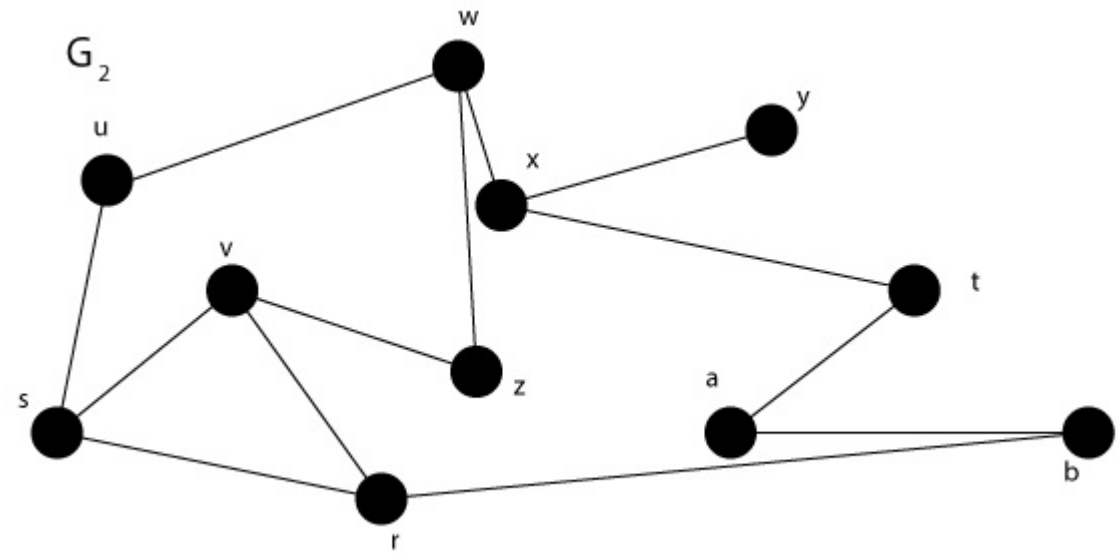
   (c) List the neighbours of the vertex v
   $G_1 : r, s, y$
   $G_2 : r, s, z$

   (d) How many edges are incident with s
   G_1 : 3 The edges are: sv, su, sw
   G_2 : 3 The edges are; sv, su, sr

   (e) Find a walk between s and t. Is your walk a path? Why or why not?
   Answers will vary for G_2 from student to student one possible answer is provided.
   There is a unique walk in G_1
   $G_1 : s, v, r, t$. This walk is a path as no vertices repeat.
   $G_2 : s, v, r, s, u, w, z, v, r, b, a, t$ This walk is not a path as multiple vertices (s, v, r) repeat.

   (f) Is the graph a tree? if not, find a cycle.
   G_1 : This graph is a tree.
   G_2 : This is not a tree. One cycle is s, v, r, s

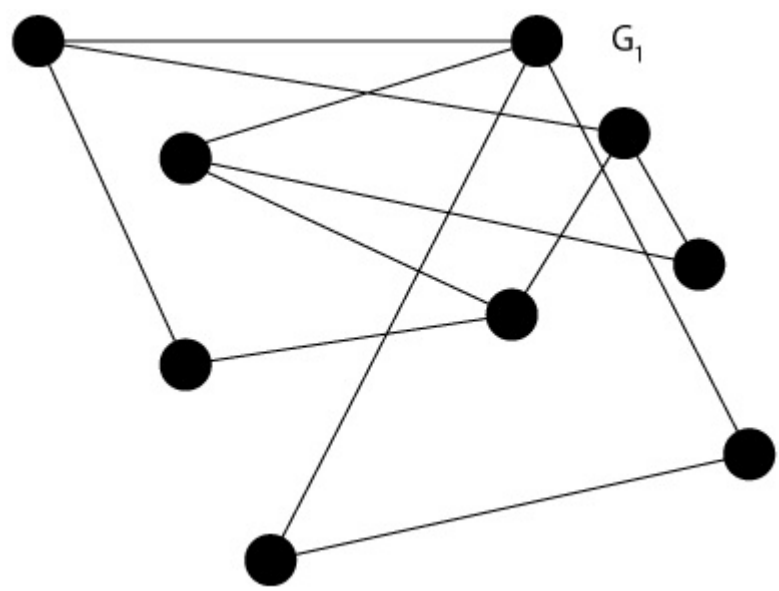


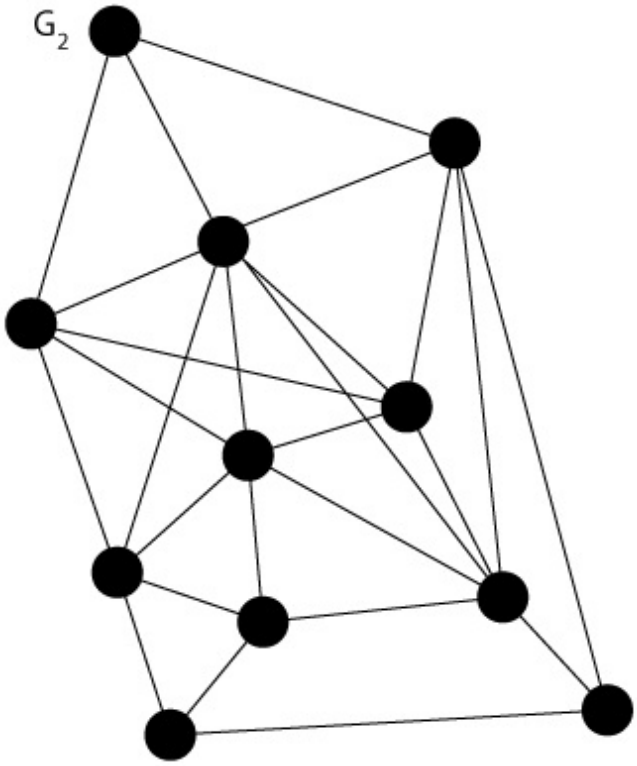


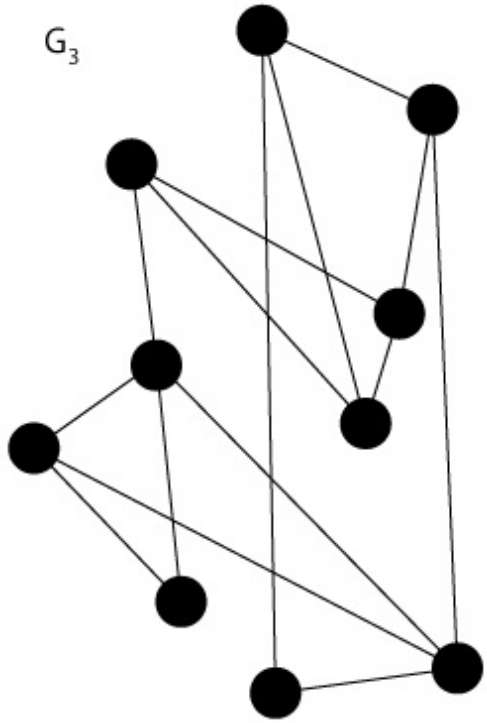
2. For each graph below:

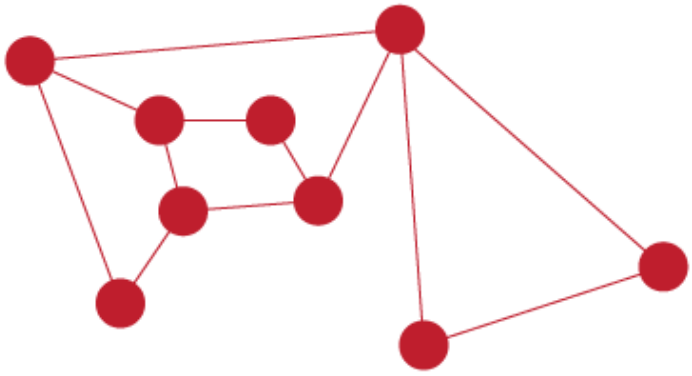
   (a) Determine whether or not the graph is planar
   G_1 and G_3 are planar. G_2 is non-planar.

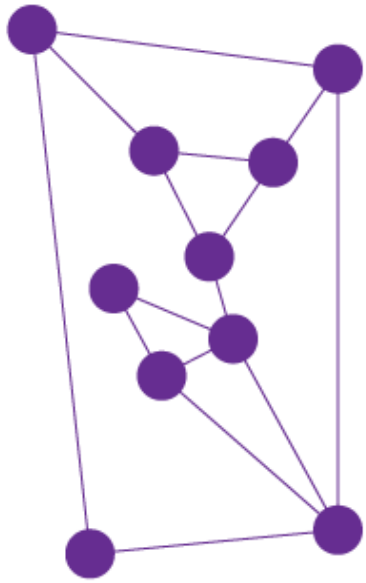
   (b) If the graph is planar, draw a planar embedding.
   A planar embedding of G_1 is in red, and a planar embedding of G_3 is in purple.



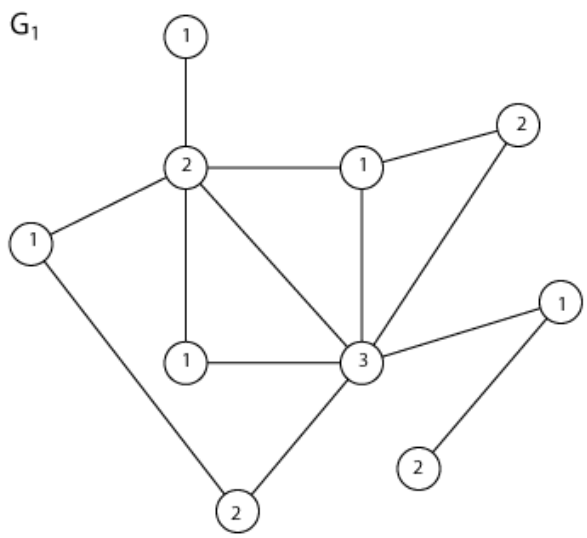

G_2

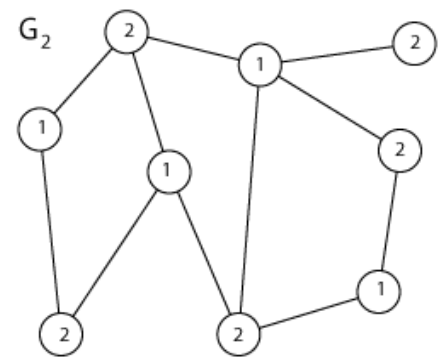

G_3



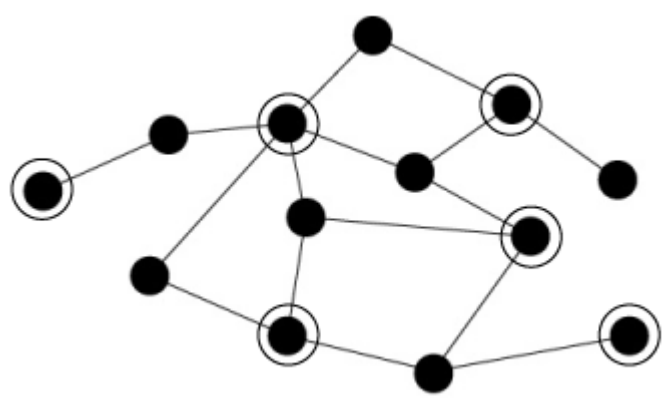


3. Colour the following graphs using the fewest colours possible. How many colours do you need?
   G_1 requires 3 colours and G_2 requires only 2 colours. Sample colourings are indicated by numbers in the vertices.

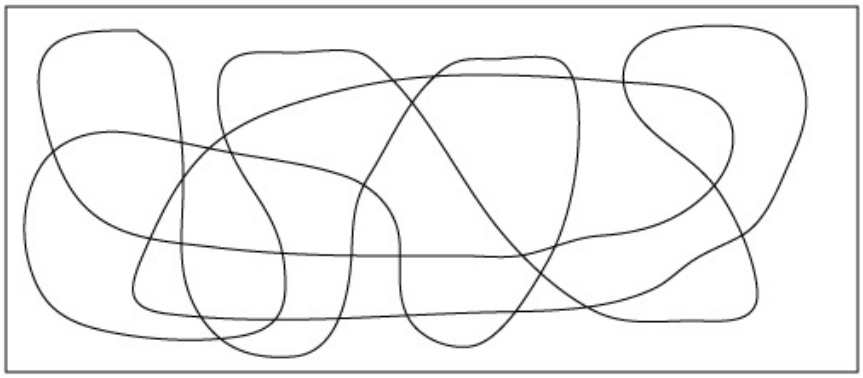




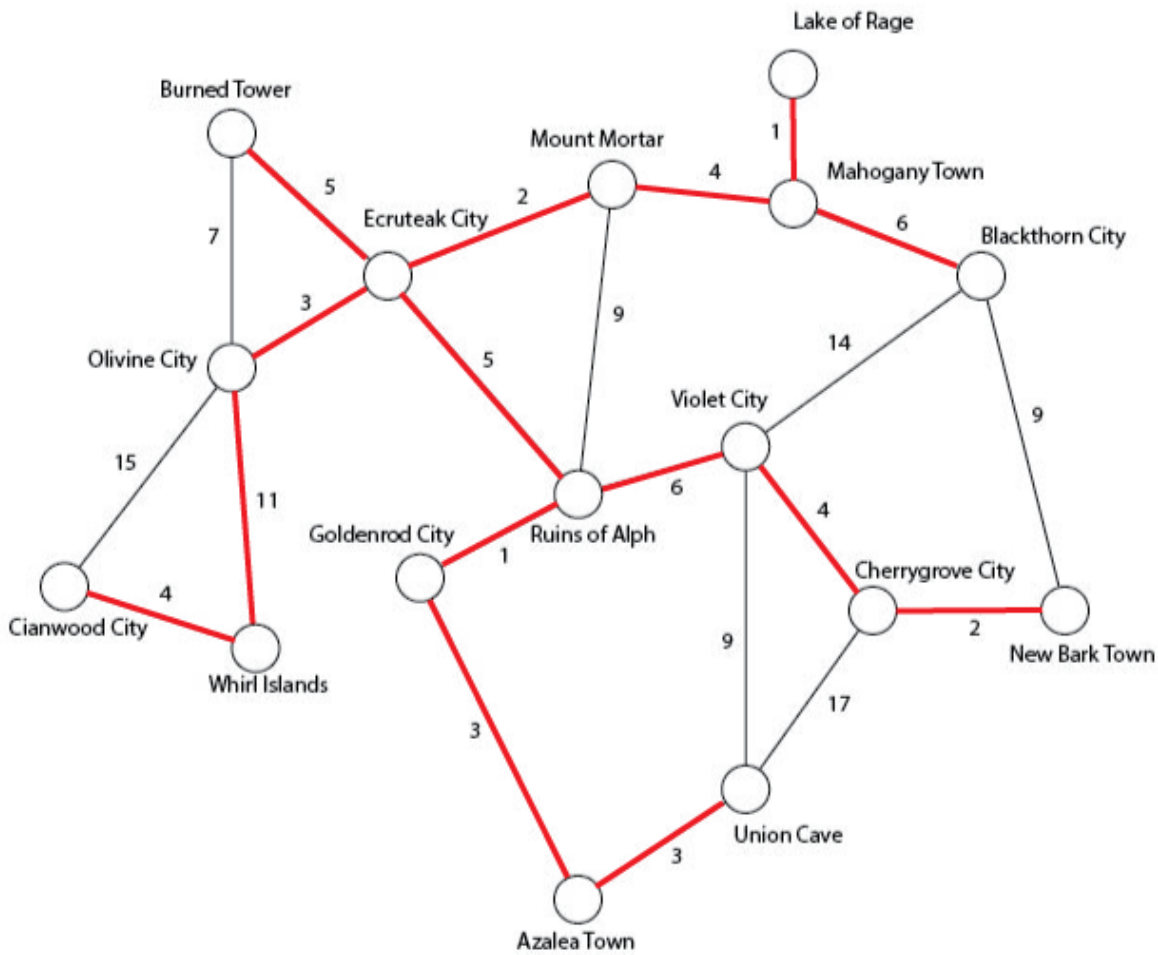
4. Determine whether the following graph is bipartite. If it is, find the groups A and B.
   This graph is bipartite. Group A is the circled vertices. Group B is the uncircled vertices.



5. Colour the regions (including the outer portion) of this map using the fewest colours possible.
   This can be done using 2 colours.



6. Using Prim's Algorithm, find a minimum spanning tree for the following railroad network. Use New Bark Town as your starting vertex.
   The minimum spanning tree found using Prim's Algorithm is highlighted in red



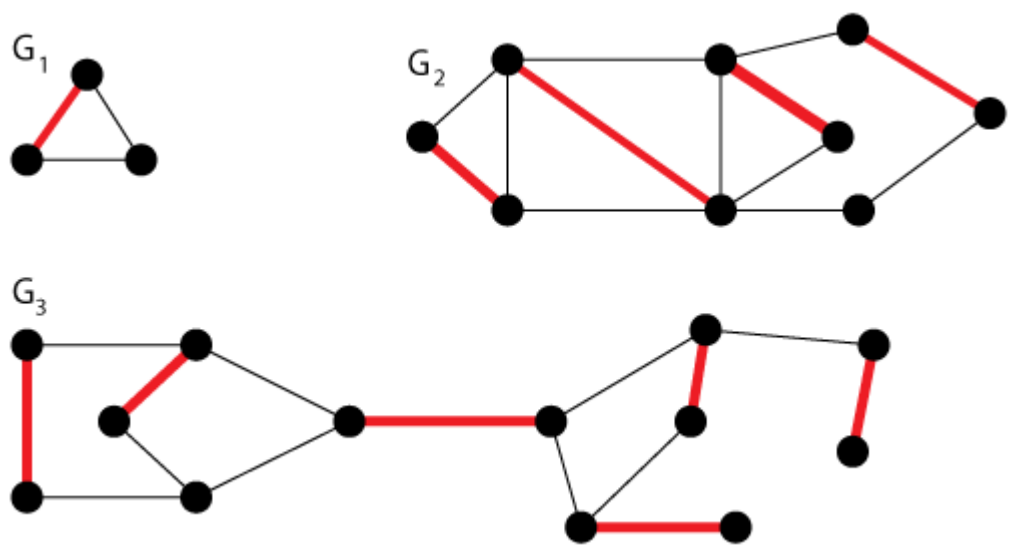
Challenge Problems

Matchings

Let $G = (V, E)$ be a graph. Let M be a subset of the edges of G. M is a **matching** of G if no two edges of M share an endpoint. For example, in the graphs below, the red lines are a matching, since no two red edges share an end point, but the blue edges do not form a matching.

The **size** of a matching is just the number of edges in it. Our goal is often to find a matching of maximum size. That is, we cannot make a larger matching.

1. Find a maximum matching in the following graphs:

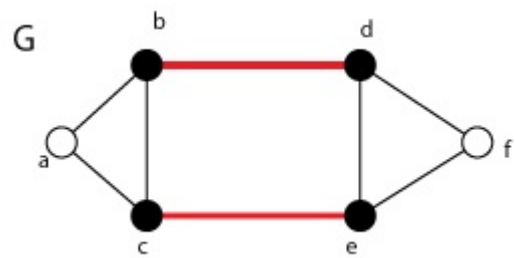


While it may be simple to check that a matching has maximum size on a small graph, what should we do on larger graphs?

To do this we introduce the concept of *saturation*. Given a graph G and a matching M, we call a vertex v **saturated** if it is the endpoint of one of the edges in M. If v is not an endpoint of an edge in M, then we call v **unsaturated**.

We want to know if the matching M is of maximum size. To do this, we want to find an *augmenting path*. A path P is an augmenting path if it begins and ends at unsaturated vertices, and the edges in the path alternate between edges not in the matching, and edges in the matching.

For example, in the graph below the red edges are the matching M. The black vertices are saturated, and the white vertices are unsaturated.
The path $P = \{\{a, b\}, \{b, d\} \{d, f\}\}$ is an augmenting path.



If an augmenting path exists, then M is not a maximum matching.

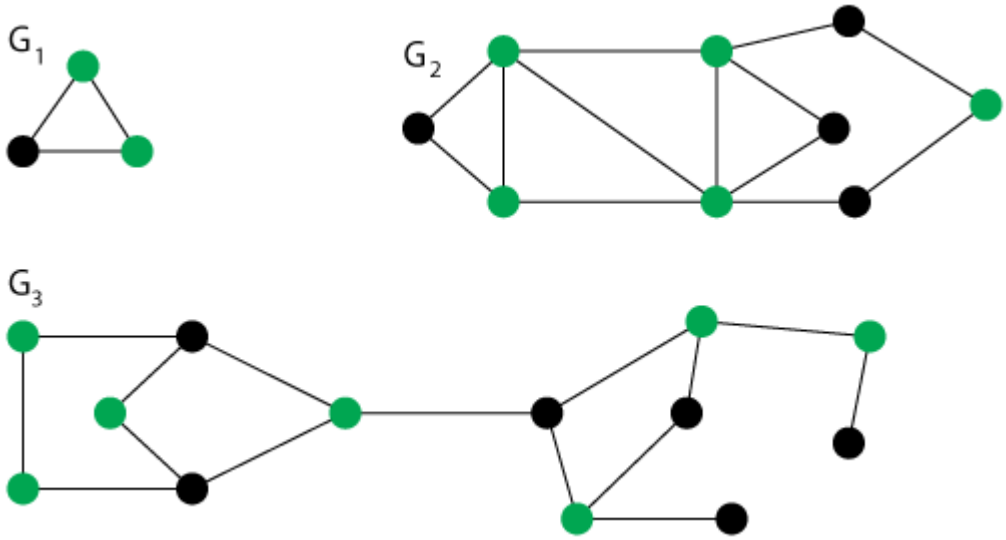
2. Why is M not a maximum matching if an augmenting path exists? How can we create a new, larger matching?
   Since an augmenting path starts at an unsaturated vertex, the first edge in the path must be an edge that is not currently in the matching (otherwise the vertex would be saturated). Also, since our augmenting path ends at an unsaturated vertex, the last edge in our path must also be an edge not currently in the matching. The edges (starting from the second edge) in the augmenting path alternate between edges in the matching and edges not in the matching. This means that if we ignore the first edge in our augmenting path, we have an equal number of edges that are in the matching and edges not in the matching. Now, if we include the first edge we know we have one more edge that is not in the matching.
   We can then make a new matching that is bigger by 'flipping' all of the edges in the augmenting path. That is, any edge in the path that was not in the matching now is in the matching, and any edge that was in the matching is no longer in it.
   Since we had one more non-matching edge originally, when we flip the edges we will have one more matching edge.

3. Using your knowledge of augmenting paths, confirm that your matchings from question 1 are maximium.
   The matchings from above are all maximum matchings.

   Vertex Covers
   A vertex cover is a set C of vertices such that every edge in the graph has an endpoint in C. For example, the green vertices in the graph below are a vertex cover.

Our goal is find a vertex cover with the smallest number of vertices.

4. Find a vertex cover with the smallest number of vertices in the following graphs The green vertices make up the vertex cover



5. What do you notice about the size of the vertex covers you found and the size of the matchings you found earlier? Is one always bigger?
The size of a vertex cover will always be bigger than the size of a matching

6. Explain why the size of a vertex cover will always be larger than the size of a matching.
Suppose we have a matching of size M. We know that a vertex cover needs to have one vertex from each edge in it. Since no edges in this matching share a vertex, then we need at least one vertex from each of the M matching edges. This means that the vertex cover will be at least as big as M.

7. What can we say about the a vertex cover of size V and a matching of size M where $M = V$?
Since we know that any vertex cover is at least as big as the size of a matching, then we know that any vertex cover of size S has $S \geq M$. Since $M = V$ then we know that for any vertex cover of size S, $S \geq V$. This means that V is the size of the minimum vertex cover. Similarly, any matching of size P has $P \leq V$. This means that for any matching of size P, $P \leq M$ and so M is the size of a maximum matching.