1. (10 pts) How many graphs on 4 vertices (numbered 1 through 4) are there? 
   There are infinitely many graphs, if one allows parallel edges, and loops. If not, i.e. if we restrict our attention to \textit{simple} graphs, there are \(2^{13/2} = 2^6 = 64\).

2. (10 pts) How many simple non-isomorphic graphs on the above 4 vertices are there? 
   Assuming connectivity: A counting argument works here: One full graph, 
   one full graph with an edge removed, 2 with 2 edges removed (either adjacent edges or not), and 2 graphs with 3 edges - these are just trees. 
   That’s 6 altogether. If not connected, either a component is a triangle, an L-shape, a segment, or no edges at all, and that gives 10 graphs altogether.

3. (10 pts) How many of the graphs from the previous question are planar? 
   They are all planar - only a graph containing a \(K_5\) or \(K_{33}\) isn’t.

4. (10 pts) Give the definition of a tree. Which trees have an Euler cycle? 
   Prove your statement. 
   A tree is a graph with exactly one path connecting every 2 vertices. A tree is acyclic, hence no Euler cycle.

5. (10 pts) For which values of \(n\) does the full graph \(K_n\) that contain Hamiltonian cycles? 
   \(n=2\) - no cycles. Higher \(n\)’s always have a Hamiltonian cycle - just number the vertices, and go from one to the other in ascending order.

6. (10 pts) What is the adjacency matrix of the full graph \(K_n\)? 
   0’s on the diagonal, 1’s everywhere else.
7. (10 pts) Describe an algorithm to find a spanning tree for a connected graph.
   Just give a description of BFS or DFS - no need for pseudocode.

8. (10 pts) What is the worst case time it takes to determine if two binary trees are isomorphic?
   O(n) - fixed number of operations per vertex.

Here's a proof for the problem from the last class: it turns out the smallest edge from a vertex is contained in every minimal spanning tree (assuming the weight of all other edges from that vertex are of strictly greater weight).

Proof: Let T be an MST, disconnect the vertex by severing all the edges contained in T. Get a number of components, each of which was connected to the vertex by exactly one severed edge (else there was a cycle). But the edge with smallest weight connected the vertex to one of these components, and so using this edge, rather than any other, to reconnect the MST would give the least total weight, hence this edge must be in the MST.