Bill Davis
605.421

1. 21.1-1 Given an adjancy list representation:
   The out degree can be computed in $\mathbf{O}$(E+V), since we need only to iterate through the vertex array and compute the length of each constituent adjacency list.

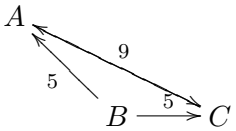
   The in degree can be computed in $\mathbf{O}(V \times E + V)$ time, since we need to traverse the entire adjacency list for each vertex.

2. 22.1-5 We can compute G^2 by following each edge in G, and adding an edge to every edge pointed to by that vertex.
   i.e.
   foreach v $\in$ V
   ..foreach (v,w) $\in$ E[v]
   ....foreach (w, y) $\in$ E[w]
   ......add edge (v,y) to G^2
   This algorithm requires us to traverse the edges foreach edge. This can happen in $\mathbf{O}(E^2)$ time.

   Things are slightly different for an adjacency matrix since we need to interate through the whole array to determine where the edgees are.
   foreach v $\in$ V
   ..foreach (v,w) $\in$ E[v]
   ....if (v,w) $\neq$ 0
   ......foreach (w, y) $\in$ E[w]
   ....if (w,y) $\neq$ 0
   ....add edge (v,y) to G^2

   This algorithm requires us to iterate through the whole array of vertices once for each edge in the array. This can be done in $\mathbf{O}(V^3)$ time.

3. 23.1-6: Show that a graph has a minimum spanning tree if for every cut there is a unique light edge crossing it

As a counter example 

The cut between trees containing (B) and (AC) does not have a unique light edge and this clearly has two minimum spanning trees. Both BAC and BCA satisfy the requirement.

We can show that the exists a spanning by
First selecting an vertex and adding it to a subtree. By assumption there exists a light edge between the subtree containing only this vertex and the rest of the tree.
Add this light edge and the vertex it connects too to the tree. This is now a MST containing these two vertices.
Repeat until all vertices have been added.

This tree is unique since there exists only 1 light edge between any subtree and the remainder of the graph.

4. 22.2-3 Runtime for BFS for an adjacency matrix.
   Since each vertex is enqueued at most 1 time, the total time for queue operations if $\mathbf{O}$(V).
   Also the adjacency matrix has size $V \times V$, therefore the time spent scanning the matrix for edges is $\mathbf{O}(V^2)$, hence the algorithm should run in $\mathbf{O}(V^2 + V)$.

5. 23.2-5 Runtime of Prim's Algorithm if the weights are in the range 1 to W for some constant W.
   If we can assume the weights of all the edges in the tree are in this range, then we can replace the EXTRACT-MIN operation with an array implementation. In this case we can scan for the lowest cost vertex in $\mathbf{O}$(W) and since W is a constant this time is $\mathbf{O}(1)$. Also the decrease key operation simply entails removing one item from the array which can be done in $\mathbf{O}(1)$. This means the whole algorithm has a runtime $\mathbf{O}$(E+V).

   If the weights are in the range 1 to $|V|$ then the above analysis shows that the extract min runtime will be $\mathbf{O}(|V|)$, and the that total runtime will be $\mathbf{O}(|V|^2 + \text{E})$.