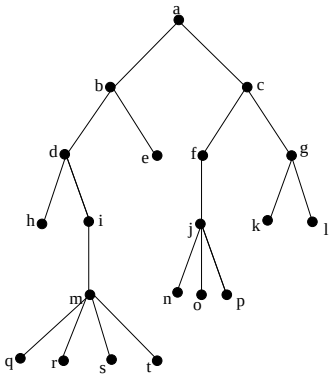
Math 210
Exam 4 - Practice Problem Solutions

1. Answer the following questions based on the rooted tree shown below:



   (a) List the children of vertex j.
   n, o, p

   (b) List the ancestors of vertex s
   m, i, d, b, a

   (c) List the siblings of vertex q
   r, s, t

   (d) Find the number of leaves in this rooted tree.
   $4 + 3 + 3 + 1 = 11$

   (e) List all level 3 vertices in this rooted tree.
   h, i, j, k, l

   (f) Find the least m for which this tree is a rooted m-ary tree.
   $m = 4$ since m has 4 children and no other vertex has more than 4 children.

   (g) Find the height of this rooted tree.
   $h = 5$.

   (h) Find the order that you would visit the vertices of this tree if you use postorder traversal to visit the vertices.
   $h, q, r, s, t, m, i, d, e, b, n, o, p, j, f, k, l, g, c, a$

   (i) Find the order that you would visit the vertices of this tree if you use preorder traversal to visit the vertices.
   $a, b, d, h, i, m, q, r, s, t, e, c, f, j, n, o, p, g, k, l$

   (j) Find the order that you would visit the vertices of this tree if you use inorder traversal to visit the vertices.
   $h, d, q, m, r, s, t, i, b, e, a, n, j, o, p, f, c, k, g, l$

2. A chain letter starts when a person sends a letter to 5 people. Each person who sends the letter to 5 other people who have never received it or does not send it to anyone. Suppose that 10,000 people send out the letter before the chain ends and that no one receives more than one letter. How many people receive the letter? How many people do not send it out?

   Notice that since every person who sends out the letter sends it to exactly 5 other people, and no two people receive the letter twice, this situation can be modeled using a full rooted 5-ary tree. The root represents the person who first sends out the letter, and the children of any vertex represent the 5 people that the related person sent letters. From this, we see that $i = 10,000$.

   The total number of people who received the letter can be found by computing the number total number of vertices in the rooted tree. Using Theorem 4, this is $n = mi + 1 = 5(10,000) + 1 = 50,001$. This counts the root, who started the letter but did not receive it, so 50,000 people received the letter.

   Notice that leaves represent people who did not mail the letter to 5 other people. Thus, again using Theorem 4, $l = (m-1)i + 1 = 4(10,000) + 1 = 40,001$, so 40,001 people did not send it out after they received it.

3. Prove that every tree on n vertices has $n-1$ edges.

   We will proceed by induction on the number of vertices in the graph.

   Base Case: Consider a tree with $n = 1$ vertices. Notice that such a tree must have no edges, since a tree is a simple graph and hence has no loop edges. Then the theorem is true when $n = 1$.

   Inductive Step: Suppose that a tree with k vertices has $k-1$ edges.

   Consider a tree T with $k+1$ vertices. We claim that there must be at least one edge in T that is a pendant edge.

   To see this, consider a maximal path in T. Since a tree has no circuits and the graph T is finite, the final edge in this path must be a pendant edge. Let v be the final vertex in the degree sequence of this path. Let u be the unique vertex adjacent to v. If we remove remove vertex v along with the pendant edge $\{u, v\}$, we obtain a graph T'. We claim this T' is a tree. Notice that since the only edge we removed was a pendant edge, and we also removed the vertex v, then T' is connected. Also note that since edges were removed and no edges new were added, there are no simple circuits in T'.

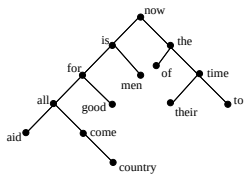
   Then T' is a tree with k vertices. Hence, applying the induction hypothesis to T', this tree must have $k-1$ edges. But then the original tree T has k edges. This proves the theorem. □.

4. Prove that an m-ary tree of height h has at most m^h leaves.

   We will proceed using strong induction on the height of the tree. **Base Case:** Suppose T is a rooted m-ary tree with height 1. Then T consists of a root vertex and children of that root vertex. Since T is m-ary, the root has at most m children, so the graph has at most m leaves.

   Inductive Step: Suppose that any rooted m-ary tree of height $k < h$ has at most m^k leaves. Consider an m-ary tree of height h. As above, the root vertes of T has at most m children. Let $T_1, T_2, ..., T_l$ be the subtrees of T rooted at the level 1 children of the root vertex. Then $l \leq m$, and each T_i, $1 \leq i \leq l \leq m$ can be thought of as a rooted tree with height at most $h-1$.
   Using the inductive hypothesis on each T_i, each of these subtrees has at most m^{h-1} leaves. However, each leaf of T is a leaf of one of these subtrees, so L, the number of leaves of T satisfies the inequality $L \leq \sum_{1}^{l} m^{h-i} \leq m \cdot m^{h-1} = m^h$.
   This proves the theorem. □.

5. (a) Draw a binary search tree for the sentence "Now is the time for all good men to come to the aid of their country."



   (b) How many comparisons are needed to locate the word *time* in this tree?
   3

   (c) How many comparisons are needed to add the word *waffle* to this tree?
   4

6. How many weighings are needed to find a counterfeit coins among 8 total coins if the counterfeit could be either heavier or lighter than a normal coin. Give an algorithm that proves your answer.

   The counterfeit coin can be found in three weighings. First, use the scale to compare coins 1 and 2 on one side against coins 3 and 4 on the other side.

   There are two cases:

   Case 1: Suppose the two sides do not balance. Then the counterfeit must be one of these four coins, and the remaining four are all genuine. Therefore, to find the counterfeit, we weigh coins 1 and 2 on one side against coins 5 and 6 on the other side.

Subcase 1a: If coins 1 and 2 and coins 5 and 6 balance, then the counterfeit is either 3 or 4. To finish, we weigh coin 3 against coin 5 (which we know is genuine). If 3 and 5 weigh the same, then the counterfeit is coin 4. Otherwise, 3 must be the counterfeit.

Subcase 1b: If coins 1 and 2 and coins 5 and 6 do not balance, then the counterfeit is either 1 or 2. To finish, we weigh coin 1 against coin 5 (which we know is genuine). If 1 and 5 weigh the same, then the counterfeit is coin 2. Otherwise, 1 must be the counterfeit.

Case 2: Suppose the two sides do balance. Then the counterfeit must be one of the other four coins, and the first four are all genuine. Therefore, to find the counterfeit, we weigh coins 1 and 2 on one side against coins 5 and 6 on the other side.

Subcase 2a: If coins 1 and 2 and coins 5 and 6 balance, then the counterfeit is either 7 or 8. To finish, we weigh coin 1 (which we know is genuine) against coin 7. If 1 and 7 weigh the same, then the counterfeit is coin 8. Otherwise, 7 must be the counterfeit.

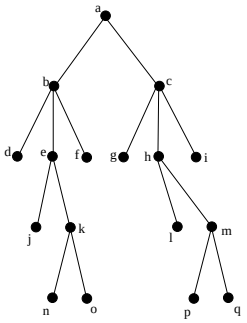
Subcase 2b: If coins 1 and 2 and coins 5 and 6 do not balance, then the counterfeit is either 5 or 6. To finish, we weigh coin 1 (which we know is genuine) against coin 5. If 1 and 5 weigh the same, then the counterfeit is coin 6. Otherwise, 5 must be the counterfeit.

7. How many weighings are needed to find *two* counterfeit coins among 5 total coins, one heavier than a normal coin and the other lighter than a normal coin. Give an algorithm that proves your answer.

   Note that it is possible (althought not necessary) that the heavy and light coin to combine to weigh the same as 2 normal coins.

   *** I'll add the algorithm and decision tree for this later if I have time ***

8. Consider the following tree:



   (a) List the vertices in the order that you would visit them if you traversed the tree in preorder.

   $a, b, d, e, j, k, n, o, f, c, g, h, l, m, p, q, i$

   (b) List the vertices in the order that you would visit them if you traversed the tree in inorder.

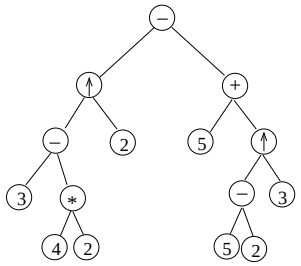
   $d, b, j, e, n, k, o, f, a, g, c, l, h, p, m, q, i$

   (c) List the vertices in the order that you would visit them if you traversed the tree in postorder.

   $d, j, n, o, k, e, f, b, g, l, p, q, m, h, i, c, a$

9. Consider the expression: $(3 - 4 \cdot 2)^2 - (5 + (5 - 2)^3)$

   (a) Draw the rooted tree representing this computation.



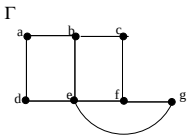
(b) Write this expression in prefix notation.

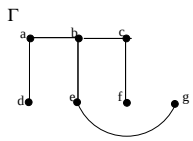
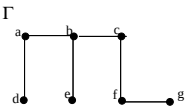
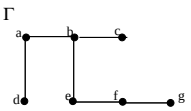
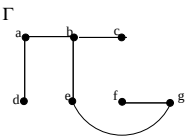
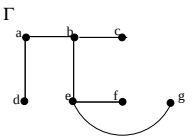
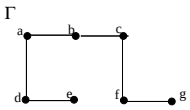
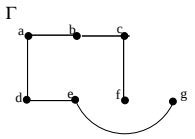
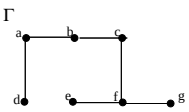
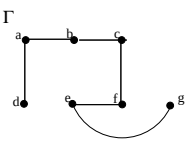
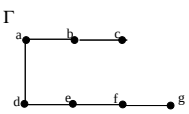
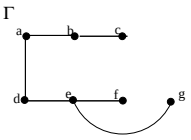
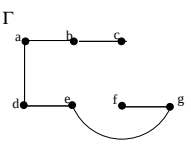
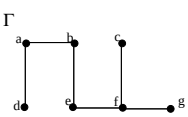
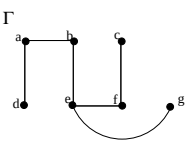
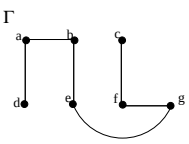
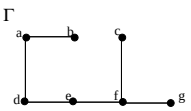
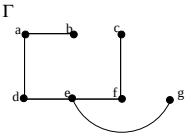
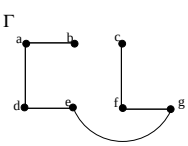
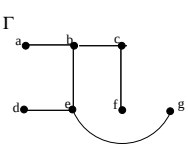
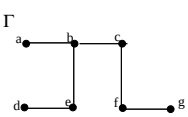
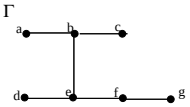
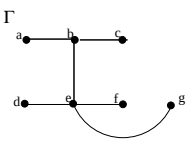
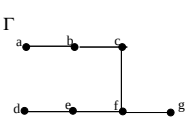
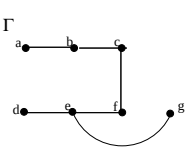
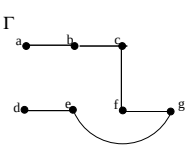
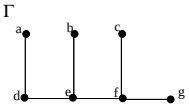
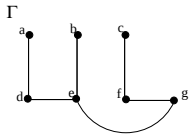
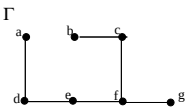
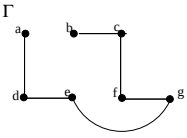
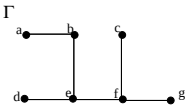
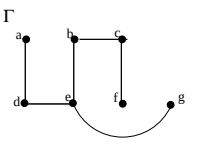
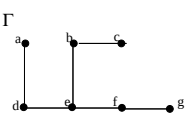
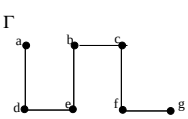
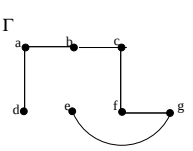
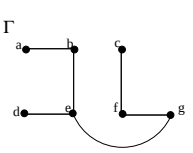
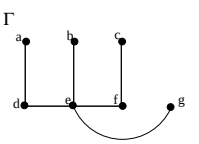
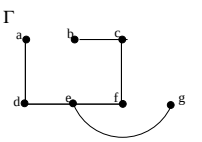
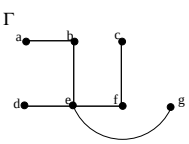
$- \uparrow -3 * 422 + 5 \uparrow -523$

(c) Write this expression in postfix notation.

$342 * -2 \uparrow 552 - 3 \uparrow +-$

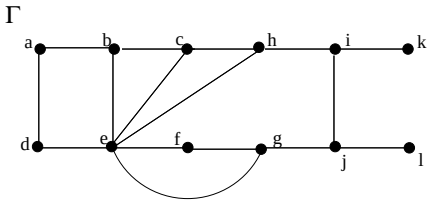
10. Draw all possible spanning trees for the following graph:



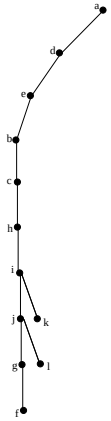
1. 
2. 
3. 
4. 
5. 
6. 
7. 
8. 
9. 
10. 
11. 
12. 
13. 
14. 
15. 
16. 
17. 
18. 
19. 
20. 
21. 
22. 
23. 
24. 
25. 
26. 
27. 
28. 
29. 
30. 
31. 
32. 
33. 
34. 
35. 
36. 
37. 
38. 

[I think that this is a complete list – if you see one I missed, let me know. The first person to point out each missing spanning treee gets an extra credit point]

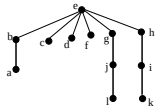
11. Given the following graph:



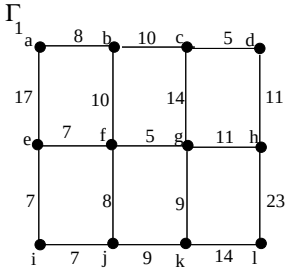
(a) Use a depth first search to find a spanning tree for this graph starting at root vertex a.

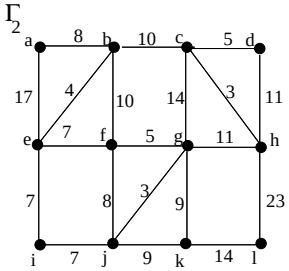


(b) Use a breadth first search to find a spanning tree for this graph starting at root vertex e

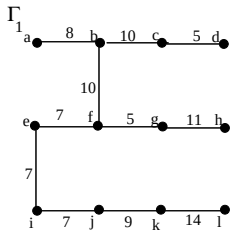


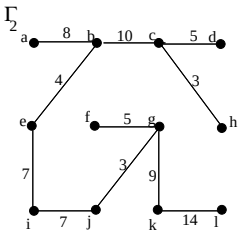
12. For the weighted graphs given below:





(a) Use Prim's Algorithm to find a minimum spanning tree for each graph.

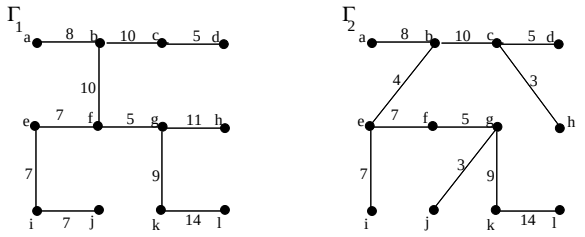




For Γ_1, the edges are added in the following order:
$\{f,g\},\{e,f\},\{e,i\},\{i,j\},\{j,k\},\{b,f\},\{a,b\},\{b,c\},\{c,d\},\{g,h\},\{k,l\}$

For Γ_2, the edges are added in the following order:
$\{j,g\},\{f,g\},\{i,j\},\{e,i\},\{e,b\},\{a,b\},\{g,k\},\{b,c\},\{c,h\},\{c,d\},\{k,l\}$

(b) Use Kruskal's Algorithm to find a minimum spanning tree for each graph.



For Γ_1, the edges are added in the following order:
$\{f,g\},\{c,d\},\{e,f\},\{e,i\},\{i,j\},\{a,b\},\{g,k\},\{b,f\},\{b,c\},\{g,h\},\{k,l\}$

For Γ_2, the edges are added in the following order:
$\{j,g\},\{c,h\},\{b,e\},\{f,g\},\{c,d\},\{e,f\},\{e,i\},\{a,b\},\{g,k\},\{b,c\},\{k,l\}$

13. Describe an algorithm to determine whether or not a simple connected graph has a Hamilton Circuit using depth first searches.

Start by putting a total order on the vertices of the simple connected graph Γ; $\{v_1, v_2, ..., v_n\}$.

Next, begin carrying out a depth first search for a spanning tree starting at v_1. Whenever you have a choice of which vertex to add next, choose the vertex that appears earliest on the ordered list of vertices we chose above. When you can no longer add any more vertices to your initial path, if you have visited every vertex, check to see if you can return to v_1 using an edge that has not already been used.

If you can, the path you found is a Hamilton Path, and the edge taking you back to v_1 can be added to the path to form a Hamilton Circuit.

Otherwise, if there is a vertex that has not yet been visited or if the end vertex is not adjacent to v_1 via an unused edge, then the current path cannot be extended to a Hamilton Circuit.

Next, go back and carry out a new depth first search starting at v_1 again, but the first time you have a choice between more than one vertex to add to the path, choose the earliest vertex at that step not chosen in a previous point in the construction. Repeat until you either construct a Hamilton Circuit or until all possible ways of carrying out a depth first search have been exhausted.

Note: we can be certain that this algorithm will produce a Hamilton Circuit whenever one exists, since if Γ has a Hamilton Circuit, if we delete one of the edges in the circuit that are adjacent to v_1, the result is a path graph spanning tree of Γ beginning at v_1.