MATH 239

TUTORIAL 9

Question 1:
A forest is a graph with no cycles. Prove that a forest with p vertices and q edges has $p - q$ components.

Answer 1:
Let $G = (V, E)$ be a forest with k components. Since each edge e is not in a cycle, it is a bridge. Hence, $G \backslash e$ has exactly 1 more component than G. By removing all the edges one by one, $G \backslash E$ has $k + q = p$ components. Therefore, we have $k = p - q$ as desired.

Question 2:
Prove that if a graph G has a closed walk of odd length, then it must have an odd cycle.

Answer 2:
Let $C = v_1 v_2 \cdots v_n v_1$ be a shortest closed walk of odd length. Suppose for contradiction it is not a cycle. Then, $v_i = v_j$ for some $i < j$. Hence, $v_i v_{i+1} \cdots v_j = v_i$ is a closed walk of length $j - i$. On the other hand, the walk $v_1 v_2 \cdots v_i = v_j \cdots v_n v_1$ is a closed walk of length $i + n - j$. Since n is odd, both of these numbers cannot be even as they sum to n. Therefore, one of them is an odd closed walk of shorter length. Contradiction.

Question 3:
Show that for each edge $e \in E(G)$ for a connected graph G, there is a spanning tree T_e that contains the edge e.

Answer 3:
Let $e = uv$, and let T_e be the tree you get when you run BFS on u. As BFS will immediately pick up all neighbours of u, e is contained in T_e.

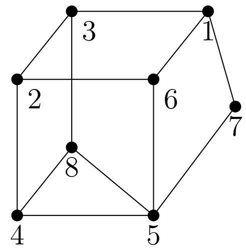
Alternative Answer:

Let T_0 be any spanning tree of G. If T_0 contains e, we are done. Otherwise, let $e = uv$. Now, there must be a path P from u to v in T_0, since it is a tree. Therefore, $C = e + P$ is a cycle. Let $e' = xy$ be an edge in P, and consider the graph $T_e = (T \cup \{e\}) - e'$. Since e' is in a cycle of $T \cup \{e\}$, e' is not a bridge. Therefore, T_e must be connected. Now, $|E(T_e)| = |E(T_0)| = |V(G)| - 1$, since all spanning trees of G with $|V(G)|$ vertices has $|V(G)| - 1$ edges. As T_e is connected and has $|V(G)| - 1$ edges, it must be a spanning tree.

Question 4:
A search algorithm that is similar, but not the same as the one taught in the course is called the Depth First Search. To do this search, change algorithm 5.4.2 of your book as follows: At each stage consider the unexhausted vertex u that joined the tree *latest* among all unexhausted vertices, and choose an edge incident with this vertex and a vertex v not in the tree.

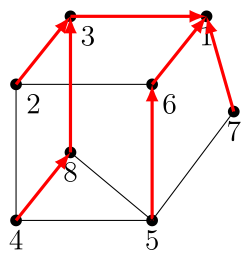
Run the BFS and the DFS on the following graph, using vertex 1 as the root, and add vertices to the tree using the smallest label first. Are the BFS and DFS trees the same?

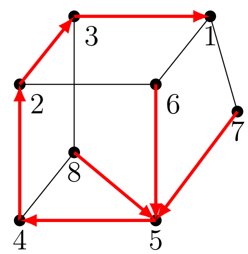
Date: 10/26/12.



Answer 4:
Aside: For those of you in Computing Science, the difference between the BFS and DFS is the difference between implementing the algorithm using a queue as compared to a stack. By varying the data structure used, one can even get a minimum spanning tree, or a tree with shortest path information, when you assign a weight on the edges.

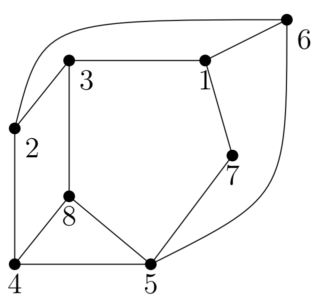
The BFS and DFS of the above graph is given by below. The order of which the vertices are added to the tree is 13672854 and 13245678 respectively. It can be easily seen that they are not equal.





Question 5:
Provide a planar embedding of the graph in Question 4.

Answer 5:



Question 6:
Let m, n be integers with $m, n \geq 1$. Let G be a graph with m vertices of distance n from a given vertex v. Prove that G has a spanning tree with at least m vertices of degree 1.

Answer 6:
The BFS starting from v satisfies this property. For each vertex u that is distance n from v, either u is a leaf, there exist a vertex w such that w is a leaf, and u is an ancestor of w. If u is not a leaf, then there must be some vertices $w_1, \ldots, w_t$ that has u as its ancestor. Take w such that it is furthest from v. Then, w must have no children, so it must be a leaf. Furthermore, as $w, pr(w), pr^2(w), \ldots, v$ are vertices of different distance from v, there cannot be two vertices of distance n that are both ancestors of w. This gives a implies that we have a leaf corresponding to each vertex of distance n, which proves that the BFS starting from v has at least m leaves.