Name:	GSI:	DISC #:

Solutions to Homework 13.

Prob 1. Suppose that $G = (V, E)$ is a bipartite graph with bipartition (V_1, V_2) and that $A \subseteq V_1$. Show that the maximum number of vertices of V_1 that are the endpoints of a matching of G equals $|V_1| - \max_{A \subseteq V_1} \mathrm{def}(A)$ where $\mathrm{def}(A) = |A| - |N(A)|$ is called the **deficiency** of A.

Solution. First note the the empy set has deficiency zero, so the maximum deficiency is always nonnegative (although there may not be a non-empty subset of V_1 with positive deficience; that's entirely possible).

Case 1. If the maximal deficiency is zero, then the deficiency of every non-empty subset $A \subseteq V_1$ is nonpositive, i.e., $|A| \leq |N(A)|$. By Hall's marriage theorm, there is a complete matching from V_1 to V_2. All vertices of V_1 are the endpoints of that matching. And we cannot more than $|V_1|$ of edges in any matching.

Case 2. Otherwise the maximal deficiency k is positive. Add k new vertices to the set V_2 and connect them to all vertices of V_1. In the resulting new bipartite graph H, subsets of V_1 have maximal deficiency zero. Indeed, the deficiency of any non-empty subset A in H is its deficiency in G minus k, so

$$\mathrm{def}_H(A) = \mathrm{def}_G(A) - k \leq k - k = 0.$$

Hence the new graph H satisfies the condition of Case 1. By the conclusion in Case 1, H therefore contains a complete matching from V_1. Remove k vertices of V_1 that are matched to the added vertices of H, and obtain a matching in G with $|V_1| - k$ edges.

Finally, note that we cannot have more than $|V_1| - k$ edges in any matching of G because then there would be a matching of H with more than $|V_1|$ edges obtained by matching each of the added k vertices with the k vertices in V_1 that are left out. But the number of edges in a matching of a bipartite graph cannot exceed the size of a part in its bipartition. This concludes the proof.

Prob 2. The Good Will Hunting Problem.

(a) How many trees are there with n distinguishable/labeled vertices?

Solution. Let us denote the number of (undirected) trees with n labeled vertices by T_n. We will count slightly different objects in two ways to solve this problem. These other objects are rooted and labeled trees with n vertices. In other words, their vertices are labeled and one of the vertices is further distinguished as a 'root'. Moreover, we will also make all edges directed away from the root, and listed in some order.

We can count such rooted, directed-away-from-the-root, edge-ordered trees (let us call them **fancy trees**) in two ways. First, each undirected tree can be turned into a fancy tree by

1. choosing a root, which can be done in n ways;
2. orienting each edge; this choice is uniquely determined by the root;
3. listing all edges in some order, which can be done in $(n-1)!$ ways since there are $n-1$ edges.

This gives us $T_n \cdot n \cdot (n-1)! = n! \cdot T_n$ fancy trees.

On the other hand, a fancy tree can be formed from the graph of n isolated vertices by adding one edge at a time (and recording that addition order as the order on the edges). At stage k, we will have a forest of $n-k$ rooted trees. A new edge can be added from any vertex v of the graph to any vertex u not belonging to the connected component of v. That edge is directed away from v, and the process continues. This gives

$$\prod_{k=1}^{n-1} (n(n-k)) = n^{n-1}(n-1)!$$

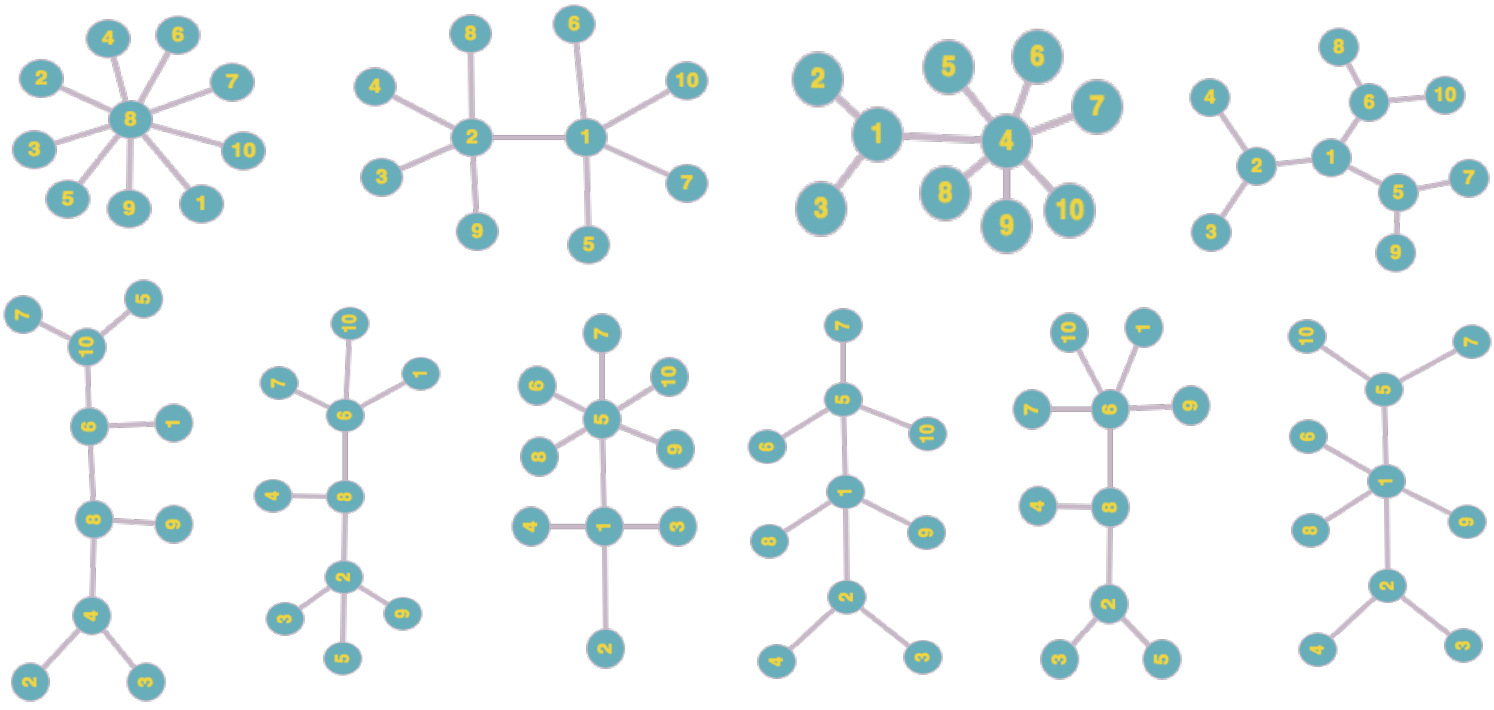
fancy trees. So $n!T_n = (n-1)!n^{n-1}$, hence $T_n = n^{n-2}$.
Answer: n^{n-2}. This marvellous proof is due to Jim Pitman.

(b) A homeomorphically irreducible tree is one where each vertex has either one or at least three edges incident with it. Draw all homeomorphically irreducible trees with 10 vertices.

Solution. By the Handshake Theorem, the sum of all total degrees must be equal to $2 \cdot (10-1) = 18$ since there are exactly 9 edges in any tree on 10 vertices by the result of Prob. 1. Moreover, no degree is allowed to be equal to 2. So we have seven degree sequences available corresponding to these partitions of 18:

$$9+9\cdot 1,\ \ 7+3+8\cdot 1,\ \ 6+4+8\cdot 1,\ \ 2\cdot 5+8\cdot 1,\ \ 5+2\cdot 3+7\cdot 1,\ \ 2\cdot 4+3+7\cdot 1,\ \ 4\cdot 3+6\cdot 1.$$

They gives us the following 10 graphs. Drawing courtesy of Alex Peterson:



Prob 3. Fleury's algorithm, published in 1883, constructs Eulerian circuits by first choosing an arbitrary vertex of a connected multigraph and then forming a circuit by choosing edges successfully. Once an edge is chosen, it is removed. Edges are chosen successively so that each edge begins where the last edge ends and so that this edge is not a cut edge unless there is no alternative.

Prove that Fleury's algorithm always produces an Eulerian circuit.

Proof: For an Eulerian circuit to exist, the necessary condition that all degrees are even needs to be satified. Assuming this condition, we need to show that Fleury's algorithm always produces an Eulerian circuit.

Starting at an arbitrary vertex u of the graph, run Fleury's algorithm until you stop with no edges available to you for further steps. First we can see that we can only stop at u. Indeed, once we leave u and remove the first traversed edge, the degree of u becomes odd. When we arrive at any other vertex, its degree becomes odd, and when we pass it, it becomes even again, and altogether it decreases by 2. The degree of a terminal vertex must turn even when we arrive there. So it can only be u.

We now need to show that we have used all edges in our multigraph when the algorithm halts. Suppose not, i.e., suppose that there are some edges still left and not removed. The subgraph remaining is certainly disconnected because u has no more edges left incident to it, i.e., its degree is now zero. So, at a minimum, u is now disconnected from the rest of the graph.

Of all the edges left, consider their endpoints. At least one of them was passed by Fleury's algorithm because initially all other vertices were connected to u and now they are not. Take the one such endpoint v that was visited *last* by Fleury's algorithm.

That vertex v still has a positive (and even) degree. Moreover, its remaining connected component still satisfies Euler's condition that all its degrees are even. So, if we were to run Fleury's algorithm starting at v on its remaining connected component, we would come back to v.

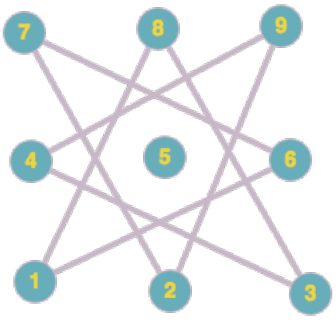
When we last arrived at v, the edge we picked to traverse (and then remove) disconnected v from u as a result. Hence it was a cut edge. So, by the Fleury condition, there were no non-cut edges available. But by our observation, one can still walk from v using the remaining edges and come back to v. So at least some remaining edges were not cut edges when we last visited v!

This contradiction shows no such endpoints, and hence no edges are left after Fleury's algorithm halts. In other words, Fleury's algorithm finds an Eulerian circuit.

Prob 4. A **knight's tour** is a sequence of legal moves by a knight starting at some square of a chessboard and visiting each square exactly once. A knight's tour is called **reentrant** if there is a legal move that takes the knight from the last square of the tour to its first. One can model knight's tours via the graph that has a vertex for each square, with an edge connecting two vertices iff a knight can legally move between them.

(a) Draw the graph that represents the legal moves of a knight on a 3×3 chessboard.

Solution. Here is the graph, courtesy of Alex Peterson again!



(b) Prove that finding a knight's tour on an $m \times n$ chessboard is equivalent to finding a Hamiltonian path on the graph representing the legal moves of a knight on that board.

Proof. The knight's tour is defined by visiting every square exactly once, which translates into visiting every vertex of the graph exctly once. Since the legal moves of a knight translate into traversing an edge of the graph, finding a knight's tour is equivalent to finding a Hamiltonian path on the corresponding graph.

(c) Prove that finding a reentrant knight's tour on an $m \times n$ chessboard is equivalent to finding a Hamiltonian circuit on the graph representing the legal moves of a knight on that board.

Proof. Removing any edge from a Hamiltonian circuit on the graph gives us a Hamiltonian path whose initial and terminal vertices are adjacent in the graph, i.e., the corresponding squares have a legal knight's move between them. Conversely, a reentrant knight's tour represents a Hamiltonian path on the graph that is missing only one edge to be completed to a Hamiltonian circuit, which can be done since that edge is in the graph and available, i.e., is not used by the Hamiltonian path. So finding a reentrant knight's tour is equivalent to finding a Hamiltonian circuit on the graph.

(d) Show that there is no knight's tour on a 3×3 chessboard.

Proof. The graph in part (a) is not connected: it has one isolated vertex. So it is impossible to find a Hamiltonian path since either that vertex or the rest of the graph cannot be visited by the same path.