1. Apply Ford-Fulkerson’s Maximum Flow Algorithm to find the Max Flow in the network shown below. Using diagrams show all the intermediate steps and the corresponding residual networks. You are not required to give the algorithm, but all the steps and residual networks should be correct.  

   ![Network Diagram]

   20 Points

2. Let $A = a_1a_2...a_n$ and $B = b_1b_2...b_m$ be two strings of characters that come from a finite set (for example, the English alphabet set). The objective is to change $A$ character by character such that it becomes equal to $B$ – called the ‘string-edit’ problem. We allow three types of changes or edit steps and assign a cost of ‘1’ to each. The edit steps are: i) insert – insert a character into the string; ii) delete – delete a character from the string, and iii) replace – replace one character with a different character. The number of edits performed is called the ‘edit distance’ between $A$ and $B$. Provide an efficient algorithm to find the Minimum edit distance between two strings. Briefly describe one application of the ‘Edit Distance’ algorithm.  

   40 points

3. The edge connectivity of an undirected graph is the minimum number $k$ of edges that must be removed to disconnect the graph. For example, the edge connectivity of a tree is 1, and the edge connectivity of a cyclic chain of vertices is 2. Show how the edge connectivity of an undirected graph $G = (V,E)$ can be determined by running a maximum-flow algorithm on at most $|V|$ flow networks, each having $O(V)$ vertices and $O(E)$ edges.  

   40 points

4. Given a set of $n$ points in the plane $P$, write an algorithm to compute a pair of closest points. A straightforward or brute force algorithm will take $O(n^2)$ but will fetch you 20 points. You get 40 points for an $O(n \log n)$ algorithm.

Additional Sample Questions for Quiz 3 (2010)
Please note these are sample questions only.

1. Let $P$ be a simple (not necessarily convex) polygon enclosed in a given rectangle $R$, and $q$ be an arbitrary point inside $R$. Design an efficient algorithm to find a line segment connecting $q$ to any point outside $R$ such that the number of edge of $P$ that this line intersects is minimum.

2. Let $P$ be a set of $n$ points in a plane. We define the depth of a point $p$ in $P$ as the number of convex hulls that need to be ‘peeled’ (removed) for $p$ to become a vertex of the convex hull. Design an $O(n^2)$ algorithm to find the depths of all points in $P$.

3. Given a set of $n$ points in the plane $P$. A straight forward or brute force algorithm will take $O(n^2)$ to compute a pair of closest points. You get a bonus if you can give an $O(n \log n)$ algorithm.

4. The input is two strings of characters $A = a_1 a_2 \ldots a_n$, and $B = b_1 b_2 \ldots b_n$. Design an $O(n)$ algorithm to determine whether $B$ is a cyclic shift of $A$. In other words, the algorithm should determine whether there exists an index $k$, $1 \leq k \leq n$ such that $a_i = b_{(k+i) \mod n}$ for all $i$, $1 \leq i \leq n$.

5. The largest common subsequence (LCS) of two sequences $T$ and $P$ is the largest sequence $L$ such that $L$ is a subsequence of both $T$ and $P$. The smallest common supersequence (SCS) of two sequences $T$ and $P$ is the smallest sequence $L$ such that both $T$ and $P$ are subsequences of $L$.
   a. Design efficient algorithms to find the LCS and SCS of two given sequences.
   b. Let $d(T,P)$ be the smallest edit distance between $T$ and $P$ such that no replacements are allowed (in other words, we have to insert or delete). Prove that $d(T,P) = |SCS(T,P)| - |LCS(T,P)|$, where $|SCS(T,P)|$ is the size of the smallest SCS of $T$ and $P$ and $|LCS(T,P)|$ is the smallest LCS of $T$ and $P$.

6. Write a program to extend the Rabin-Karp method to handle the problem of looking for a given $m \times m$ pattern in an $n \times n$ array of characters. (The pattern may be shifted vertically or horizontally, but it may not be rotated).