Collaboration is encouraged, but students MUST write up solutions independently and list those with whom they worked. No materials from prior years’ classes or the Internet can be consulted.

1. (20 pts: 10,10) A connected graph is vertex biconnected if there is no vertex whose removal disconnects the graph. A connected graph is edge biconnected if there is no edge whose removal disconnects the graph. Give a proof or counterexample for each of the following statements:

(a) A vertex biconnected graph with more than one edge is edge biconnected.
(b) An edge biconnected graph is vertex biconnected.

2. (20 pts: 5,5,5,5) Prove that in a breadth-first search of a directed graph, where $d[u]$ is the depth of node $u$, the following properties hold:

(a) There are no forward edges.
(b) For each tree edge $(u, v)$, we have $d[v] = d[u] + 1$.
(c) For each cross edge $(u, v)$, we have $d[v] \leq d[u] + 1$.
(d) For each back edge $(u, v)$, we have $0 \leq d[v] \leq d[u]$.

3. (20 pts) We are given a directed graph $G = (V, E)$ on which each edge $(u, v) \in E$ has an associated value $r(u, v)$, which is a real number in the range $0 \leq r(u, v) \leq 1$ that represents the reliability of a communication channel from vertex $u$ to vertex $v$. We interpret $r(u, v)$ as the probability that the channel from $u$ to $v$ will not fail, and we assume that these probabilities are independent. Give pseudocode for an efficient algorithm to find the most reliable path between two given vertices.

4. (20 pts: 6,7,7) A bottleneck spanning tree $T$ of an undirected graph $G$ is a spanning tree of $G$ whose largest edge weight is minimum over all spanning trees of $G$. We say that the value of the bottleneck spanning tree is the weight of the maximum-weight edge in $T$.

(a) Argue that a minimum spanning tree is a bottleneck spanning tree.
(b) Give a linear-time algorithm ($O(V)$, $O(E)$, or $O(V + E)$) that, given a graph $G$ and integer $b$, determines whether the value of the bottleneck spanning tree is at most $b$.
(c) Use your algorithm from the previous part as a subroutine in a linear-time algorithm for the bottleneck-spanning-tree problem.

5. (20 pts: 10,10) Let $G = (V, E)$ be a flow network with source $s$, sink $t$ and integer capacities. Suppose that we are given a maximum flow in $G$.

(a) Suppose that the capacity of a single edge $(u, v) \in E$ is increased by 1. Give a $O(V + E)$ time algorithm to update the maximum flow.
(b) Suppose that the capacity of a single edge $(u, v) \in E$ is decreased by 1. Give a $O(V + E)$ time algorithm to update the maximum flow.