Code Jam 2012 - Round 1C

Analysis: Diamond Inheritance

We are given a directed graph, and have to determine if there is a pair of nodes (X,Y) such that there are two or more paths from X to Y.

For each node, we do a depth-first search with that node as the root. If during the depth-first search we reach the same node twice, then we must have followed two different paths to get to that node from the root node, so we have found a pair (X,Y). Conversely, if there are two paths between X and Y, we will reach Y at least twice when doing a DFS from X. So if this algorithm finds no pair (X,Y), then none exists in the graph.

If there are V nodes and E edges in a graph, then a DFS is O(V+E) in general. But each DFS will never follow more than V edges, because after following that many edges, some node will have been reached twice, so we can stop at that point. Therefore this algorithm is $O(V^2)$.

We can also just use a variation of Floyd's algorithm, which is $O(V^3)$, but very simple to write. With only 1000 nodes in the graph, a fast implementation will finish within the time limit.