Problem Set 8

Problem 1 The BFS (Breadth-First Search) algorithm given in the lecture notes uses multiple lists. Modify the algorithm so that it uses only one queue to replace multiple lists.

Problem 2 Describe, in pseudo code, an O(n+m)-time algorithm for computing all the connected components of an undirected graph G with n vertices and m edges.

Problem 3 Given an undirected graph G and a vertex v_i, describe an algorithm for finding the shortest paths from v_i to all other vertices. The shortest path from a vertex v_s to a vertex v_t is a path from a vertex v_s to a vertex v_t with the minimum number of edges. What is the running time of your algorithm?

Problem 4 A connected undirected graph is said to be biconnected if it contains no vertex whose removal would divide G into two or more connected components. Give an O(n+m)-time algorithm for adding at most n edges to a connected graph G, with n>3 vertices and m>n-1 edges, to guarantee that G is biconnected.

Problem 5 An n-vertex directed acyclic graph G is **compact** if there is some way of numbering the vertices of G with the integers from 0 to n-1 such that G contains the edge (i, j) if and only if i<j, for all i, j in [0, n-1]. Give an $O(n^2)$-time algorithm for detecting if G is compact.

Problem 6 An Euler tour of a directed graph G with n vertices and m edges is a cycle that traverses each edge of G exactly once according to its direction. Such a tour always exists if G is connected and the in-degree equals to the out-degree for each vertex in G. Describe an O(n+m)-time algorithm for finding an Euler tour of such a directed graph.

Problem 7 An independent set of an undirected graph G=(V, E) is a subset I of V such that no two vertices in I are adjacent. That is, if u and v are in I, then (u, v) is not in E. A maximal independent set is an independent set such that if any additional vertex is added to it, it will not be an independent set. Give an efficient algorithm for finding a maximal independent set for an undirected graph, and analyse its time complexity.