CompSci 163/265, Spring 2016, Homework 1

1. A cube has eight vertices and twelve edges. One way to represent this as a graph is to number the vertices from 0 to 7, and to connect two vertices by an undirected edge whenever their numbers differ by a power of two. Write down a representation for this graph (with these vertex numbers) using van Rossum's dictionary-of-lists adjacency list format.

2. In depth-first search, the order in which we loop over the outgoing edges from each vertex is arbitrary: if a vertex has k outgoing edges, then there are $k!$ different choices for what order to loop over them, and this may affect what happens in the rest of the algorithm. In particular, one graph may have more than one possible depth first search tree, depending on this choice of loop order. For the same graph of the cube from problem 1, find the following two different depth first search trees, both starting from vertex 0:

    a. a depth first search tree in which every vertex has at most one child.

    b. a depth first search tree in which at least one vertex has more than one child.

3. (163 students:) List the vertices of the cube (with the same numbering) in an order that could be generated by performing a breadth-first search of the cube, starting with vertex 0.

   (265 students:) List the vertices of the cube (with the same numbering) in an order that is sorted by distance from vertex 0, but could not be generated by performing a breadth-first search of the cube.

4. (163 students:) Define a graph to be "weakly connected" if, for every two vertices v and w, there exists either a path from v to w or a path from w to v (but not necessarily both). Draw a directed graph G that obeys all of the following properties:

    - G is weakly connected.
    - G has at least three strongly connected components.
    - At least one of the strongly connected components of G has only one vertex, and.
    - At least one of the strongly connected components of G has three or more vertices.

   (265 students:) Define an edge of a directed graph to be "loopy" if there exists at least one cycle in the graph that contains the given edge. Describe a linear-time algorithm that lists all of the loopy edges of a directed graph. (Hint: use the strongly connected component algorithm as a subroutine. You do not need to describe the details of this subroutine. You may assume that it outputs each component as a list of vertices.)