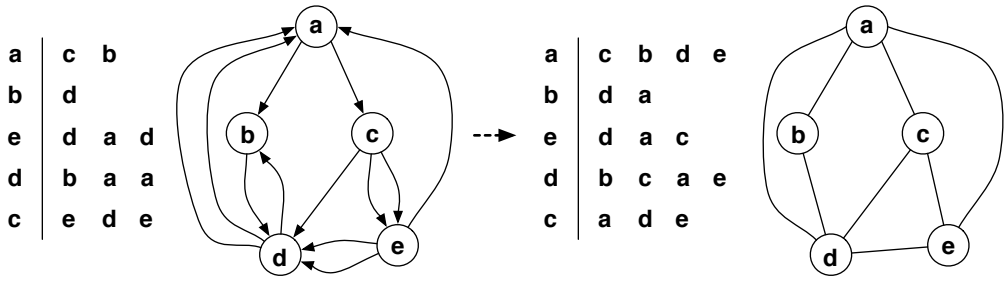
1. (15 pts total) Prof. Dumbledore needs your help to compute the in- and out-degrees of all vertices in a directed multigraph G. However, he is not sure how to represent the graph so that the calculation is most efficient. For each of the three possible representations, express your answers in asymptotic notation (the only notation Dumbledore understands), in terms of V and E, and justify your claim.

   (a) (5 pts) An *edge list* representation. Assume vertex indices can be arbitrary.

   (b) (5 pts) An *adjacency list* representation. Assume the vector's length is known.

   (c) (5 pts) An *adjacency matrix* representation. Assume the size of the matrix is known.

2. (20 pts) Let $G = (E, V)$ denote a directed multigraph. A simple and undirected graph is a $G' = (V, E')$, such that E' is derived from the edges in E so that (i) every directed multi-edge, e.g., $\{(u, v), (u, v)\}$ or even simply $\{(u, v)\}$, has been replaced by a single pair of directed edges $\{(u, v), (v, u)\}$ and (ii) all self-loops (u, u) have been removed.



An example of transforming $G \to G'$

Describe and analyze an algorithm (explain how it works, give pseudocode if necessary, derive its running time and space usage, and prove its correctness) that takes time $O(V+E)$ and $O(V+E)$ space to convert G into G'. Assume both G and G' are stored as adjacency lists.

Dumbledore's hints:
(i) Do not assume adjacencies `Adj[u]` are ordered in any particular way.
(ii) BFS and DFS will not work.
(iii) You can add edges to the list and then remove ones you don't need.

3. (15 pts total) A graph (V, E) is bipartite iff the vertices V can be partitioned into two subsets L and R, such that every edge has one end in L and the other in R.

   (a) (8 pts) Prove that every tree is a bipartite graph.
   Hint: Assume a tree is not bipartite and think about odd-length cycles.

   (b) (7 pts) Adapt an algorithm described in class so that it will determine whether a given undirected graph is bipartite. Give and justify its running time.

4. (20 pts) Prof. Snape tells you that when an *adjacency matrix* representation is used, most graph algorithms take $\Omega(V^2)$ time, but there are some exceptions. Snape claims that determining whether a directed graph G contains a *universal sink*, which is defined as a vertex with in-degree $|V|-1$ (i.e., every other vertex points to this one) and out-degree 0, can be done in time $O(V)$, given an adjacency matrix for G. Show that Snape is correct.
   Hint: To *show* this, describe the algorithm, provide pseudocode, and prove that it yields the correct answer on every type of input ("yes" when a universal sink exists and "no" when it does not). It will be helpful to identify and analyze the *boundary* cases, i.e., the cases where it is the most difficult to distinguish a correct "yes" from a correct "no."

5. (15 pts) Implement an efficient algorithm that computes the *diameter* of an unweighted, undirected input graph $G = (V, E)$, when G is given in adjacency list format, and which takes $O(V + E)$ additional space, and where the diameter is defined as the length of the longest non-infinite-length geodesic path between a pair of vertices $i, j \in V$. Write a paragraph explaining the algorithm, its running time, and how it satisfies the space requirement.

   No credit if you do not include the paragraph *and* submit your code as part of your solutions.

6. (15 pts total) Use your implementation from 5 to conduct the following numerical experiment to produce three nice-looking figures.

   No credit if you do not label your axes *and* include lines showing $f(n)$ for an asymptotic relation $O(f(n))$ *and* include the paragraphs.

   An Erdös-Rényi random graph, denoted $G(n, p)$, is a graph with n vertices and where each pairwise connection (u, v) for $u \neq v$ exists independently with probability p (edges

are unweighted, undirected and self-loops prohibited). When $p = c/(n-1)$, each vertex will have expected degree $E[k] = c$.

(a) (8 pts) For $c = 5$, show numerically that the expected diameter of $G(n, p)$ increases like $O(\log n)$. Because $G(n, p)$ is a random variable, you should average the diameter at a given value of n over several independent realizations (e.g., > 10) to get as smooth a curve as you can. Don't forget to vary n over a broad range. (One figure.) Include a paragraph the summarizes how you conducted the experiment.

(b) (7 pts) For the same choice of c, show that the running time and space requirements also follow your analytic prediction. (Two figures.) Include a paragraph the summarizes how you conducted the experiment.

Dumbledore's hints:
(i) If two vertices are not reachable, their distance is ∞; exclude such distances from your diameter calculation. (Because $G(n, p)$ is connected only probabilistically, at $c = 5$ there will sometimes be a few small, disconnected components. You may simplify your calculation by only analyzing the largest component of the input graph.)
(ii) As before, you'll need to implement some measurement code within your `diameter()` function that counts the number of atomic operations it performs. This time, you'll also need to implement code to measure the amount of space it uses.
(iii) You will also need to write a function `Random-Graph(n,p)` that returns an instance of $G(n, p)$ as an adjacency list. Note that this takes $\Theta(V^2)$ time since you need to flip a coin for each of the $\binom{n}{2}$ possible connections. Do not count this time toward the diameter calculation's running time.

7. (10 pts extra credit) Give an example of a directed graph $G = (V, E)$, a source vertex $s \in V$ and a set of tree edges $E_\pi \subseteq E$ such that for each vertex $v \in V$, the unique path in the graph (V, E_π) from s to v is a shortest path in G, yet the set of edges E_π cannot be produced by running a breadth-first search on G, no matter how the vertices are ordered in each adjacency list. Include a paragraph explaining why your example works.