Problem 1. (CLRS, Exercise 17.1-3) Suppose we perform a sequence of n operations on a data structure in which the ith operation costs i if i is an exact power of 2, and 1 otherwise. Use aggregate analysis to determine the amortized cost per operation.

Problem 2. Analyze the same problem as in problem 1, but with the accounting method.

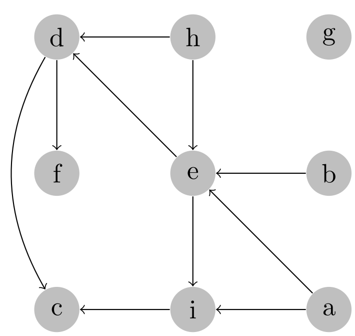
Problem 3. Let V be a fixed set of n vertices. How many distinct *undirected* graphs with vertex set V are there? How many distinct *directed* graphs with vertex set V are there? Briefly explain your solutions.

Problem 4. Suppose we represented graphs with a *list* of adjacency lists, instead of an *array* of adjacency lists. Give asymptotic bounds for the space complexity of this representation, for the time complexity of the lookup operation (checking for an edge from u to v), and for the time complexity of the insertion and deletion operations for both edges and vertices. What are the advantages and disadvantages over the representation by an array of adjacency lists?

Problem 5. It is often helpful to know, for a given vertex v, how many edges exist from v to other vertices u (this is called the "out-degree" of v) and how many from other vertices u to v (this is called the "in-degree" of v). For undirected graphs, the in-degree and out-degree match and are simply called the "degree" of v.

Give asymptotic bounds for computing the in-degree and the out-degree, respectively, of a single vertex v in a directed graph given in adjacency list representation and in matrix representation, respectively. Do the same for computing the in- and out-degrees of all vertices at once. That is, you are asked to give eight bounds in total.

Problem 6. Run DFS on the directed graph given below. Assume that the algorithm considers vertices in alphabetical order when choosing sources and that all adjacency lists are in alphabetical order as well. Give the discovery and finishing times for each vertex.



Problem 7. State an interpretation of the 15-puzzle (`https://en.wikipedia.org/wiki/15_puzzle`) as a graph problem, i.e., describe the vertices, edges, and the task. Describe an algorithm that receives this graph and the representation of the starting configuration as an input and solves the puzzle in the minimum number of steps, if a solution exists. Do not try to draw the graph, it is too big. How many vertices and edges does the graph have?