CS170 Discussion Section 3: *2/1*

1. *Practice with FT*
   What is the FT of $P(x) = 1 + x^3$? What values of x would we use with the FFT?

2. *Practice with FFT*

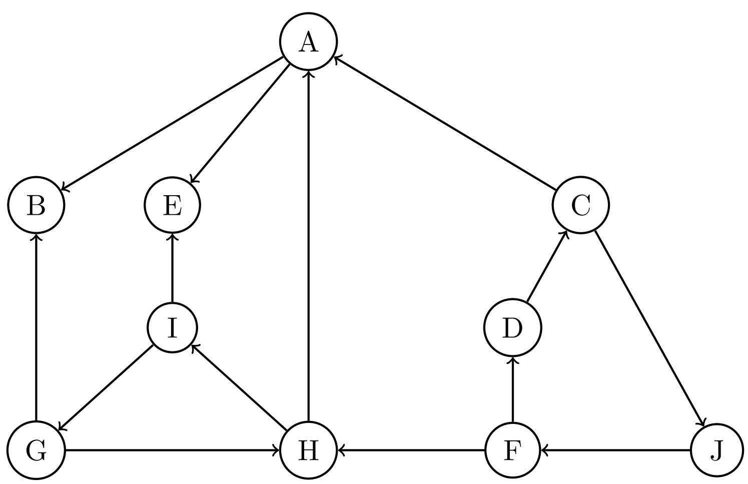
   (a) Run the FFT on $A(x) = 4 + 2x + 3x^2 + x^3$.

   (b) Is this result enough to compute $A(x)^2$?

   (c) *Extra Practice:* Compute $A(x)^2$ using the FFT.

3. *Graph traversal.*

   (a) For the directed graph below, perform DFS starting from vertex A, breaking ties alphabetically. As you go, label each node with its pre- and post-number, and mark each edge as **T**ree, **B**ack, **F**orward or **C**ross.



   (b) What are the strongly connected components of the above graph?

   (c) Draw the DAG of the strongly connected components of the graph.

4. *Short answer* For each of the following, either prove the statement is true or give a counterexample to show it is false.

   (a) If (u, v) is an edge in an undirected graph and during DFS, post$(v) <$ post(u), then u is an ancestor of v in the DFS tree.

   (b) In a directed graph, if there is a path from u to v and pre$(u) <$ pre(v) then u is an ancestor of v in the DFS tree.

   (c) In any connected undirected graph G there is a vertex whose removal leaves G connected.

5. **Extra Practice:** *Compiling on a parallel cluster* We want to compile a large program containing n modules. We are given a dependency graph $G = (V, E)$: G is a directed, acyclic graph with a vertex for each module, and an edge from module v to module w means we must finish compiling v before starting to compile w. Each module takes exactly one minute to compile. We want to compile the program as quickly as possible. We are willing to use Amazon EC2 for this purpose, so we can compile as many modules in parallel as we want, as long as we satisfy the dependencies. Find a linear-time algorithm that computes the minimum time needed to compile the program.

6. **Extra Practice:** *True Source* Design an efficient algorithm that given a directed graph G determines whether there is a single vertex v from which every other vertex can be reached. Hint: first solve this for directed acyclic graphs. Note that running DFS from every single vertex is not efficient.