Chapter 3, Problem 3.4: The reverse graph $G^R$ of (i)

(a) The order of the strongly connected components (SCCs) found in graph (i) is $[C, D, F, J] \Rightarrow [H, G, I] \Rightarrow [A] \Rightarrow [E] \Rightarrow [B]$.

(b) The source SCCs in graph (i) are $[B]$ and $[E]$; The sink SCC in graph (i) is $[C, D, F, J]$; The source SCCs in graph (ii) are $[A, B, E]$; The sink SCC in graph (ii) is $[D, G, H, F, I]$.

(c) Metagraphs of (i) and (ii).
(d) Two and one respectively.

Chapter 3, Problem 3.13: (a) Consider the DFS search tree (or any spanning tree of $G$). If we remove any leaf node of the tree, the graph is still connected since the other nodes are still connected by the tree.

(b) A graph with one directed cycle.

(c) A graph with two vertex-disjoint directed cycles.

Chapter 4, Problem 4.4: In the following counterexample, the shortest cycle is A-H-D-E-A whose length is 4 but the proposed algorithm can’t detect such cycle (we start the DFS with node A).

Chapter 4, Problem 4.5: Denote $\#(v)$ to be the number of shortest paths from $u$ to $v$. Our algorithm is similar to the BFS algorithm as follows

For all $v \in V \setminus \{u\}$:
#(v) = 0;
de(v) = \infty;
EndFor
#(u) = 1;
dist(u) = 0;
Q = [u]: (queue containing just u)
While Q is not empty:
    w = eject(Q);
    For all edge (w, l) ∈ E:
        If dist(l) == \infty:
            inject(Q, l);
            dist(l) = dist(w) + 1;
        EndIf
        If dist(l) == dist(w) + 1:
            #(l) = #(l) + #(w);
        EndIf
    EndFor
EndWhile

Since each node can only be injected and ejected from queue once and each edge can only be visited twice, similarly as BFS, the complexity of above algorithm is \(O(|V| + |E|)\).