CLRS 22-3 (a) Prove that a directed graph has an Euler circuit if and only of for all v in G, $indeg(v) = outdeg(v)$.

Solution: First note that the proof must have two parts:

$\Longrightarrow$: If G has an Euler circuit C, then C is either a simple cyle (does not intersect itself), or not. If C is a simple cycle, each vertex in a simple cycle has indeg=outdeg=1, so the claim is true. If C is a cycle but not a simple cycle, then it must contain a simple cycle; remove it from G and from C; the remaining C is still an Euler circut for the remaining G. Repeat removing (simple) cycles until no edges left. When removing a cycle, an in-edge and out-edge of the vertices on the cycle are removed. After a cycle deletion, the in-degree and out-degree of a node on the cycle decrease by exactly 1. At the end, when no edges are left, all in-degrees and out-degrees are 0. So all vertices v must have started with $indeg(v) = outdeg(v)$.

$\Longleftarrow$: If every vertex v has $indeg(v) = outdeg(v)$, the first observation is that for any vertex v, there must be a path starting from v that comes back to v (need to prove this, see below). Assuming this is true, pick a random vertex v and find a cycle C that comes back to v. Delete all the edges on C from G. Each vertex in the new G still has $indeg(v) = outdeg(v)$, so we pick a vertex v' on C that has edges incident (such a vertex must exist) and repeat. Overall we find a cycle C, then another cycle C' that has (at least) a common vertex with C, and so on. We can build a big cycle that goes arund C, jumps into C' ad goes around C', then comes back to C and finishes C.

Proof of the claim that we made above: For any vertex v, there must be a cycle that contains v. Start from v, and chose any outgoing edge of v, say (v, u). Since $indeg(u) = outdeg(u)$ we can pick some outgoing edge of u and continue visiting edges. Each time we pick an edge, we can remove it from further consideration. At each vertex other than v, at the time we visit an entering edge, there must be an outgoing edge left unvisited, since $indeg = outdeg$ for all vertices. The only vertex for which there may not be an unvisited outgoing edge is v—because we started the cycle by visiting one of v's outgoing edges. Since there's always a leaving edge we can visit for any vertex other than v, eventually the cycle must return to v, thus proving the claim.

(b) Describe how to find an Euler circuit in G.

Solution: First we can check in $O(|E|)$ time whether $indeg(v) = outdeg(v)$ is true for every vertex. If yes, then we can q find the Euler circuit by finding and deleting cycles as above. Let's argue that it all takes $O(|E|)$ time.

Pick a vertex v and perform DFS from it until finding a back edge that links back to v. Once you find this cycle, traverse all edges of the cycle, and delete the corresponding edge in the adjacency list of G; to delete edges of G quickly assume we midify DFS so that we store, for each edge that we traverse, a pointer to the corresponding edge in the adjacency list of G. With this information, deleteion of an edge can be done in constant time, basically because we don't need to "search" for the edge in G. Then we repeat. Overall this $O(|E|)$ time.