P1

Claim:

A connected graph G with m $\geq$ n edges can have it's edges oriented such that the degree of every vertex is at least 1. We use DFS to find a vertex u that is part of a cycle. We use DFS again starting from u. To every pair of vertices {v, w} encountered after u we assign directed outgoing edges (v, w). A special case is made for u on the first loop iteration. Only the first vertex x that u encounteres receives a directed incoming edge (u, x). All other edges from u on the first iteration are left undirected.

Algorithm:

```
// Where G is a connected graph with m >= n edges
OneIncoming(G):
    // Use modifiedDFS from HW2 P3a to find the vetices of a cycle in G
    Set V to modifiedDFS(G)
    Set v such that v is the first vertex in V
    // Output the orientation of edges of G
    // such that every vertex has at least one incoming edge
    Orient(v):
        Initialize S to be a stack with one element s
        Set b to true representing the first iteration
        Set L to be an empty list of directed edges
        While S is not empty
            Take a node u from S
            If Explored[u] = false then
                Set Explored[u] = true
                // Every iteration but the first
                If b is false then
                    For each edge (u, v) incident to u
                        Add v to the stack S
                        // All new edges are outgoing
                        If directed edge (v, u) not in L
                            Add directed edge (u, v) to L
                        EndIf
                    EndFor
                // Handle the first iteration differently
                Else if b is true then
                    For each edge (u, v) incident to u
                        Add v to the stack S
                        // Only the first edge is outgoing
                        If b is true then
                            Set b to false
                            Add directed edge (u, v) to L
                        EndIf
                    EndFor
                EndIf
```

```
        EndIf
    EndWhile
    Return L
```

Claim:

The algorithm terminates in O(m + n) time.

Proof:

We are using two instances of DFS which is known to terminate in O(m + n). Because no modification has added additional runtime to DFS beyond O(1) work, both instances still terminate in O(m + n). Because one instance of DFS is run after another, the runtime of the sum of the instances of DFS is O(m + n). Thus, the overall runtime of the algorithm is O(m + n). $\square$

Claim:

The algorithm orients the edges of connected graph G with m $\geq$ n edges such that that indegree of every vertex is at least 1.

Proof:

Consider arbitrary graph G with n vertices and m $\geq$ n edges. We know from HW2 P3 that any graph with m $\geq$ n edges contains a cycle. So, G contains a cycle. We also know from HW2 P3 that any graph with a cycle has at least one edge that can be removed from that cycle without disconnecting the graph. Remove such an edge e between vertices {u, v} from G to create graph G' where G' = G - e. We know DFS can reach every connected vertex in G' starting from u. G' is connected so we can reach every vertex in G' from u. Use DFS starting from u. For every edge between vertices {x, y} in G' encountered by DFS, draw a directed edge (x, y). Because G' is connected we know v will be encountered by DFS from another vertex z with an incoming edge (z, v). We now know that all vertices in G' except for u have an incoming edge. Copy the direction of every edge in G' to the corresponding edge in G such that all vertices in G have at least one incoming edge except for u. Add the directed edge (v, u) to G. Thus, all vertices in G have at least one incoming edge. $\square$