Θ(*n*+*m*) algorithm to print in-degree and out-degree

Pseudocode

```
ALGORITHM IN-DEGREE-OUT-DEGREE(G)
1 // inDegreeCount will be the counter for inDegrees of each vertex
2 // outDegreeCount will be the counter for outDegrees of each vertex
3
4 let inDegreeCount[1..G.V.length] and outDegreeCount[1.. G.V.length] be new arrays
5
6
7 // initialize inDegreeCount and outDegreeCount to 0
8 // Θ(n) time
9 for each vertex u ∈ G.V
10 inDegreeCount[u] = 0
11 outDegreeCount[u] = 0
12
13 // iterate over each edge so inDegreeCount outDegreeCount will be incremented
14 // Θ(m) time - where u is the Vertex and v is the Edge connected to Vertex
15 for each vertex u, edge v ∈ _G.E
16 outDegreeCount[u] += 1
17 inDegreeCount[v] += 1
18
19 // iterate over each vertex and print in-degree and out-degree of every vertex in an m-edge
20 // n-vertex directed graph
21 // Θ(n) time
22 for each vertex u ∈ _G.V
23 PRINT "out-degree({0}): ".format(u), outDegreeCount[u]
24 PRINT "in-degree({0}): ".format(u), inDegreeCount[u]
```

Analysis of algorithm **Θ(*n*+*m*)**

Θ(*n*) + Θ(*m*) + Θ(*n*) = Θ(*n*+*m*)

Lines 9 through 11 will initialize values of 0 for the counter representation in each in-degree and out-degree array. The length of each in-degree and out-degree array will be the length of the vertex array V. As we are iterating over all *n*-elements of the V array, this will be an Θ(*n*) time operation. Lines 15 through 17 will iterate over all the *m*-edges in the array E which will be an Θ(*m)* operation. For each out-degree and for each in-degree vertex of each edge, the counter will be incremented by 1 each time. Lines 22 through 24 will output the out-degree counts and in-degree counts for every vertex. This will be a Theta - Θ time operation. The Θ(*n+m*) represents the bounding of a function from both above and below, an asymptotically tight bound – which represents the exact asymptotic complexity of an algorithm. Theta notation Θ is used to denote that the rate of growth is equivalent to the specified values of V and E, and since we are iterating over every single vertex and edge presented, this will be a Θ(*n+m*) time complexity.