CIS 575. Introduction to Algorithm Analysis
Assignment #5, Spring 2019
Due Thursday, March 7, 11:59pm

You may if you so prefer work in groups of two in which case each name should be listed on your answer but only one of you should submit.

1. (15p). In this question, we shall consider undirected graphs.

1. (9p) Draw, or describe in words:
   (a) (3p) a graph with 5 edges that is *acyclic* and which has as *few* nodes as possible.
   (b) (3p) a graph with 6 edges that is *connected* and which has as *few* nodes as possible.
   (c) (3p) a graph with 6 edges that is *connected* and which has as *many* nodes as possible.
2. (6p) State (not prove) a general result: for n nodes,
   (a) (2p) an *acyclic* graph will have at ____________ edges
   (b) (2p) a *connected* graph will have at ____________ edges.
   (c) (2p) a *tree* will have __________________ edges.

2. (13p). Consider the algorithm below whose input is a directed graph G with nodes $1..n$ and with a edges, and which for each $i \in 1..n$ computes in $A[i]$ the number of *in*coming edges to i.

for $i \leftarrow 1$ **to** n
    $A[i] \leftarrow 0$
for $i \leftarrow 1$ **to** n
    edges $\leftarrow G.\text{ALLFROM}(i)$
    foreach $e \in$ edges
        $j \leftarrow$ the target of e
        $A[j] \leftarrow A[j] + 1$

Your task is, for each of the graph representations listed below, to analyze the running time of this algorithm. You should first express the running time of the second **for** loop as a sum

$$\sum_{i=1}^{n} \ldots$$

and use that to find f, as simple as possible, such that the total running time is in $\Theta(f(n, a))$. (*Hint*: it may help to use a_i to denote the number of edges with source i.)

1. (6p) G is represented by an adjacency matrix.
2. (7p) G is represented by adjacency lists.

3. (12p). Consider a directed graph where each edge e, in addition to a source $s(e)$ and a target $t(e)$, has a weight $w(e)$.

1. (6p) Write an algorithm that given a graph $G = (\{1..n\}, E)$ builds a new graph $G' = (\{1..n\}, E')$ where

$$E' = \{e \in E \mid w(e) > 7\}.$$

   You may assume that G' is initially $(\{1..n\}, \emptyset)$; the only way to access G is through ALLFROM, and the only way to construct G' is through PUT (which doesn't check if an edge is already there).
2. (6p) Assuming an *adjacency list* representation, analyze the running time of your algorithm, as a function of n and a (recall that $a = |E|$).