CSCI 411 - Advanced Algorithms and Complexity
Assignment 1

January 22, 2022

Solutions to the written portion of this assignment should be submitted via PDF to Blackboard. Make sure to justify your answers. C++ code should be submitted both on Blackboard and on turnin. Both parts of the assignment are due before **February 6th at 11:59 pm**.

There may be time in class to discuss these problems in small groups and I highly encourage you to collaborate with one another outside of class. However, you must write up your own solutions **independently** of one another. Feel free to communicate via Discord and to post questions on the appropriate forum in Blackboard. Do not post solutions. Also, please include a list of the people you work with at the top of your submission.

Written Problems

1. (10 pts) Sort the following functions in terms of asymptotic growth from smallest to largest. In particular, the resulting order $f_1, \ldots, f_{12}$ should be such that $f_1 = O(f_2)$, $f_2 = O(f_3)$, and so on. Identify any groups of functions that are Θ of one another.

n	2^n	n^3	$n\ln(n)$	2	$n!$	$\log_2((4n)^n)$	$\ln(n^2)$	$\left(\frac{3}{2}\right)^n$	$n^{1/5}$	$\ln^2(n)$	$52!$

2. Consider the following intuition for a sorting algorithm. Let A be a list of real numbers. If A is of size 0 or 1, return it since it is already sorted. Otherwise, pick the last element of A to be used as a pivot and call it p. For each element e of A except the last element, if $e \leq p$, place e in a list called L. On the other hand, if $e > p$, place e in a another list called R. Repeat this procedure on L and R and call the resulting lists L' and R'. Make a new list by adding p between L' and R'. Return the result.

   (a) (10 pts) Write pseudocode following the above intuition.

   (b) (5 pts) Determine the **worst-case** asymptotic run time of this algorithm and explain why this is the worst case.

   (c) (5 pts) Assume that the sizes of L and R are equal at each step of the algorithm. Find a recurrence relation describing the run time in this case.

   (d) (10 pts) Given this recurrence relation, what is the asymptotic run time of the algorithm? Be sure to justify your answer.

3. Let $G = (V, E)$ be a directed graph, $s \in V$, $N(u) = \{v | (u, v) \in E\}$ be the set of neighbors of $u \in V$, and $d(u, v)$ represent the shortest path distance between $u, v \in V$. If there is no path from u to v, $d(u, v) = \infty$. We say that G has property P with respect to s if, for $u, v \in V$, $|N(u)| \leq |N(v)|$ when $d(s, u) > d(s, v)$. Put another way, G has property P with respect to s if vertices that are further away from s have at most as many neighbors as those closer to s.

   (a) (5 pts) Describe an intuitive approach for determining whether or not G has property P with respect to s.

   (b) (15 pts) Write pseudocode for a function `hasP(G, s)` which returns `True` if G has property P with respect to s and `False` otherwise.

   (c) (5 pts) Argue that your pseudocode is correct.

   (d) (5 pts) Analyze the asymptotic run time of your algorithm.

4. Let $G = (V, E)$ be a directed graph. We would like to partition the vertices of G into three groups, A, B, and C:

$$A = \{v | u, v \in V, (u \rightsquigarrow v) \implies (v \rightsquigarrow u), \exists w \in V \text{ s.t. } v \rightsquigarrow w \text{ and } w \not\rightsquigarrow v\}$$
$$B = \{v | u, v \in V, (v \rightsquigarrow u) \implies (u \rightsquigarrow v), \exists w \in V \text{ s.t. } w \rightsquigarrow v \text{ and } v \not\rightsquigarrow w\}$$
$$C = \{v | v \in V, v \notin A \cup B\}$$

   In words, A is the set of vertices v such that (1) if $u \rightsquigarrow v$, then $v \rightsquigarrow u$ and (2) there is some vertex $w \in V$ such that $v \rightsquigarrow w$ but $w \not\rightsquigarrow v$. B is the set of vertices v such that (1) if $v \rightsquigarrow u$, then $u \rightsquigarrow v$ and (2) there is some vertex $w \in V$ such that $w \rightsquigarrow v$ but $v \not\rightsquigarrow w$. And C is the set of all vertices not included in A or B. There are several specific examples of A, B, and C at the end of this document.

   (a) (5 pts) Describe an intuitive approach for determining the size of the sets A, B, and C.

   (b) (15 pts) Write pseudocode for a function `getSetSizes(G)` which returns $(|A|, |B|, |C|)$, a triple with the sizes of each set.

   (c) (5 pts) Argue that your pseudocode is correct.

   (d) (5 pts) Analyze the asymptotic run time of your algorithm.

Coding Problem

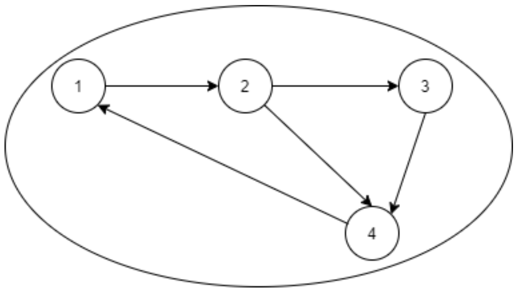
(20 pts) Write a C++ implementation of the pseudocode you developed for problem (4b) and submit to Blackboard and to turnin as assignment_1.cpp. Some skeleton code that you might find useful is available on Blackboard (assignment_1_skeleton.cpp).

- Input will come from cin
  - The first line will contain two integers, n and m, separated by a space.
  - n is a number of vertices and m is a number of edges.
  - The next m lines will contain two integers, u and v, separated by a space.
  - Each of these pairs represents a directed edge (u, v).
- Print output to cout
  - On one line print three space separated integers representing the sizes of of the sets A, B, and C in that order.

Examples

In the following examples, green nodes belong to A, red nodes belong to B, and white nodes belong to C.

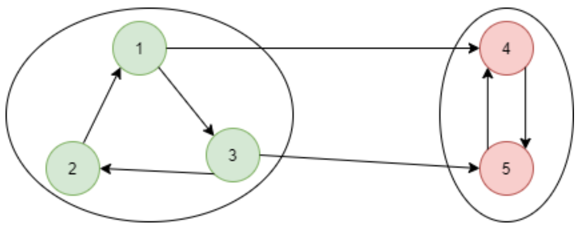
Example 1:



Input:
4 5
1 2
2 3
3 4
4 1
2 4

Expected output:
0 0 4

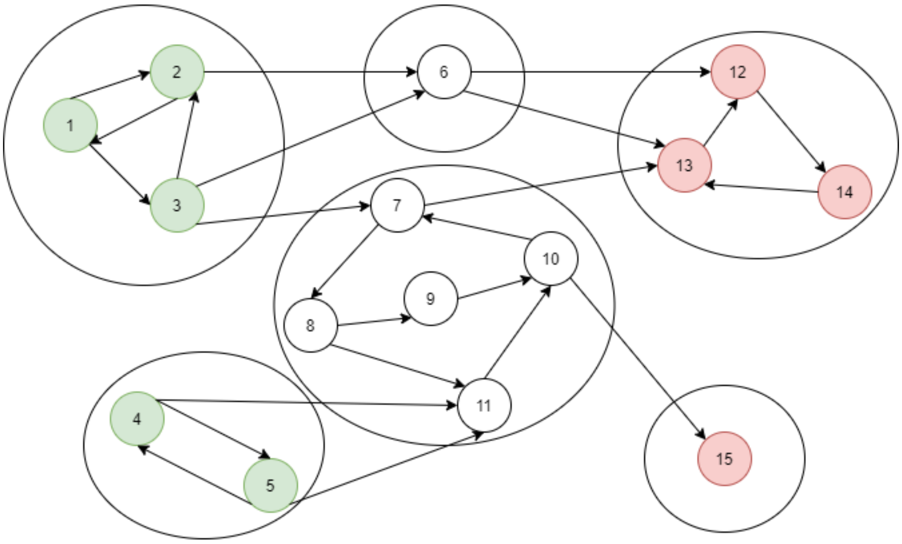
Example 2:



Input:
5 7
1 3
3 2
2 1
1 4
3 5
4 5
5 4

Expected output:
3 2 0

Example 3:



Input:
15 24
1 2
2 1
1 3
3 2
3 6
2 6
6 12
6 13
13 12
12 14
14 13
3 7
7 13
7 8
8 9
8 11
9 10
11 10
10 7
4 5
5 4
4 11
5 11
10 15

Expected output:
5 4 6