CSE 431/531: Algorithm Analysis and Design — Spring 2019

Homework 1

Instructor: Shi Li — **Deadline: 2/28/2019**

Problems	1	2	3	Total
Max. Score	20	20	40	80
Your Score				

Problem 1 (20 points) For each pair of functions f and g in the following table, indicate whether $f = O(g), f = \Omega(g)$ and $f = \Theta(g)$ respectively. Prove or disprove "$\lceil \sqrt{3n^2 + 100}\, \rceil = O(n)$", using the definition of O-notation.

$f(n)$	$g(n)$	O	Ω	Θ		$f(n)$	$g(n)$	O	Ω	Θ
$3n^2 - 10$	n					$\lceil \sqrt{3n^2 + 100}\, \rceil$	n			
$3n^2 - 10$	n^3					$n^{\sqrt{n}}$	$\sqrt{n}^n$			
$5n^3 - 10$	$4n^3 + 5n$					$n^{\sin n}$	n^2			
$\ln(n^{10})$	$\log_{10} n$					$2^{\log_3 n}$	$n^{1.5}$			
$(\ln n)^{10}$	$\log_{10} n$					2^n	e^n			

Problem 2 (20 points) Assume $f(n)$ and $g(n)$ are asymptotically positive functions. Whether each of the following statements is true or false? Prove or disprove (b) and (c), using definitions of asymptotic notations.

(a) If $f(n) = O(g(n))$, then $\sqrt{f(n)} = O\left(\sqrt{g(n)}\right)$.

(b) If $f(n) = O(g(n))$, then $(f(n))^2 = O\left((g(n))^2\right)$.

(c) If $f(n) = O(g(n))$, then $2^{f(n)} = O\left(2^{g(n)}\right)$.

Problem 3 (40 points) Given a directed graph $G = (V, E)$, design an algorithm that decides if G contains a cycle or not. In the directed graphs, a cycle is a sequence of distinct vertices $v_1, v_2, \cdots, v_t$ in V, with $t \geq 2$, such that: $(v_t, v_1) \in E$ and for every $i \in \{1, 2, 3, \cdots, t-1\}$, we have $(v_i, v_{i+1}) \in E$. If the graph contains a cycle, you need to output one; otherwise, you report there is no cycle. The running time of your algorithm should be $O(n + m)$.

This is a programming problem. You need to

1. write down the pseudo-code for your algorithm,

2. briefly explain why the algorithm is correct and why it runs in $O(n+m)$ time,
3. and use C++, Java or Python to implement your algorithm.

Implementation of the algorithm You need to read the graph G from the standard input (i.e, the terminal) and output the result to the standard output (i.e, the screen).

- **Input format**: In the first line of the input, there are two positive integers n and m. n is the number of vertices in the graph and m is the number of edges in the graph. The vertices are indexed from 1 to n. You can assume that $1 \le n \le 10000$ and $1 \le m \le 100000$. In the next m lines, each line contains 2 different integers u, v in $\{1, 2, \cdots, n\}$, indicating an edge (u, v) in the graph G. Every edge appears only once in the input.
- **Output format**: If the graph G does not contain a cycle, simply output an integer 0. If the graph contains a cycle, you need to output $t\ v_1\ v_2\ \cdots\ v_t$, where t indicates the length of the cycle you found, and $(v_1, v_2, \cdots, v_t)$ is the cycle.

Input #1:	**Output #1**:	**Input #2**:	**Output #2**:
3 3 1 2 2 3 1 3	0	6 9 1 2 4 2 1 4 3 4 4 6 3 6 5 3 2 5 1 5	4 2 5 3 4