CSE 431/531: Algorithm Analysis and Design Spring 2018

Homework 1

Instructor: Shi Li **Deadline: 2/26/2018**

Problems	1	2	3	Total Score
Max. Score	15	15	40	70
Your Score				

Collabroation Policy for This Homework You can not discuss Problem 1 and Problem 2 with your classmates. You can discuss the algorithm for Problem 3. However you must implement it completely by yourself.

Problem 1 (15 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. Justify your answer for the question "whether $\lceil\sqrt{3n+100}\rceil = O(\sqrt{n})$?", using the definition of the O-notation.

$f(n)$	$g(n)$	O	Ω	Θ
$3n^2+10$	$4n^2-n$			
$\log_{10} n$	$\log_2 n$			
$\lceil\sqrt{3n+100}\rceil$	$\sqrt{n}$			
n^3-100n	$10n^2\log n$			
$n^{\sqrt{n}}$	$\sqrt{n}^n$			
$2^{\log_3 n}$	$n^{1.5}$			

Problem 2 (15 points) Assume $f(n)$ and $g(n)$ are asymptotically positive functions. Whether each of the following statements is true or false? Justify your answers for questions (b) and (e), 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 $f^2(n) = O\left(g^2(n)\right)$.

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

(d) If $f(n) = O(g(n))$, then $\log(10+f(n)) = O\left(\log(10+g(n))\right)$.

(e) If $f(n)+g(n) = \Theta(n)$, then $f(n) = \Theta(n)$ or $g(n) = \Theta(n)$.

Problem 3 (40 points) This is a programming problem. Using depth-first-search to check if a given graph is bipartite or not. You can use C++, Java or Python to implement your algorithm.

- **Input**: You need to read the input graph from the standard input. 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 $1 \leq n \leq 10000$ and $1 \leq m \leq 100000$. In the next m lines, each line contains 2 integers: u, v and w, with $1 \leq u < v \leq n$. This indicates that there is an edge (u, v) in the graph G. It is guaranteed that the graph has no self-loops and parallel edges.
- **Output**: You need to print your result to the standard output. The output contains a single line, which is either "yes" or "no", indicating whether the graph is bipartite or not.

Input #1:	**Output #1**:	**Input #2**:	**Output #2**:
3 3 1 2 2 3 1 3	no	6 7 1 2 3 4 5 6 1 3 3 5 4 6 2 5	yes