Homework 1

Course: Algorithm Design and Analysis **Semester**: Spring 2024

Instructor: Shi Li **Due Date**: **2024/3/17**

Student Name: _____ Student ID: _____

Problems	1	2	3	4	5	6	Total
Max. Score	15	10	15	15	30	15	100
Your Score							

- Remark: In each algorithm design problem, you are expected to design the algorithm, prove its correctness and show that it achieves the required running time, unless otherwise stated. However, if the running time and/or correctness are easy to see, brief explanations are sufficient. Specifically, you may use the knowledge taught in class without proofs.

Problem 1. For each pair of functions f and g in the following table, indicate whether $f = O(g), f = \Omega(g), f = \Theta(g), f = o(g)$ and $f = \omega(g)$ respectively. You only need to give a yes/no answer for each question.

problem	$f(n)$	$g(n)$	O	Ω	Θ	o	ω
(1a)	$3n^2 + 10$	$5n^2 - 6n$					
(1b)	$\log_{10} n$	$5 \log_2(n^3) + 3$					
(1c)	$10n^2 - n$	$n^2 \log n$					
(1d)	$n^3 - 4n^2 + 10$	n^2					
(1e)	2^{3n}	7^n					
(1f)	$n^{\sin(n)}$	1					

Problem 2. Prove $\log_2(\lceil n^{3.5} \rceil) = \Theta(\log_{10} n)$, using the definition of the O-notation.

Problem 3. Assume both f and g are asymptotically positive functions. Decide if each of the following three statements is true or false. It it is true, prove it; if it is false, give a counterexample.

(3a) If $f(n) = O(g(n))$, then $f(n)^2 = O(g(n)^2)$.

(3b) If $f(n) = O(g(n))$, then $2^{f(n)} = O(2^{g(n)})$.

(3c) If $f(n) = O(g(n))$, and $f(n) \geq 2, g(n) \geq 2$ for every $n \geq 2$, then $\log f(n) = O(\log g(n))$.

Problem 4. Consider the following Euclidean algorithm for computing the greatest common divisor of two integers $a, b > 0$:

Algorithm 1 Euclid(a, b)

```
1: while b > 0 do
2:     t ← b, b ← a mod b, a ← t
3: return a
```

(4a) Prove that the algorithm terminates in $O(\log a)$ iterations.

(4b) Show that the $O(\lg a)$ bound is tight. In other words, prove that there exists a constant $c > 0$ such that for every $n_0 > 0$ there exist two integers a, b such that $a > b > n_0$ and Euclid(a, b) takes at least $c \lg a$ iterations to complete.

Problem 5. A cycle in an undirected (directed, resp.) graph $G = (V, E)$ is a sequence of $t \geq 3$ ($t \geq 2$, resp.) distinct vertices $v_1, v_2, \cdots, v_t$ such that $(v_i, v_{i+1}) \in E$ for every $i = 1, 2, \cdots, t-1$ and $(v_t, v_1) \in E$.

Given a undirected/directed graph $G = (V, E)$ with $n = |V|$ and $m = |E|$ using the linked-list representation, the goal is to design an $O(n + m)$-time algorithm that decides if G contains a cycle, and outputs an arbitrary one if yes.

There are three sub-problems here:

(5a) Solve the problem for undirected graphs.

(5b) Solve the problem for directed graphs using depth-first-search.

(5c) Solve the problem for directed graphs by extending the topological sort algorithm.

Problem 6. In class, we mentioned that the cut-vertices of a graph $G = (V, E)$ can be found in $O(n + m)$-time. Your goal is to give the algorithm. For your convenience, a pseudo-code for or a description of the algorithm is sufficient. You will get a full score if your algorithm is correct and has $O(n + m)$ running time.