THE HONG KONG POLYTECHNIC UNIVERSITY Fall 2020
COMP3011: Design and Analysis of Algorithms

Assignment 1

Due: Monday, 28 September 2020, 3:30 p.m. (before the start of the lecture)

To submit the assignment, upload a .pdf-, .doc-, .docx-, .txt-, or .jpg-file with your solutions to Blackboard before the deadline. The filename should start with "`A1_99999999Z`" (replace "`99999999Z`" by your student ID number). Include your name and student ID number at the top of each page. Scanned, handwritten solutions are OK as long as they are easily readable. You may specify your algorithms in pseudocode or in plain English.

The maximum possible score is 15 marks. Late submissions will not be accepted. The assignment is to be completed individually, i.e., every student must turn in his or her own solutions. Students are allowed to discuss solution strategies with other students but they are not allowed to copy each others' answers.

1. Give asymptotic upper and lower bounds for $T(n)$ in each of the following recurrences using Θ-notation. Assume that $T(n)$ is constant for sufficiently small values of n. Justify your answers.

   a) $T(n) = 9\,T(n/3) + n + 3011$ *(1 mark)*

   b) $T(n) = T(n/2) + 2\,T(\lg n) + n$ *(2 marks)*

   c) $T(n) = \sqrt{n} \cdot T(\sqrt{n}) + n$ *(2 marks)*

2. Consider the following problem:

   Input: Two unsorted lists A and B of numbers.
   Output: "YES" if some number belongs to both A and B; "NO" otherwise.

   Let m be the length of A and let n be the length of B. Design a fast algorithm for the special case of the problem where m is much smaller than n and analyze its time complexity. When $m = O(\lg n)$, your algorithm must have a worst-case running time of $o(n \lg n)$.

   Hint: Hashing is not needed here. If you decide to use hashing anyway, you have to explain how to ensure that the correct answer is found within the allowed time, even in the worst case. *(2 marks)*

3. Give an $O(\lg n)$-time algorithm for the following problem, justify why it works, and analyze its time complexity.

   Input: A sorted array $A[1..n]$ of distinct integers.
   Output: Any index i such that $A[i] = i$, if such an index exists; -1 otherwise.

   For example, if the input is $\langle -2020, -928, -1, 0, 5, 6, 330, 3011 \rangle$ then a valid output is 5. Alternatively, 6 would also be a valid output. *(3 marks)*

4. In this question, we shall use the definitions below.

   - Let $G = (V, E)$ be a graph and let k be a positive integer. A *k-coloring of G* is a partition of V into disjoint subsets $V_1, V_2, V_3, \ldots, V_k$ called *color classes* such that for any $\{x, y\} \in E$, it holds that x and y belong to different color classes, i.e., $x \in V_i$ and $y \in V_j$ with $i \neq j$. A *minimum graph coloring* of G is a k-coloring of G for the smallest possible integer k.
   - For every positive integer $n \geq 3$, define *the cycle graph C_n* as the graph (V_n, E_n), where $V_n = \{x_1, x_2, \ldots, x_n\}$ and $E_n = \big\{\{x_1, x_2\}, \{x_2, x_3\}, \ldots, \{x_{n-1}, x_n\}, \{x_n, x_1\}\big\}$.
   - The *degree* of a vertex x in graph, denoted by $\deg(x)$, is the number of neighbors of x.

(Continued $\longrightarrow$)

The *greedy graph coloring algorithm* takes as input an undirected graph $G = (V, E)$ and does the following:

```
Fix any ordering $\langle v_1, v_2, \ldots, v_n \rangle$ of $V$ such that $\deg(v_i) \geq \deg(v_{i+1})$ for every $i \in \{1, 2, \ldots, n-1\}$.
for $i = 1, 2, \ldots, n$
    Assign vertex $v_i$ the smallest color not used by any of its neighbors.
```

Will applying the greedy graph coloring algorithm to C_n always compute a minimum graph coloring of C_n when n is an odd integer? If your answer is "yes", prove it. If your answer is "no", give a counterexample. *(2 marks)*

5. Suppose that P is a sequence of points in the plane. Any point $(x, y) \in P$ is called *maximal with respect to* P if there is no $(x', y') \in P$ with $(x, y) \neq (x', y')$ for which both $x \leq x'$ and $y \leq y'$ hold. The set of all points in P that are maximal with respect to P is denoted by $\mathcal{M}(P)$. For example, if $P = \langle(3, 4), (5, 2.5), (1, 5), (4, 4), (2.8, 2)\rangle$ then $\mathcal{M}(P) = \{(1, 5), (4, 4), (5, 2.5)\}$.

   Describe an $O(nm)$-time greedy algorithm that takes as input a sequence P of points in the plane and outputs all points in $\mathcal{M}(P)$, where $n = |P|$ and $m = |\mathcal{M}(P)|$, by selecting one point from P in each iteration and inserting it into the solution. State clearly how a point is selected in each iteration and how P is updated. Prove the correctness of your algorithm and analyze its time complexity to make sure it runs in $O(nm)$ time. *(3 marks)*