Assignment 2
COMP 3804, Fall 2018
Upload in cuLearn by 1PM on October 15, 2018

1 Guidelines

General guidelines are as follows:

1. Since we are only accepting assignments via CU-Learn, no late submissions will be entertained after the cut-off time & date.

2. Please write clearly and answer questions precisely. It is your responsibility to ensure that what is uploaded is clearly readable. If we can’t read, we can’t mark!

3. Please cite all the references (including web-sites, names of friends, etc.) which you used/consulted as the source of information for each of the questions.

4. All questions/problems carry equal marks.

5. When a question asks you to design an algorithm - it requires you to
   (a) Clearly spell out the steps of your algorithm in pseudo code.
   (b) Prove that your algorithm is correct
   (c) Analyze the running time.
2 Problems

1. Let \( S \) be a set of \( n \) distinct real numbers, and we are interested in finding the median element of this set. In the class (refer to the class notes), we saw an algorithm that forms groups of five elements each, finds median element of each group, finds the median of these medians, and uses that element to partition the set. The overall running time of that algorithm for finding median was \( O(n) \), as the underlying recurrence \( T(n) = T(n/5) + T(7n/10) + O(n) \) (and \( T(c) = O(1) \) for any constant \( c \leq 200 \)) evaluates to linear. Determine what will be the corresponding recurrence equation, if we form \( n/3 \) groups, each group consisting of three elements. Now we will compute the median of each group, then median of \( n/3 \) medians, and use that median to partition the set. Does the new recurrence equation which you obtained also evaluates to \( O(n) \)? Justify your answer.

2. Let \( S \) be a set of \( n \) distinct real numbers. Devise an algorithm, running in \( O(n + k \log k) \) time, to report the \( k \) smallest elements of \( S \) in sorted order, where \( k \in \{1, \ldots, n\} \).

3. Let \( S \) be a set of \( n \)-distinct real numbers and let \( k \leq n \) be a positive integer (\( k \) may not be a constant). Design an algorithm, running in \( O(n) \) time, that determines the \( k \) numbers in \( S \) that are closest to the median of \( S \). For example, for the set \{21, 70, 3, 1, 6, 7, 11, 2, 9, 8, 17, 13, 25\}, median is 9, and the \( k = 4 \) closest numbers to 9 are 8, 7, 11, and 6.

4. Let \( A \) and \( B \) be two sorted arrays, each array consists of \( n \) real numbers in ascending order. Give an algorithm, running in \( O(\log n) \) time, to compute the median of the elements formed by the union of the elements in both the arrays. (You may assume that the union consists of \( 2n \) distinct real numbers.) Hint: Assume that the median element is from \( A \), and assume that it is at index \( i \). Then for \( x = A[i] \) to be the median element, you can say something about how many elements in \( B \) need to be smaller than \( x \) and that can be checked in \( O(1) \) time. The problem to solve here is how fast you can search for \( x \) in \( A \)?

5. For the graph in the figure, perform depth-first search starting at the vertex marked \( A \), and whenever there is a choice of vertices, pick the one the lexicographic smallest. Classify each edge as a tree edge, forward edge, back edge or a cross edge, and give the pre and post number of each vertex. Show your work!
6. Let $G = (V, E)$ be a simple undirected graph. Provide an algorithm running in $O(|V| + |E|)$ time, which outputs whether $G$ contains a cycle or not. If it contains a cycle - then it needs to output at least one cycle. What graph representation you have used for your algorithm. Justify why you used that and remember to link this justification with your complexity analysis.

7. Design an algorithm that determines whether a directed graph $G = (V, E)$ is an acyclic graph (i.e., it doesn’t contain a directed cycle). Your algorithm must run in $O(|V| + |E|)$ time.

8. Typically departments in universities (like Carleton) offer many courses, but to register in a course, one needs to have completed all the required prerequisite courses. We can easily model this relationship as a directed graph, where each course is a vertex, and a directed edge from course $u$ to $v$ if and only if $u$ is a prerequisite course for taking $w$. It should be clear that this graph should not contain any directed cycles (otherwise we won’t graduate!). (For example, if COMP 1405 and COMP 1805 are required for taking COMP 2402, and COMP 2402 is required for taking COMP 3804, we will have directed edges from vertices corresponding to COMP 1805 and COMP 1405 to COMP 2402, and a directed edge from COMP 2402 to COMP 3804.) Given a directed graph $G = (V, E)$ in adjacency list representation, representing the courses and their prerequisites, your task is to compute minimum number of terms one needs to spend in the department to complete the degree, where you can assume that you can do any number of courses in any term, provided that the prerequisite conditions are met.

9. Given a directed graph $G = (V, E)$, where each vertex has a distinct integer label. For each vertex $v$, define $R(v)$ to be the set of all vertices $w \in V$ for which there is a directed path from $v$ to $w$ in $G$. Furthermore, for each vertex $v \in V$, define $\text{MinLabel}(v)$ to be the vertex with the minimum label in the set $R(v)$. Provide an algorithm, running in $O(|V| + |E|)$ time, that computes $\text{MinLabel}(v)$ for all vertices $v \in V$. 
Let $s$ and $t$ be two specific vertices of an undirected connected simple graph $G = (V, E)$ on $n$-vertices where any path between $s$ and $t$ in $G$ consists of at least $n/2 + 1$ vertices. Show that there is a vertex $v \in V$, $v \neq s$ and $v \neq t$, such that any path from $s$ to $t$ passes through $v$. Also, provide an algorithm running in $O(|V| + |E|)$ time for identifying such a vertex $v$ for a given pair of vertices $s, t \in V$. (Note that by removing $v$ from $G$, we disconnect $s$ and $t$.)