CSC-421 Applied Algorithms and Structures
Spring 2021-22

Instructor: Iyad Kanj
Office: CDM 832
Phone: (312) 362-5558
Email: `ikanj@cs.depaul.edu`
Office Hours (Office/Zoom): Monday 4:40-5:40 & Wednesday 1:00-3:00
Course Website: https://d2l.depaul.edu/

Assignment 1
(Due April 11)

Remarks

- When asked to give an algorithm that meets a certain time bound, you need to give the algorithm (pseudocode/description) and analyze its running time to show that it meets the required bound; giving only the algorithm is not enough to receive full credit.
- Please submit your solutions as a single PDF file. If your solutions consist of multiple files, convert all your files into a single PDF file and upload it on D2L.
- Please double check that you submitted the correct and complete file after uploading your (final) submission. No late resubmissions will be allowed.

1. (30 points) For each of the following two functions $f(n)$ and $g(n)$, indicate whether $f = O(g)$, or $f = \Omega(g)$ or both (in which case $f = \Theta(g)$).

   (a) $f(n) = n - 100$ and $g(n) = n - 200$.

   (b) $f(n) = n^{1/2}$ and $g(n) = n^{2/3}$.

   (c) $f(n) = 100n + \lg n$ and $g(n) = n + (\lg n)^2$.

   (d) $f(n) = n \lg n$ and $g(n) = 10n \lg (10n)$.

   (e) $f(n) = 10 \lg n$ and $g(n) = \lg (n^2)$.

   (f) $f(n) = n^2 / \lg n$ and $g(n) = n(\lg n)^2$.

   (g) $f(n) = n^{0.1}$ and $g(n) = (\lg n)^{10}$.

   (h) $f(n) = \sqrt{n}$ and $g(n) = (\lg n)^3$.

   (i) $f(n) = n2^n$ and $g(n) = 3^n$.

   (j) $f(n) = 2^n$ and $g(n) = 2^{n+1}$.

2. (25 points) Given a collection of n nuts and a collection of n bolts, arranged in an increasing order of size, give an $O(n)$ time algorithm to check if there is a nut and a bolt that have the same size. The sizes of the nuts and bolts are stored in the sorted arrays $NUTS[1..n]$ and $BOLTS[1..n]$, respectively. Your algorithm can stop as soon as it finds a single match (i.e, you do not need to report all matches).

3. (25 points) Let $A[1..n]$ be an array of distinct positive integers, and let t be a positive integer.

   (a) (10 points) Assuming that A is sorted, show that in $O(n)$ time it can be decided if A contains two distinct elements x and y such that $x + y = t$.

   (b) (15 points) Use part (a) to show that the following problem, referred to as the 3-SUM problem, can be solved in $O(n^2)$ time:

   > 3-SUM
   > Given an array $A[1..n]$ of distinct positive integers that is not (necessarily) sorted, and a positive integer t, determine whether or not there are three distinct elements x, y, z in A such that $x + y + z = t$.

4. (20 points) Let $A[1..n]$ be an array of positive integers (A is not sorted). Pinocchio claims that there exists an $O(n)$-time algorithm that decides if there are two integers in A whose sum is 1000. Is Pinocchio right, or will his nose grow? If you say Pinocchio is right, explain how it can be done in $O(n)$ time; otherwise, argue why it is impossible.

5. **Suggested Programming Problems on LeetCode; not to be submitted.** Below are the titles of coding problems (related to the materials we covered so far) that I suggest that you do in LeetCode.com (under the tab "Problems"). You can test your submissions and view the solutions in LeetCode.

   (i) Two Sum.

   (ii) 3Sum.

   (iii) Valid Triangle Number.