ASSIGNMENT 1

DUE: Wednesday, January 19, 11:59pm. DO NOT COPY. ACKNOWLEDGE YOUR SOURCES.

Please read `http://www.student.cs.uwaterloo.ca/~cs341` for general instructions and policies.
Note: All logarithms are base 2 (i.e., $\log x$ is defined as $\log_2 x$).

Exercises. The following exercises are for practice only. You may ask about them in office hours. Do not hand them in.

1. Apply the instructions for Problem 1 below to $f(n) = \log n^{10}$, $g(n) = \log_{10} n$.
2. Analyze the number of scalar multiplications that are needed to multiply an $n \times m$ matrix A with an $m \times k$ matrix B using the standard method.

   Now suppose you needed to multiply three matrices ABC, where C is a $k \times \ell$ matrix. Is the number of scalar multiplications the same whether you do $(AB)C$ or $A(BC)$?

   What about asymptotically? That is, if we let the runtimes be $T_1(n, m, k, \ell)$ and $T_2(n, m, k, \ell)$, are we guaranteed that $T_1(n, m, k, \ell) \in \Theta(T_2(n, m, k, \ell))$?

   Note that the resulting final matrix will be the same either way, since matrix multiplication is associative.

Problems. To be handed in.

1. [12 marks, 3 marks each] **Order notation.** For each of the following pairs of functions $f(n)$ and $g(n)$ defined on positive integers, determine the "most appropriate" symbol in the set $\{O, o, \Theta\}$ to fill in the blank in the statement $f(n) \in$ ____$(g(n))$, if one of these symbols applies. (It is possible that none of the choices applies.) "Most appropriate" means that you should not answer "O" if you could answer "o" or "Θ". *Justify your answers.*

   You may use the following facts (based on Skiena, p. 56), where $f(n) \ll g(n)$ is shorthand for $f(n) \in o(g(n)$:

   $$1 \ll \log\log n \ll \log n \ll \log^2 n \ll \sqrt{n} \ll n \ll n \log n \ll n^2 \ll 2^n \ll n!$$

   Furthermore, $n^a \in o(n^b)$ for $0 < a < b$, and $\log^a n \in o(n^b)$ for any $b > 0$, and $n^a \in o(2^n)$.

   (a) $f(n) = 3.14159n^3 + 4000n$, $g(n) = .0001n^3 - 2^{2^{10}} n^2$;
   (b) $f(n) = \log(n!), g(n) = n(\log n)^2$;
   (c) $f(n) = (\log n)^{\log\log n}, g(n) = 2^{\sqrt{\log n}}$;
   (d) $f(n) = (n \bmod 3) \cdot n, g(n) = 0.1n$. Here $(n \bmod 3) \in \{0, 1, 2\}$ is the remainder of n when divided by 3.

2. [10 marks] **Reductions.** A special case of the 3SUM problem from lecture 2 is called 3SUM0: given an array $A[1 \dots n]$ of numbers, decide whether there exist indices $i, j, k \in \{1, \dots, n\}$ (duplicate indices are allowed) such that $A[i] + A[j] + A[k] = 0$.

   Now, we consider the problem (ARITHPROG): given three arrays of numbers $X[1 \dots n]$, $Y[1 \dots n]$, and $Z[1 \dots n]$, decide whether there are three distinct entries that form an arithmetic progression. Formally, decide whether there exist indices $i, j, k \in \{1, \dots, n\}$ such that $Y[j] - X[i] = Z[k] - Y[j]$.

   In this question, you are asked to show that the two versions are equivalent by providing reductions from one to another in both directions. Recall that, when you are reducing from problem P1 to problem P2, you assume that you have a subroutine that solves problem P2, and you use this subroutine to design an algorithm for P1.

   (a) [3 marks] Give a linear time reduction from 3SUM0 to ARITHPROG and prove correctness.

   (b) [7 marks] Give a linear time reduction from ARITHPROG to 3SUM0 and prove correctness.

3. [10 marks] **Recurrences.** Consider the following recursive algorithm which takes as input a list of n numbers, and outputs another list that's a permutation of these n numbers.

   $B[1 \dots n] = \textsc{GooseSort}(n, A[1 \dots n])$
   if $n < 5$ **then**
   | Sort A using bubble sort, and return the sorted array;
   else
   | Let $m \leftarrow \lceil \frac{3n}{5} \rceil$;
   | $A[1 \dots m] = \textsc{GooseSort}(m, A[1 \dots m])$;
   | $A[(n - m + 1) \dots n] = \textsc{GooseSort}(m, A[(n - m + 1) \dots n])$;
   | $A[1 \dots m] = \textsc{GooseSort}(m, A[1 \dots m])$;
   | Return A;
   end

   For example, on the input sequence

   $$1, 5, 3, 2, 4$$

   The first recursive call turns it into

   $$1, 3, 5, 2, 4,$$

   the second recursive call turns it into

   $$1, 3, 2, 4, 5,$$

   and the last recursive call produces

   $$1, 2, 3, 4, 5.$$

   (a) [2 marks] Write down, with a brief explanation, the recurrence for the running time of the above recursive algorithm. When you encounter some term in the form of $T(\lceil an \rceil + c)$ with constant a and c then you can ignore the ceiling (similar to floor) and constant c, and write $T(an)$ for simplicity.

(b) [3 marks] Based on the recurrence in part (a), use the recursion tree method to guess the correct order of the running time of GooseSort($A[1 \dots n]$, k).

(c) [3 marks] Derive the runtime bound of the recurrence in part (b) using the Master Theorem.

Be sure to specify the values of a, b, and k, as well as the case of Master Theorem that applies here.

(d) [2 marks] Give an example sequence that's a permutation of $\{1, 2, 3, 4, 5\}$ where GooseSort does not terminate with the sorted array.

Challenge Question This is for fun and enrichment only. Do not hand it in.

1. In class you learned about the 3SUM problem. For this question, we will use the following variant of 3SUM: Given a set S of n distinct numbers, are there three (distinct) elements of S that sum to 0? Finding an algorithm that is faster than $O(n^2)$ is a subject of recent research.

   A basic question about planar point sets is **Collinearity**: given n points in the plane, do any three of them lie on a line? Here also, there are $O(n^2)$ time algorithms, and there is recent research on finding a faster algorithm.

   Show that these problems are related, by showing that 3SUM reduces to Collinearity in linear time. (This means that a faster algorithm for Collinearity would give a faster algorithm for 3SUM.) Hint: Show that for a, b, c distinct, $a + b + c = 0$ if and only if the three points $(a, a^3), (b, b^3), (c, c^3)$ lie on a line.