CS 473, Fall 2017
Homework 0 (due September 6 Wednesday at 8pm)

For Homework 0, you should work *independently*. Carefully read `http://engr.course.illinois.edu/cs473/policies.html` and `http://engr.course.illinois.edu/cs473/integrity.html`. Submit via Gradescope.

1. [*16 pts*]

   (a) [*3 pts*] Solve the following recurrence (i.e., give tight Θ bound): $T(n) = 3\,T(n/4) + \sqrt{n}$ if $n \geq 4$, and $T(n) = 1$ if $n < 4$. Justification is not necessary.

   (b) [*7 pts*] Consider the following recurrence: $T(n) = 2\,T(\sqrt{n}) + \log n$ if $n \geq 4$, and $T(n) = 3$ if $n < 4$. Use induction to prove that $T(n) = O(\log n \log\log n)$.

   (c) [*6 pts*] The following recurrence bounds the number of comparisons of array elements made by the SMAWK algorithm:

   $$T(m, n) \;=\; \min\{T(m/2, n) + n,\; T(m, m) + n + m\}$$

   if $m > 1$, and $T(1, n) = n$. As noted in class, a linear upper bound on $T(n, n)$ can be obtained by "alternating" between the two options: $T(n, n) \leq T(n/2, n) + n \leq (T(n/2, n/2) + n + n/2) + n = T(n/2, n/2) + 5n/2$, which implies $T(n, n) \leq 5n + O(1)$. Improve the constant factor, i.e., prove that the above function satisfies $T(n, n) \leq cn + O(1)$ for some constant $c < 5$. (Try to make c as small as you can.)

2. [*12 pts*] We are given an $n \times n$ matrix A where all entries are integers from $\{1, 2, \ldots, U\}$, with the property that all rows are monotonically increasing and all columns are monotonically increasing, i.e., $i < i'$ implies $A[i, j] \leq A[i', j]$, and $j < j'$ implies $A[i, j] \leq A[i, j']$. Describe an $O(n \log U)$-time algorithm to find the median element in A (i.e., the $(n^2/2)$-th smallest element).

   [Hint: first describe an $O(n)$-time algorithm to count the number of elements less than a given value.]

3. [*12 pts*] We are given a set of n line segments in 2D, where each line segment is either vertical (with endpoints (x_i, y_i) and (x_i, y_i') for some x_i, y_i, y_i') or or horizontal (with endpoints (x_i, y_i) and (x_i', y_i) for some x_i, x_i', y_i). We are also given two points $s = (x_s, y_s)$ and $t = (x_t, y_t)$. Describe an efficient algorithm to decide whether there is a way to travel from s to t without crossing any of the given line segments. [Hint: consider an $n \times n$ grid, define a graph (with how many vertices and edges?), and apply a standard graph search algorithm.]