CSE 101 Fall 2018
Homework 1
Due: Tuesday, October 9, 2018 at 11:59pm

---

INSTRUCTIONS

This homework assignment may be completed in groups of size 1-4. The solutions must be **typed** (using a computer.) Figures and graphs can be handdrawn. For algorithm descriptions we require a high-level English description AND an implementation description. If you find it necessary to also include a pseudocode to help understand your description then feel free to include it.

Please refer to the course page for requirements in writing up answers for algorithm questions.

---

1. (25 points) Consider the following programs:

   Alg1(n):
   For $i = 1$ to $n$ For $j = 1$ to $n$
   If $i + j <= n$
   Print((i,j))

   Alg2(n):
   For $i = 1$ to $n$ For $j = 1$ to $n$
   If $i * j <= n$
   Print((i,j))

   For each of these algorithms, compute the asymptotic number of printed lines in the form $\Theta()$.

2. (25 points) The Fibonacci numbers $F_0, F_1, ...$, are defined by

   $$F_0 = 0, F_1 = 1, F_n = F_{n-1} + F_{n-2}$$

   Use induction to prove that:

   (a) Use induction to prove that $F_n \geq 2^{0.5n}$ for $n \geq 6$

   (b) Use induction to prove that $F_n \leq 2^n$ for $n \geq 0$

   (c) Based on the previous parts, write an upperbound of $F_n$ using $O$ notation and a lower bound using $\Omega$ notation.

3. (25 points) Let $G$ be an undirected graph with nodes $v_1, ..v_n$. The *adjacency matrix* representation for $G$ is the $n \times n$ matrix $M$ given by: $M_{i,j} = 1$ if there is an edge from $v_i$ to $v_j$, and $M_{i,j} = 0$. A *triangle* is a set $\{v_i, v_j, v_k\}$ of 3 distinct vertices so that there is an edge from $v_i$ to $v_j$, another from $v_j$ to $v_k$ and a third from $v_k$ to $v_i$. Give and analyze an algorithm for determining if a graph $G$, given by its adjacency matrix representation, has a triangle. Analyze your algorithm's worst-case performance first in terms of just the number of nodes $n$ of the graph, then in terms of $n$ and the number of edges $m$ of the graph. Your algorithm should be faster when $m << n^2$.

4. (25 points) The reverse of a directed graph $G$ is another directed graph $G^R$ with the same vertex set with the property that if $(u, v)$ is an edge in $G$ then $(v, u)$ is an edge in $G^R$.

   Consider the following algorithm that takes the adjacency list $A[v_1, v_2, \ldots, v_n]$ of a directed graph $G$ as input and outputs an adjacency list of $G^R$.

   **procedure** *reversegraph*$(A[v_1, v_2, \ldots, v_n])$
   initialize a list $A^R[v_1, \ldots, v_n]$
   **for** each $i = 1 \ldots n$:
   **for** each $u \in A[v_i]$:
   add $v_i$ to the list $A^R[u]$
   **return** $A^R$

   (a) Justify the correctness of this algorithm

   (b) Analyze the runtime of this algorithm assuming that $G$ has $n$ vertices and $m$ edges.