**Fundamental Algorithms**
**Courant, Fall 2017**

# Homework 2

**Instructor: Amir Shpilka**
**Due Date: Wed Sep 20, 11:55pm**

1. (10 points) For each of the following pairs of functions $f(n)$ and $g(n)$, state whether $f$ is $O(g)$; whether $f$ is $o(g)$; whether $f$ is $\Theta(g)$; whether $f$ is $\Omega(g)$; and whether $f$ is $\omega(g)$. (More than one of these can be true for a single pair!)
   (a) $f(n) = 11n^{19} + \log(n) + 31; \quad g(n) = \frac{5n^{20}+11111n^2+3}{111} - 52n.$
   (b) $f(n) = \log(n^3 - 30n); \quad g(n) = \log(n^2 + 400).$
   (c) $f(n) = \log(4^n + n^3); \quad g(n) = \log(n^{999}).$
   (d) $f(n) = n^3 \cdot 2^n; \quad g(n) = n^2 \cdot 3^n.$
   (e) $f(n) = (n^n)^4; \quad g(n) = n^{(n^4)}.$

2. (Polynomial Evaluation) A degree-$n$ polynomial $P(x)$ is a function

$$P(x) = a_0 + a_1 x + \ldots + a_{n-1}x^{n-1} + a_n x^n = \sum_{i=0}^{n} a_i x^i$$

   (a) (2 points) Express the value $P(x)$ as

$$P(x) = a_0 + a_1 x + \ldots + a_{n-2}x^{n-2} + b_{n-1}x^{n-1} = \sum_{i=0}^{n-1} b_i x^i$$

   where $b_0 = a_0, \ldots, b_{n-2} = a_{n-2}$. What is $b_{n-1}$ as a function of the $a_i$'s and $x$?

   (b) (4 points) Using part (a) above write a recursive procedure $\mathsf{Eval}(A, n, c)$ to evaluate the polynomial $P(x)$ whose coefficients are given in the array $A[0 \ldots n]$ (i.e., $A[0] = a_0$, etc.) at point $c$. For example, if $A = [7, -3, 5], n = 2, c = 1$ then we need to evaluate the polynomial $P(x) = 7 - 3x + 5x^2$ at $x = 1$ and return the result $P(1) = 9$. Make sure you do not forget the base case $n = 0$.

   (c) (3 points) Let $T(n)$ be the running time of your implementation of $\mathsf{Eval}$ (you can count the costs of just additions and multiplications). Write a recurrence equation for $T(n)$ and solve it in the $\Theta(\cdot)$ notation.

   (d) (6 points) Assuming $n$ is a power of 2, try to express $P(x)$ as $P(x) = P_0(x) + x^{n/2}P_1(x)$, where $P_0(x)$ and $P_1(x)$ are both polynomials of degree $n/2$. Assuming the computation of $x^{n/2}$ takes $O(n)$ time, describe (in words or pseudocode) a recursive procedure $\mathsf{Eval}_2$ to compute $P(x)$ using two recursive calls to $\mathsf{Eval}_2$. Write a recurrence relation for the running time of $\mathsf{Eval}_2$ and solve it. How does your solution compare to your solution in part (c)?

   (e) (5 points) (**Bonus**[1]) Explain how to fix the slow "conquer" step of part (d) so that the resulting solution is as efficient as part (c).

3. Consider the following recursive procedure:

```
1 def foo(n):
2     if n == 1:
3         return 1
4     else:
5         return foo(n/3) + foo(n/3)
```

[1] Bonus questions are not mandatory, and not attempting them will not harm you in any way. They are often more difficult (so that you don't complain the homework isn't challenging enough) and/or teach extra material.

(a) (3 points) What function of $n$ does `foo` compute, assuming it is always called on an $n$ which is a power of 3?

(b) (3 points) What is the running time of `foo`? You can give an asymptotic answer using $\Theta$-notation.

(c) (4 points) How do the previous two answers change if line 5 of `foo` was replaced by **`return`** `2 * foo(n/3)`?

4. (5 points) Find a *divide-and-conquer* algorithm that finds the maximum and the minimum of an array of size n using at most $3n/2$ comparisons.

   (**Hint**: First, notice that we are not asking for some iterative algorithm (which is not hard). We are asking for you to explicitly use recursion. In fact, your divide/conquer step should take time $O(1)$. Also, you have to be super-precise about constants and the initial case $n = 2$ to get the correct answer.)

5. (5 points) (**Bonus**) Stirling's approximation (given below) gives us an asymptotic bound for $n!$ (see Wikipedia for more details).

$$\log n! = \Theta(n \log n - n)$$

   Using this, prove that $\log \binom{n}{k} = \Theta(k \log \frac{n}{k})$ for $0.1 \log n < k \leq \frac{n}{2}$.