CS 473: Algorithms, Fall 2010
HW 0 (due Tuesday, August 31st in class)

This homework contains four problems. **Read the instructions for submitting homework on the course webpage**. In particular, *make sure* that you write the solutions for the problems on separate sheets of paper. Write your name and netid on each sheet.

Collaboration Policy: For this home work, each student should work *independently* and write up their own solutions and submit them.

Read the course policies before starting the homework. Problems 1-3 should be answered in Compass as part of the assessment HW0-Online.

Note: Before starting to answer the questions on compass, read the following recaps:

- $\lg n = \log_2 n$ and $\ln n = \log_e n$.
- $\lg^2 n = (\lg n)^2$ and $\lg \lg n = \lg(\lg n)$.
- H_n is the n'th harmonic number and $H_n = \sum_{i=1}^{n} 1/i \simeq \ln n + 0.577215\ldots$.
- F_n is the n'th Fibonacci number and satisfies the recurrence $F_n = F_{n-1} + F_{n-2}$ with $F_0 = 0, F_1 = 1$. It can be verified by induction (try it!) that $F_n = (\phi^n - (-1/\phi)^n)/\sqrt{5}$ where $\phi = (1 + \sqrt{5})/2$ is the golden ratio.

---

1. (10pts) True/False questions on background.
2. (25pts) Asymptotics.
3. (25 pts) Basic recurrences.
4. (40pts) Euclid's algorithm for finding the greatest common divisor (gcd) of two non-negative numbers a, b is the following.

   ```
   Algorithm Euclid(a, b):
     If (b = 0)
        return a
     Else
        return Euclid(b, a mod b)
   ```

   Prove via induction that the algorithm correctly computes the gcd of a, b. Also prove that the running time of the algorithm is polynomial in the input size. Note that the input size is $\Theta(\log a + \log b)$. Assume that the mod operation along with other basic arithmetic operations take constant time. *Hint:* For both parts think about how $a + b$ is changing in each recursive call. A slow version of the Euclid algorithm is the following.

```
Algorithm SlowEuclid(a, b):
  If (b > a)
    return SlowEuclid(b, a)
  Else if (b = 0)
    return a
  Else
    return SlowEuclid(b, a − b)
```

Verify for yourself that the above algorithm correctly computes the gcd of a and b. Show that the above algorithm can take exponential time in the input size. You can do this by giving a class of instances $(a_1, b_1), (a_2, b_2), \ldots, (a_n, b_n), \ldots$ where $\log a_n + \log b_n \to \infty$ and the running time of the algorithm on (a_n, b_n) is exponential in $\log a_n + \log b_n$ (the input size) for each n.