6331 - Algorithms, Autumn 2016, CSE, OSU
Homework 1
Instructor: Anastasios Sidiropoulos
Due date: Aug 26, 2014

Notation. Let $\mathbb{N}$ denote the set of natural numbers, that is $\mathbb{N} = \{1, 2, 3, \ldots\}$. We denote by log, the logarithm to the base 2, that is $\log(n) = \log_2(n)$.

Problem 1. Order the following functions by asymptotic dominance. That is, give an ordering $f_1, f_2, \ldots$, such that for all $i \geq 1$, we have $f_i(n) = O(f_{i+1}(n))$.

- $f(n) = 2^n$.
- $f(n) = 1000 \cdot n$.
- $f(n) = n^{\log n}$.
- $f(n) = n^2$.
- $f(n) = 2^{2^n}$.
- $f(n) = \log(n)$.
- $f(n) = \log(\log(n))$.
- $f(n) = n \cdot \log(n)$.

Problem 2. Prove or disprove that $2^{n+1} = O(2^n)$. Prove or disprove that $2^{2n} = O(2^n)$.

Problem 3. Give an example of a function $f : \mathbb{N} \to \mathbb{N}$, such that $f(n) = \Omega(n)$, and f is not monotonically increasing.

Problem 4. Give the asymptotic running time of the following algorithm in Θ notation. Briefly justify your answer. Be sure to justify both the upper and the lower bound.

```
Func1(n)
s ← 0;
i ← 5;
while (i < n^2 + 7) do
    for j ← i to i^3 log i do
        s ← s + 1;
    end
    i ← i × 4;
end
```

Problem 5. Give an example of a function $f : \mathbb{N} \to \mathbb{N}$, such that for any constant $c > 0$, we have $f(n) = \omega(n^c)$, and for any constant $c' > 0$, we have $f(n) = o(2^{c' \cdot n})$.