COMS 311: Homework 1
Due: Feb 9^{th}, 11:59pm
Total Points: 30 (+10 extra credit)

Late submission policy. Any assignment submission that is late by not more than two business days from the deadline will be accepted with 20% penalty for each business day. That is, if a homework is due on Friday at 11:59 PM, then a Monday submission gets 20% penalty and a Tuesday submission gets another 20% penalty. After Tuesday no late submissions are accepted.

Submission format. Homework solutions will have to be typed. You can use word, LaTeX, or any other type-setting tool to type your solution. Your submission file should be in pdf format. Do **NOT** submit a photocopy of handwritten homework except for diagrams that can be hand-drawn and scanned. We reserve the right **NOT** to grade homework that does not follow the formatting requirements. Name your submission file: `<Your-net-id>-311-hw1.pdf`. For instance, if your netid is `asterix`, then your submission file will be named `asterix-311-hw1.pdf`. Each student must hand in their own assignment. If you discussed the homework or solutions with others, a list of collaborators must be included with each submission. Each of the collaborators has to write the solutions in their own words (copies are not allowed).

General Requirements

- When proofs are required, do your best to make them both clear and rigorous.
- Even when proofs are not required, you should justify your answers and explain your work.
- When asked to present a construction, you should show the correctness of the construction.

Some Useful (in)equalities

- $\sum_{i=1}^{n} i = \frac{n(n+1)}{2}$
- $\sum_{i=1}^{n} i^2 = \frac{n(n+1)(2n+1)}{6}$
- $2^{\log_2 n} = n$, $a^{\log_b n} = n^{\log_b a}$, $n^{n/2} \leq n! \leq n^n$, $\log x^a = a \log x$
- $\log(a \times b) = \log a + \log b$, $\log(a/b) = \log a - \log b$
- $a + ar + ar^2 + ... + ar^{n-1} = \frac{a(r^n-1)}{r-1}$
- $1 + \frac{1}{2} + \frac{1}{2^2} + ... + \frac{1}{2^n} = 2(1 - \frac{1}{2^{n+1}})$
- $1 + 2 + 4 + ... + 2^n = 2^{n+1} - 1$

Problems

1. **(5 pts)** Prove $\left[\frac{n(n+1)}{2}\right]^2 - \frac{n^2(n^2+1)}{4} + 78 \in O(n^3)$.

$$\begin{aligned}\left[\frac{n(n+1)}{2}\right]^2 - \frac{n^2(n^2+1)}{4} + 78 &= \frac{n^2(n^2+2n+1)}{4} - \frac{n^2(n^2+1)}{4} + 78 \\ &= \frac{n^4}{4} + \frac{2n^3}{4} + \frac{n^2}{4} - \frac{n^4}{4} - \frac{n^2}{4} + 78 \\ &= \frac{2n^3}{4} + 78 \\ &\leq 79n^3 \qquad \left[\frac{2n^3}{4} \leq n^3 \text{for } n \geq 1\right]\end{aligned}$$

Therefore, there exists a $c(=79)$ and $n_0 = 1$, such that for all $n \geq n_0$,

$$\left[\frac{n(n+1)}{2}\right]^2 - \frac{n^2(n^2+1)}{4} + 78 \leq c \cdot n^3$$

2. **(5 pts)** Prove or disprove $2^{2^n} \in O(2^{2n})$.

Proof by contradiction.

Assume that $2^{2^n} \in O(2^{2n})$. Then, $\exists c > 0$, $\exists n_0 > 0$, such that $\forall n \geq n_0 \quad 2^{2^n} \leq c \cdot 2^{2n}$.

Taking logs on both sides, we get:

$$2^n(\log_2 2) \leq \log_2 c + 2n(\log_2 2)$$

Since $\log_2 2 = 1$,

$$\begin{aligned}&\Rightarrow 2^n \leq \log_2 c + 2n \\ &\Rightarrow 2^n - 2n \leq \log_2 c\end{aligned}$$

The left hand side $2^n - 2n$ increases indefinitely with increasing values of n, and there cannot be any unique constant c such that $\log_2 c$ is greater than or equal to all valuations of $2^n - 2n$.

This results in a contradiction. Hence, our assumption is false and $2^{2^n} \notin O(2^{2n})$.

3. **(5 pts)** Prove that any function that is in $O(\log_2 n)$ is also in $O(\log_3 n)$.

We want to prove that any function $f(n)$ that is in $O(\log_2 n)$ is also in $O(\log_3 n)$. Since we know that $f(n) \in O(\log_2 n)$, there must $\exists c > 0$, $\exists n_0 > 0$ such that

$$\forall n \geq n_0 \quad f(n) \leq c \cdot \log_2 n$$

Changing the log base to 3:

$$f(n) \leq c \cdot \frac{\log_3 n}{\log_3 2}$$

Define $c' = \frac{c}{\log_3 2}$, where c' is a new constant, we have:

$$f(n) \leq c' \cdot \log_3 n$$
$$\Rightarrow f(n) \in O(\log_3 n)$$

4. **(5 pts)** Prove that if $f_1(n) \in O(g_1(n))$ and $f_2(n) \in O(g_2(n))$, then

$$\boldsymbol{f_1(n) + f_2(n) \in O(g_1(n) + g_2(n))}$$

If $f_1(n) \in O(g_1(n))$, then:

$$\exists c_1 > 0, \exists n_{01} > 0, \text{ such that } \forall n \geq n_{01} \;\; f_1(n) \leq c_1 \cdot g_1(n)$$

If $f_2(n) \in O(g_2(n))$, then:

$$\exists c_2 > 0, \exists n_{02} > 0, \text{ such that } \forall n \geq n_{02} \;\; f_2(n) \leq c_2 \cdot g_2(n)$$

We choose $c = max(c_1, c_2)$ and $n_0 = max(n_{01}, n_{02})$.

So, $f_1(n) \leq c \cdot g_1(n)$ and $f_2(n) \leq c \cdot g_2(n)$

$$\Rightarrow f_1(n) + f_2(n) \leq c(g_1(n) + g_2(n))$$
$$\Rightarrow f_1(n) + f_2(n) \in O(g_1(n) + g_2(n))$$

5. **(10 pts)** Derive the runtime of the following loop structure as a function of n and determine its Big-O upper bound. You must show the derivation of the end result. Simply stating the final answer without any derivation steps will result in zero points. Assume atomic operations take unit time.

```
for (i = 1; i <= n; i++)
{
    for (j = n; j >= 1; j--)
    {
        for (k = 1; k <= i + j; k++)
        {
            // some constant number of atomic operations

        } // end k

    } // end j

} // end i
```

$$
\begin{aligned}
\text{Runtime} &= \sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k=1}^{i+j} 1 \\
&= \sum_{i=1}^{n}\sum_{j=1}^{n}(i+j-1+1) \\
&= \sum_{i=1}^{n}\sum_{j=1}^{n}(i+j) \\
&= \sum_{i=1}^{n}\left[\sum_{j=1}^{n} i + \sum_{j=1}^{n} j\right] \\
&= \sum_{i=1}^{n}\left[ni + \frac{n(n+1)}{2}\right] \\
&= \sum_{i=1}^{n} ni + \sum_{i=1}^{n} \frac{n(n+1)}{2} \\
&= n\sum_{i=1}^{n} i + n \cdot \frac{n(n+1)}{2} \\
&= n \cdot \frac{n(n+1)}{2} + n \cdot \frac{n(n+1)}{2} \\
&\in O(n^3)
\end{aligned}
$$

6. **Extra Credit (10 pts)** Derive the runtime of the following loop structure as a function of n and determine its Big-O upper bound. You must show the derivation of the end result. Simply stating the final answer without any derivation steps will result in zero points. Assume atomic operations take unit time.

```
i = n;

while (i >= 2)
{
    for (j = 1; j <= i; j++)
    {
        // some constant number of atomic elementary operations

    } // end j

    i = i / 2;

} // end while
```

For each iteration of the outer loop, the inner loop iterates i times. We need to identify the number of times the outer loop iterates (in terms of n) and for each such iteration identify the value of i.

Let the outer loop iterate R times.

Outer Loop Iteration	i Value at Beginning of Iteration	Number of Times Inner Loop Iterates
1	n	n
2	$\frac{n}{2}$	$\frac{n}{2}$
3	$\frac{n}{2^2}$	$\frac{n}{2^2}$
4	$\frac{n}{2^3}$	$\frac{n}{2^3}$
...	...	...
R	$\frac{n}{2^{R-1}}$	$\frac{n}{2^{R-1}}$

Runtime = number of times the inner loop iterates for each iteration of outer loop:

$$n\left[1+\frac{1}{2}+\frac{1}{2^2}+...+\frac{1}{2^{R-1}}\right]=2n\left(1-\frac{1}{2^R}\right)$$

If the outer loop iterates R times, then $\frac{n}{2^{R-1}} \geq 2$ according to while loop condition. So, $2^R \leq n$. Hence, the runtime is $\leq 2n\left(1-\frac{1}{n}\right)$, which means that it is in $O(n)$.