CS 575
Homework 1

I have done this assignment completely on my own. I have not copied it, nor have I given my solution to anyone else. I under- stand that if I am involved in plagiarism or cheating I will have to sign an official form that I have cheated and that this form will be stored in my official university record. I also understand that I will receive a grade of 0 for the involved assignment for my first offense and that I will receive a grade of F for the course for any additional offense.

Aniruddha Tekade

1: Prove the following using the original definition of O, Ω, Θ, o, ω

(a) $10n^3 + 2n + 15 = O(n^3)$
Explation $\Rightarrow$ Definition of `Big-Oh` is $0 \leq g(n) \leq cf(n)$ for all $n \geq N$
Applying this definition to the given statement, we get -

$$\mathbf{0 \leq 10n^3 + 2n + 15 \leq cn^3}$$

That means there must exists c>0 and integer N>0.
Dividing both sides of the inequality by $n^3>0$ we get:

$$\mathbf{0 \leq 10 + 2/n^2 + 15/n^3 \leq c \text{ for all } n \geq N}$$

$(2/n^2 + 15/n^3) > 0$ becomes smaller as n increases
Clearly any c can be chosen anything ≥ 27 if we put N = 1.
For instance c = 27, N = 1
Hence, $10n^3 + 2n + 15 = O(n^3)$

(b) $7n^2 = \Omega(n)$
Explation $\Rightarrow$ Definition of `Big-Omega` is $0 \leq cf(n) \leq g(n)$ for all $n \geq N$
Applying this definition to the given statement, we get -

$$\mathbf{0 \leq cn \leq 7n^2}$$

That means there must exists c>0 and integer N>0.
Dividing both sides of the inequality by $n>0$ we get:

$$\mathbf{0 \leq c \leq 7n \text{ for all } n \geq N}$$

Clearly, there are many choices for c and N.

For instance c = 7, N = 1
Hence, $7n^2$ = Ω(n)

(c) $5n^2 = \omega$(n)
Explation ⇒ Definition of `small-Omega` is 0 ≤ cf(n) ≤ g(n) for all n ≥ N, where cf(n) is strictly less than g(n).
Applying this definition to the given statement, we get -

$$\mathbf{0 \leq cn \leq 5n^2}$$

That means there must exists c>0 and integer N>0.
Dividing both sides of the inequality by n>0 we get:

$$\mathbf{0 \leq c \leq 5n \text{ for all } n \geq N}$$

Clearly, there are many choices for c and N.
For instance c = 5, N = 1
Hence, $5n^2 = \omega$(n)

(d) $7n^3 + 15n^2 + 5 = \Theta(n^3)$
Explation ⇒ To prove $7n^3 + 15n^2 + 5 = \Theta(n^3)$ we nee to first prove that $7n^3 + 15n^2 + 5 = O(n^3)$ as well as $7n^3 + 15n^2 + 5 = \Omega(n^3)$
Proof that prove that $7n^3 + 15n^2 + 5 = O(n^3)$ ⇒
Acording to the definition of `Big-Oh` is $0 \leq 7n^3 + 15n^2 + 5 \leq cn^3$ for all n ≥ N.
That means there must exists c>0 and integer N>0.
Dividing both sides of the inequality by n^3>0 we get:

$$\mathbf{0 \leq 7 + 15n^2 + 5/n^3 \leq c \text{ for all } n \geq N}$$

$(15n^2 + 5/n^3) > 0$ becomes smaller as n increases
Clearly any c can be chosen anything ≥ 27 if we put N = 1.
For instance c = 27, N = 1
Hence, $\mathbf{7n^3 + 15n^2 + 5 = O(n^3)}$

Proof that prove that $7n^3 + 15n^2 + 5 = \Omega(n^3)$ ⇒
Acording to the definition of `Big-Omega` is $0 \leq 7n^3 + 15n^2 + 5 \leq cn^3$ for all n ≥ N.
That means there must exists c>0 and integer N>0.
Dividing both sides of the inequality by n^3>0 we get:

$$\mathbf{0 \leq cn^3 \leq 7 + 15n^2 + 5/n^3 \text{ for all } n \geq N}$$

$(15n^2 + 5/n^3) > 0$ becomes greater as n grows
For instance c = 27, N = 1
Hence, $\mathbf{7n^3 + 15n^2 + 5 = O(n^3)}$
Since both Big-Oh and Big-Omega complexities satisfies, it means, there exists, asymptotically Theta bound for this relationship. Hence we can conclude that - $\mathbf{7n^3 + 15n^2 + 5 = \Theta(n^3)}$

(e) p(n) = $\sum_{i=1}^{k} a^i n^i = \Theta(n^k)$; where $a_i > 0$
Explation ⇒
Let's prove that $\sum_{i=1}^{k} a^i n^i = O(n^k)$
By definition of Big-Oh, we have -
⇒ $0 \leq \sum_{i=1}^{k} a^i n^i \leq$ cn^k
⇒ $\sum_{i=1}^{k} a^i \sum_{i=1}^{k} n^i \leq$ cn^k

$\Rightarrow$ We know that - $\sum_{i=1}^{k} a^i = 1 + \text{a} + a^2 + a^3 + ... + a^k = \frac{a^{k+1}-1}{a-1}$ &
$\Rightarrow \sum_{i=1}^{k} n^i = 1 + \text{n} + n^2 + n^3 + ... + n^k = \frac{n^{k+1}-1}{n-1}$

Combining above two results we get -

$(\frac{a^{k+1}-1}{a-1})\ (\frac{n^{k+1}-1}{n-1}) \leq cn^k \Rightarrow$ which clears that there is always a constant c > 0.

Therefore, $\sum_{i=1}^{k} a^i n^i = O(n^k)$

Similarly we can prove the for Ω, that $\sum_{i=1}^{k} a^i n^i = \Omega(n^k)$
Therefore, p(n) = $\sum_{i=1}^{k} a^i n^i = \Theta(n^k)$; where $a_i > 0$

2: Prove the following using limits.

(a) $n^k = o(3^n)$; where k > 0
Explanation $\Rightarrow n^k \in o(2^n)$ where k is a positive integer.
$3^n = e^{nln3}$
$(3^n)\text{'} = (e^{nln3})\text{'} = \text{ln}2e^{nln3} = \text{ln}3(3^n) = \lim_{n\to\infty} \frac{kn^(k-1)}{3^n ln2}$
$= \lim_{n\to\infty} \frac{n^k}{3^n} = \lim_{n\to\infty} \frac{kn^(k-1)}{3^n ln2}$

$= \lim_{n\to\infty} \frac{k(k-1)n^{k-2}}{3^n ln^2 2} = ... = \lim_{n\to\infty} \frac{k!}{3^n ln^k 2}$

Therefore, $n^k = o(3^n)$; where k > 0

(b) n = ω(lg n^5)
Explanation $\Rightarrow$ We know that lg $n^5 = \frac{lnn^5}{ln2}$

Taking derivative, we get - (lg n^5)' = $(\frac{lnn^5}{ln2})\text{'} = \frac{5}{nln2}$

$\lim_{n\to\infty} \frac{n}{lgn^5} = \lim_{n\to\infty} \frac{n'}{(lgn^5)'}$

$\lim_{n\to\infty} \frac{nlg2}{5}$

Therefore, n is greater than lg n^5. We can say, n = ω(lg n^5)

3: Prove that 3^n-1 is divisible by 2 for n = 1, 2, 3, ... by induction. Divide your proof into the three required parts: Induction Base, Induction Hypothesis, and Induction Steps.

Base Case $\Rightarrow$ If n = 1, 3^n-1 is divisible by 2.
3^n-1 = 3^1-1 = 3 - 1 = 2. & 2 is divisible by 2.

Assumption ⇒ It is true that 3^k-1 is also divisible by 2 where k is any random integer that is greater than 0.

Proof ⇒
If we put k = 2 → 3^2-1 = 9 - 1 = 8. 8%2=0
If we put k = 3 → 3^3-1 = 27 - 1 = 26. 26%2=0
If we put k = 4 → 3^4-1 = 81 - 1 = 80. 80%2=0
If we put k = 5 → 3^5-1 = 243 - 1 = 242. 242%2=0

Therefore it is true that, 3^n-1 is divisible by 2 for any integer value of n.

4: Prove or disprove that n^3 = O(n^2)

Explanation ⇒ Lets apply the definition of `Big-Oh` on above question -
⇒ $0 \leq n^3 \leq cn^2 \Rightarrow n^3 \leq cn^2$ ⇒ Let's divide both sides by n^2 ⇒ n ≤ c ⇒ Above expression states that, n will always be smaller than a constant c, which is absolutely false since as n grows, c being a constant become small at some point or the other. ⇒ Therefore it is proved that $n^3 \neq$ O(n^2)

5: Just say True or False for the following

(a) $1000000n^2 + 5000 = \Theta\ (n^2)$ ⇒ **True**

(b) 2^{n+1} = O(2^n) ⇒ **True**

(c) $n^3 + n^2$ + 100n = $\Omega(n^3)$ ⇒ **True**

(d) $n^{1000} = \omega(2^n)$ ⇒ **False**

(e) log n^{100} = Ω(log n) ⇒ **True**

6: Analyze the worst case time complexity of recursive binary search using the iterative method. Assume the number of data n = 2^k

Explanation ⇒ The recurrence equation for binary search is as follows -

T(1) = 1; for
T(n) = 1+T(⌊ n/2 ⌋); for n>2

Applying recurrence oprations -
at *i=1* ⇒ T(n) = 1 + T(⌊ n/2 ⌋)
at *i=2* ⇒ T(n/2) = 2 + T(⌊ n/4 ⌋)
at *i=3* ⇒ T(n/4) = 3 + T(⌊ n/8 ⌋)
. . .

. . .
. . .
at *i=n-1* $\Rightarrow$ T(n/2^k) = K+T(1)

Terminating the recurrence, we get,
$\Rightarrow$ T(n) = k + T($\lfloor 1 \rfloor$) = ($\lfloor$log n$\rfloor$)+1
$\Rightarrow$ We know that, data is n = 2^k, which means k = log n
$\Rightarrow$ Therefore, we can prove that the worst case time complexity of binary search method is O(logn)

7: The following pseudo code computes a factorial for input parameter n that is expressed via s bits where n, s $\geq$ 1. What is the time complexity of the pseudo code?

```
unsigned int fact(unsigned int n) {
unsigned int p = 1;
for (i=1; i n; i++) p = p * i;
return p;
}
```

Explanation $\Rightarrow$ The analysis of factorial algorithms is as follows -

- Total number of operations = n
- Number bits for value n is s = $\lfloor$(log x)$\rfloor$+1
- $\lfloor$(log x)$\rfloor$ = s - 1
- $n \geq 2^{s-1}$
- Therefore, there are `Exponential` number of operations in terms of s