Data Structure Homework 1

□□ : **41247001S** □□□□□□□

Question 1 :

Use the definition of big-O to prove that $3n^2$ + $2nlog_2n^2 = O(n^2)$. Provide appropriate constants c and $n0$. (20□)

Answer :

□□ Big-O □□□□□□ $f(x) \leq cn^2$ □ $f(x) = O(n^2)$ □□ $3n^2$ + $2nlog_2n^2 \leq cn^2$ □□ $3n^2$ + $2nlog_2n^2 = O(n^2)$

□□□□□□□□□□□ c □□□□□□□□□□$3n^2$ + $2nlog_2n^2$ □□□□□ $O(n^2)$

$3n^2$ + $2nlog_2n^2 \leq cn^2$

$= 3n^2$ + $2n * 2log_2n \leq cn^2$

$= 3n^2$ + $4n * log_2n \leq cn^2$

$= 3$ + $(4log_2n)/n \leq c$

□□ $(log_2n)/n \leq 1$ □□ 3 + $(4log_2n)/n \leq 7$ □□□ $c = 7$ □□ $3n^2$ + $2nlog_2n^2 \leq cn^2$ □□□□ $n0$ □□□□ $n \geq n0$, $(log_2n)/n \leq 1$ □□□□□□□

□□□□ $3n^2$ + $2nlog_2n^2 \leq cn^2$ □□□□□□□ □□□□ $3n^2$ + $2nlog_2n^2 = O(n^2)$

Question 2 :

Show that $4n^3 + 8n^2 + 2^n$ = $\Omega()$. Please find the maximum order for the big-Ω estimation. Please also provide the values of c and n0 satisfying the definition of the big-Ω estimation. (20□)

Answer :

□□ Big-Ω □□□□□□ $f(x) \geq c * g(x)$ □□ $f(x) = \Omega(g(x))$ □□□□□□□□ **maximum order** □□□ □□□□ $g(x)$ □ 2^n (□□ $4n^3 + 8n^2 + 2^n$ □ 2^n □□□□□□□□)

$4n^3 + 8n^2 + 2^n \geq 2^n = 4n^3/2^n + 8n^2/2^n + 1 \geq c$ □□□□□□□ $c = 1$ $n >= 0$, $4n^3/2^n + 8n^2/2^n + 1 \geq 1$ □□□ $c = 1$ □ , $n0 = 0$

Question 3 :

Please determine a succinct big-Θ expression for the growth of the function $log(n^2) + n^2log(n^4) + 1000n^3 + 5000000n$. You don’t have to provide appropriate constants c1, c2, and n0 for the definition. However, please explain how to get your answer. (16□)

Answer :

□□ Big-Θ □□□□□ $c1g(x) \geq f(x) \geq c2g(x)$ □□ $f(x) = \Theta(g(x))$

□□□□□□□□□□□□□□□□□□□□□□ $f(x)$ □□□□□□□□□ □
$log(n^2) + n^2log(n^4) + 1000n^3 + 5000000n$ □□□□□□□□□ $1000n^3$ □□□□□□ $g(x)$ □ n^3□ □
□□□□□□□□□□□□□□ $c1, c2, n0$ □□□□□□□□ $\Theta(n^3)$□□□□□□□□□□□□□□□□□ n^3

Question 4 :

Analyze and give the time complexity of the following program segments in terms of n. Please briefly explain your answer. (24□)

4-1 :

```
int value = 0;
for(int i=0;i<n;i++)
    for(int j=0;j<i;j++)
        value += 1;
```

Answer of 4-1

```
 //  code                            Freq              Total Steps
int value = 0;                        1                     1
for(int i=0;i<n;i++)                  n+1                   n+1
    for(int j=0;j<i;j++)              i+1                  ((1+n)*n)/2 + 1
        value += 1;                   1                    ((1+n)*n)/2
```

Therefore, the time complexity = $((1+n)*n)/2$ = $O(n^2)$

4-2 :

```
for (int i = 1; i < n; i++) {
    i *= k;
}
```

Answer of 4-2

```
// code                               Freq              Total Steps
for (int i = 1; i < n; i++) {      logn / logk            logn / logk
    i *= k;                            1                  logn / logk
}
```

Therefore, the time complexity = $logn/logk$ = $O(log_k n)$

4-3 :

```
int i, j, k = 0;
for (i = n / 2; i <= n; i++) {
    for (j = 2; j <= n; j = j * 2) {
        k = k + n / 2;
    }
}
```

Answer of 4-3

```
// code                                    Freq           Total Steps
int i, j, k = 0;
for (i = n / 2; i <= n; i++) {             (n/2) + 1        (n/2) + 1
    for (j = 2; j <= n; j = j * 2) {       log(n)           log(n)*(n/2) + 1
        k = k + n / 2;                        1             log(n)*(n/2)
    }
}
```

Therefore, the time complexity = $log(n) * (n/2)$ = $O(nlogn)$

Question 5

(20[illegible]) (1) [illegible](bigO)[illegible] n [illegible] ([illegible] $O(n^2)$ [illegible] $O(5n^2)$)? (2) [illegible] $5n^3 + 4n + 2$ [illegible] $O(n^3)$ [illegible] $O(n^3 + n)$?

Answer

[illegible] n [illegible] n [illegible]

[illegible] Big-O Notation [illegible]

[illegible] n^3 [illegible] n [illegible] n^3 [illegible]

[illegible]

[illegible] ([illegible])

```
<script type="text/x-mathjax-config">
    MathJax.Hub.Config({ tex2jax: {inlineMath: [['$', '$']]}, messageStyle:
"none" });
</script>
```