Q1]

Algorithm QuickSort Descending(arr, low, high)

1. if low < high then
2. P ← partitionDesc(arr, low, high)
3. QuickSort Descending(arr, low, P-1)
4. QuickSort Descending(arr, P+1, high)
5. End if

Algorithm PartitionDesc(arr, low, high)

1. pivot ← arr[high]
2. i ← low - 1
3. for j ← Low to high - 1 do
4. if arr[j] ≥ pivot then
5. i ← i+1
6. swap(arr[i], arr[j])
7. End for
8. swap(arr[i+1], arr[high])
9. return i+1

Computing Time Complexity:

- Best Case: pivot always splits the list roughly in half

$$T(n) = 2T(n/2) + n$$

$a = 2, b = 2$

$$\log_b a = \log_2 2 = 1$$

$$n^1 \log^0 n = n^K \log^P n$$

$K = 1, P = 0$

$\log_2 2 = 1 = K = 1$ & $p > -1 \rightarrow \boxed{O(n \log n)}$ ↓Ans.

- Average case: random pivot.

$T(n) = 2T(n/2) + cn$

$a = 2, b = 2$

$n^1 \log^0 n = n^K \log^p n$

$K = 1, p = 0$

$\log_b a = \log_2 2 = 1 = K$ & $p > -1 \rightarrow O(n \log n)$.

- worst case: Pivot always ends up at one extreme

$T(n) = T(n-1) + cn$

$a = 1, b = 1 \longrightarrow O(n^2)$

$n^1 = n^K$

$K = 1$

Q2]

1) Algorithm Mergesort(arr)

1. if length(arr) ≤ 1 then
2. return A
3. mid ← floor(length(arr)/2)
4. L ← Mergesort(first half of arr)
5. R ← Mergesort(second half of arr)
6. return merge(L, R)

Algorithm Merge(L, R)

1. i ← 1, j ← 1
2. result ← empty list
3. while i ≤ length(L) and j ≤ length(R) do
4. if L[i] ≤ R[j] then

```
5.          append L[i] to result
6.          i ← i+1
7.      else
8.          append R[j] to result
9.          j ← j+1
10.  while i ≤ length(L) do
11.      append L[i] to result
12.      i ← i+1
13.  while j ≤ length(R) do
14.      append R[j] to result
15.      j ← j+1
16.  return result
```

2) [59, 6, 35, 12, 27, 9, 1, 18, 5, 31, 16]

1) split the original array:

Left: [59, 6, 35, 12, 27]

Right: [9, 1, 18, 5, 31, 16]

2) Splitting the left half:

L: [59, 6]

R: [35, 12, 27]

3) splitting [59, 6]:

L: [59] } merge → [6, 59]

R: [6] }

4) splitting [35, 12, 27]:

L: [35]

R: [12, 27] → further splitted into: [12], [27] → merge [12, 27]

merging [35] & [12, 27] → [12, 27, 35]

5) merging left side:

[6, 59] & [12, 27, 35] → merge → [6, 12, 27, 35, 59]

6) Now splitting the right side of the original array:

L: [9, 1, 18]

R: [5, 31, 16]

7) splitting the left array of the right side:

L: [9]

R: [1,18] → further splitted into: [1], [18] → merge → [1,18]

merging [9] & [1,18] → [1, 9, 18]

8) splitting [5, 31, 16]:

L: [5]

R: [31, 16] → further splitted into: [31], [16] → merge → [16,31]

merging [5] & [16,31] → [5,16,31]

9) merging Right side:

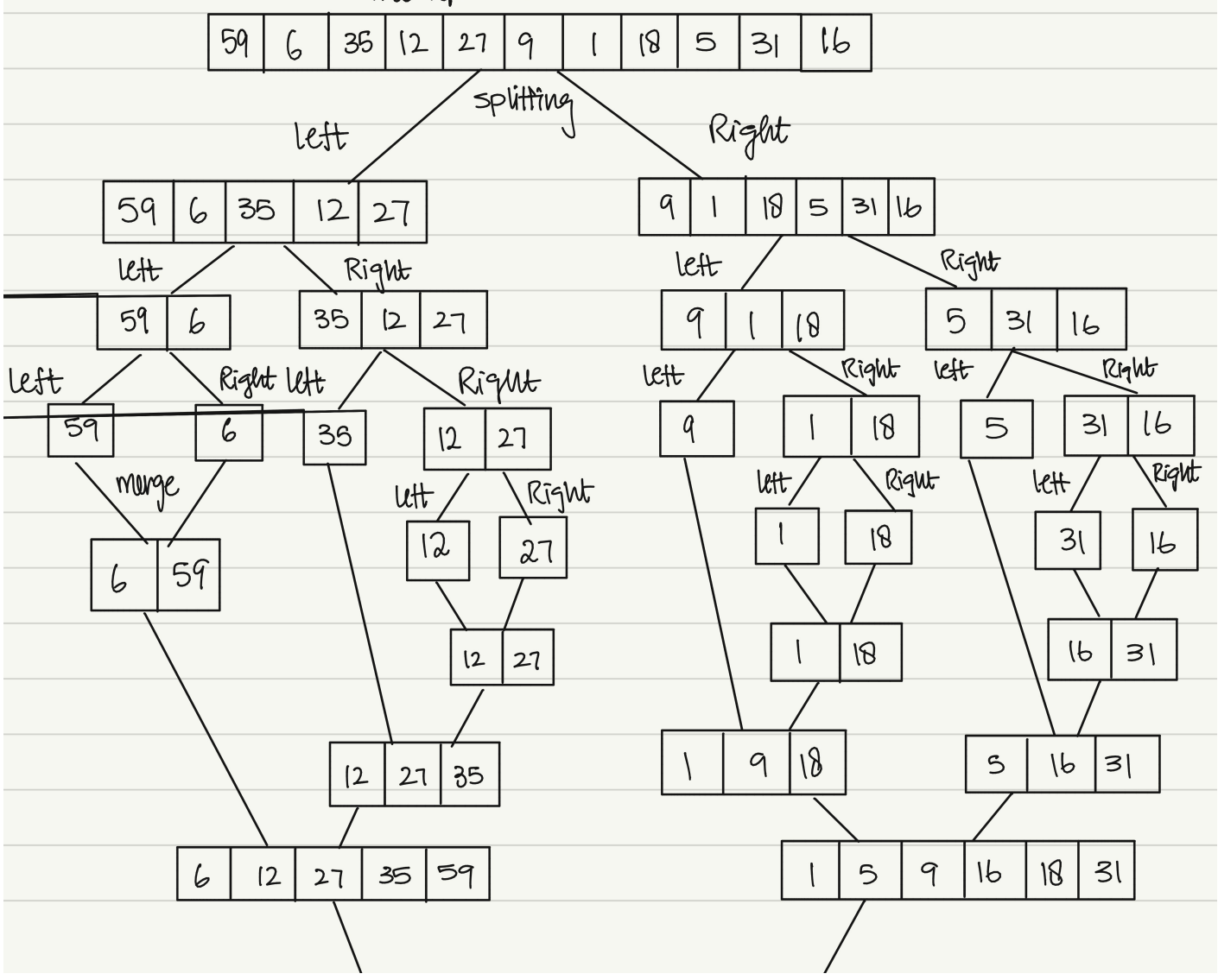
[1, 9, 18] & [5, 16, 31] → merge → [1, 5, 9, 16, 18, 31]

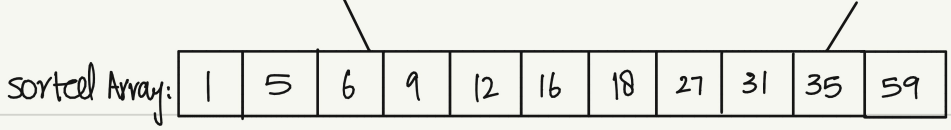
10) merging left & Right side:

[6, 12, 27, 35, 59] & [1, 5, 9, 16, 18, 31]

↳ merge → [1, 5, 6, 9, 12, 16, 18, 27, 31, 35, 59]

Tree representation:





3) Algorithm fourwaymergesort(A)

1. if length(A) ≤ 1 then
2. return A
3. n ← length(A)
4. A1 ← A[1...n/4]
5. A2 ← A[(n/4) + 1... n/2]
6. A3 ← A[(n/2) + 1... 3n/4]
7. A4 ← A[(3n/4) + 1... n]
8. L1 ← fourwaymergesort(A1)
9. L2 ← fourwaymergesort(A2)
10. L3 ← fourwaymergesort(A3)
11. L4 ← fourwaymergesort(A4)
12. return mergefour(L1, L2, L3, L4)

Algorithm Mergefour(L1, L2, L3, L4)

1. i ← 1, j ← 1, K ← 1, m ← 1
2. result ← empty list
3. while (i ≤ length(L1)) or (j ≤ length(L2)) or (K ≤ length(L3)) or (m ≤ length(L4)) do
4. minval ← min(L1[i], L2[j], L3[K], L4[m]
5. if minval = L1[i] then
6. append L1[i] to result
7. i ← i+1
8. else if minval = L2[j] then
9. append L2[j] to result.
10. j ← j+1
11. else if minval = L3[K] then
12. append L3[K] to result
13. K ← K+1
14. else

15. append L4[m] to result
16. m ← m+1
17. end while.
18. return result.

$T(n) = 4T(n/4) + n$ → modified recurrence relation.

Q3]

a) Algorithm MatrixMultiply(A, B, n)

1. For i ← 1 to n do
2. For j ← 1 to n do
3. C[i][j] ← 0
4. For k ← 1 to n do
5. C[i][j] ← C[i][j] + A[i][k] × B[k][j]
6. return C.

b) Algorithm strassen(A, B, n)

1. if n = 1 then
2. return A×B
3. else
4. Divide A & B into 2×2 submatrices: A11, A12, A21, A22 and B22, B11, B21, B12
5. $M_1 \leftarrow (A_{11} + A_{22})(B_{11} + B_{22})$
6. $M_2 \leftarrow (A_{21} + A_{22})(B_{11})$
7. $M_3 \leftarrow (A_{11})(B_{12} - B_{22})$
8. $M_4 \leftarrow (A_{22})(B_{21} - B_{11})$
9. $M_5 \leftarrow (A_{11} + A_{12})(B_{22})$
10. $M_6 \leftarrow (A_{21} - A_{11})(B_{11} + B_{12})$
11. $M_7 \leftarrow (A_{12} - A_{22})(B_{21} + B_{22})$
12. $C_{11} \leftarrow M_1 + M_4 - M_5 + M_7$
13. $C_{12} \leftarrow M_3 + M_5$
14. $C_{21} \leftarrow M_2 + M_4$

15. $C_{22} \leftarrow M_1 - M_2 + M_3 + M_6$

16. Combine $C_{11}, C_{12}, C_{21}, C_{22}$ into matrix C.
Return C.

c) $T(n) = 7T(n/2) + n^2$

d) $a = 7, \; b = 2$

$$\log_b a = \log_2 7 = 2.81$$

$$n^2 \log^0 n = n^K \log^P n$$

$K = 2, \; P = 0$

$$\log_b a > K \longrightarrow O(n^{\log_b a})$$

$$\boxed{O(n^{2.81})} \rightarrow \text{Ans.}$$

e) $O(n^3) \longrightarrow$ standard Matrix Multiplication

$O(n^{2.81}) \longrightarrow$ strassen's Matrix multiplication

∴ Therefore, strassen's algo is asymptotically faster then the standard method. For small matrices, the standard method may be faster (due to lower constant factor) but for large matrix, strassen's method provides significant improvement.

Q4]

1) for Decreasing functions:

$$T(n) = aT(n-b) + f(n)$$

$a > 0$ & $b > 0$ & $f(n) = O(n^k)$ where $k \geq 0$

1) if $a = 1$ then $O[n * f(n)]$

2) if $a > 1$ then $O[f(n)\, a^{n/b}]$

3) if $a < 1$ then $O[f(n)]$

2) for Dividing function:

$$T(n) = aT(n/b) + f(n)$$

$a > 0$, $b > 1$ & $f(n) = O(n^k \log^p n)$

1) if $\log_b a > k \longrightarrow O(n^{\log_b a})$

2) if $\log_b a = k$:

- if $p > -1$ (non-negative) $\longrightarrow O(n^k \log^{p+1} n)$
- if $p = -1 \longrightarrow O(n^k \log\log n)$
- if $p < -1 \longrightarrow O(n^k)$

3) if $\log_b a < k$:

- if $p \geq 0 \longrightarrow O(n^k \log^p n)$
- if $p < 0 \longrightarrow O(n^k)$.

Q5] yes, we can solve the modified relation in Q2 part 3.

$T(n) = 4T(n/4) + n$

$a=4, b=4 \longrightarrow \log_b a = \log_4 4 = 1$

$n^k \log^p n = n^k \log^p n$

$k=1, p=0$

$\log_b a = k \;\&\; p > -1 \longrightarrow O(n^k \log^{p+1} n)$

$\boxed{O(n \log n)} \rightarrow$ Ans.

Q6]

1) $T(n) = 2T(n/2) + n^2$

Iteration Method:

$T(n/2) = 2T(n/4) + n^2/4$

$T(n/4) = 2T(n/8) + n^2/16$

$T(n/8) = 2T(n/16) + n^2/64$

$n = 2^k$

$K = \log_2 n$

$T(n) = 2T(n/2) + n^2$

$T(n) = 2\left[2T(n/4) + n^2/4\right] + n^2$

$T(n) = 2^2 T(n/4) + n^2/2 + n^2$

$T(n) = 2^2\left[2T(n/8) + n^2/16\right] + n^2/2 + n^2$

$T(n) = 2^3 T(n/8) + n^2/4 + n^2/2 + n^2$

$T(n) = 2^3\left[2T(n/16) + n^2/64\right] + \frac{n^2}{4} + \frac{n^2}{2} + n^2$

$T(n) = 2^4 T(n/16) + \frac{n^2}{8} + \frac{n^2}{4} + \frac{n^2}{2} + n^2$

$\vdots$

$T(n) = 2^k T(n/2^k) + \frac{n^2}{2^{k-1}} + \frac{n^2}{2^{k-2}} + \ldots n^2$

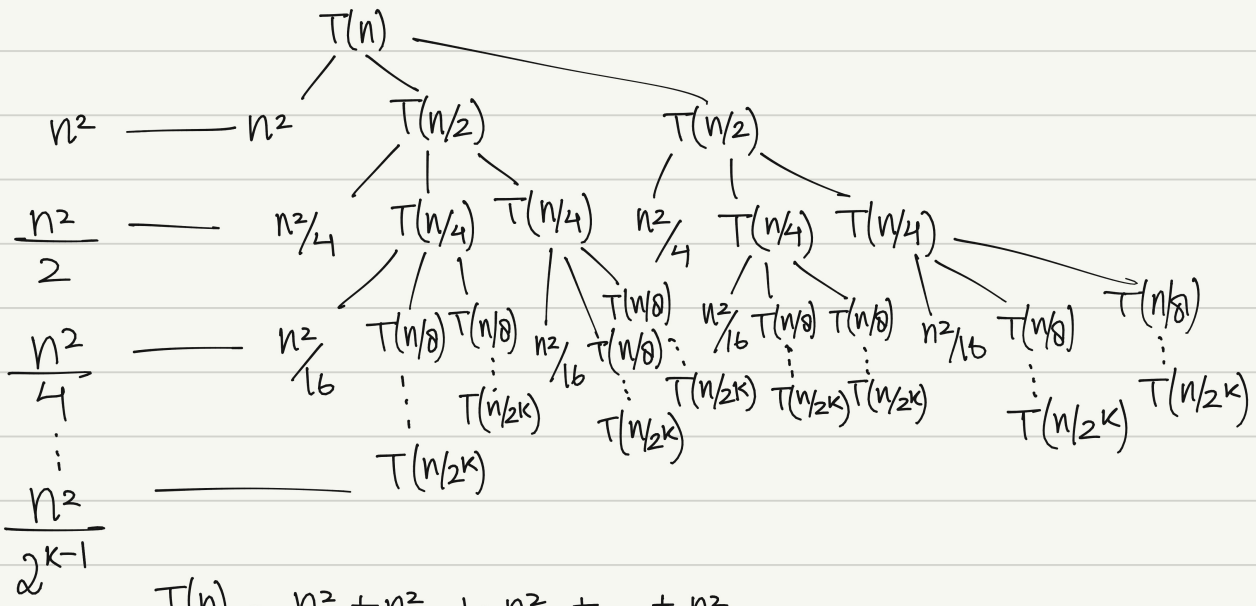
$T(n) = 2^{\log_2 n} T(n/n) + 2n^2$

$T(n) = n^{\log_2 2} T(1) + 2n^2$

$T(n) = n + 2n^2$

$\boxed{O(n^2)} \rightarrow$ Ans.

Recurrence Tree Method:



$$T(n) = n^2 + \frac{n^2}{2} + \frac{n^2}{4} + \ldots + \frac{n^2}{2^{k-1}}$$

$$T(n) = n\left[1 + 1/2 + 1/4 + \ldots + 1/2^{k-1}\right]$$

$$n^2\left[1, 1/2, 1/4, \ldots, 1/2^{k-1}\right]$$

$$S_\infty = \frac{1}{1 - 1/2} = 2$$

$$T(n) = n^2(2)$$

$$T(n) = 2n^2$$

$$\boxed{O(n^2)} \rightarrow \text{Ans.}$$

11) $T(n) = 3T(n/3) + n\log^2 n$

Iteration Method:

$T(n/3) = 3T(n/9) + n/3 \log^2 n/3$

$T(n/9) = 3T(n/27) + n/9 \log^2 n/9$

$T(n/27) = 3T(n/81) + n/27 \log^2 n/27$

$n = 3^k$

$\log_3 n = k$

$T(n) = 3T(n/3) + n\log^2 n$

$T(n) = 3\left[3T(n/9) + n/3 \log^2 n/3\right] + n\log^2 n$

$T(n) = 3^2 T(n/9) + n\log^2 n/3 + n\log^2 n$

$T(n) = 3^2\left[3T(n/27) + n/9 \log^2 n/9\right] + n\log^2 n/3 + n\log^2 n$

$T(n) = 3^3 T(n/27) + n\log^2 n/9 + n\log^2 n/3 + n\log^2 n$

$T(n) = 3^3\left[3T(n/81) + n/27 \log^2 n/27\right] + n\log^2 n/9 + n\log^2 n/3 + n\log^2 n$

$T(n) = 3^4 T(n/81) + n\log^2 n/27 + n\log^2 n/9 + n\log^2 n/3 + n\log^2 n$

$\vdots$

$T(n) = 3^k T(n/3^k) + n\log^2 n/3^{k-1} + n\log^2 n/3^{k-2} + \ldots + n\log^2 n$

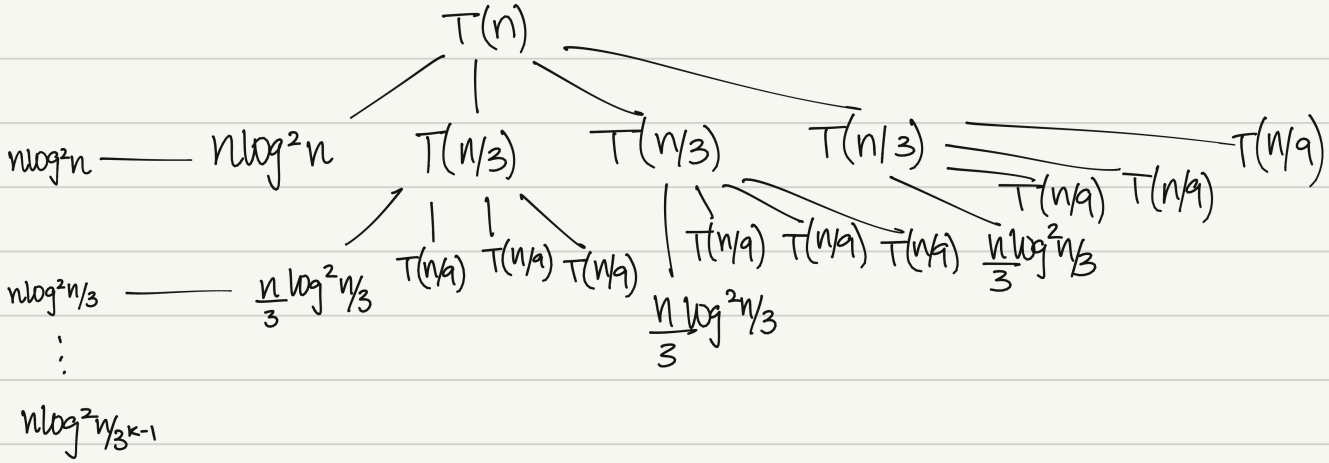
$T(n) = 3^k T(n/n) + n\log^2 n/3^{k-1} + n\log^2 n/3^{k-2} + \ldots + n\log^2 n$

$T(n) = 3^{\log_3 n} T(1) + n\log^3 n$

$T(n) = n + n\log^3 n$

$\boxed{O(n\log^3 n)} \rightarrow$ Ans.

Recurrence Tree Method:



$T(n) = n\log^2 n + n\log^2 n/3 + n\log^2 n/9 + \ldots + n\log^2 n/3^{k-1}$

$T(n) = n\left[\log^2 n + \log^2 n/3 + \log^2 n/9 + \ldots + \log^2 n/3^{k-1}\right]$

$T(n) = n\log^3 n$

$\boxed{O(n\log^3 n)} \rightarrow$ Ans.

III) $T(n) = 2T(n/2) + \log n$

Iteration Method:

$T(n/2) = 2T(n/4) + \log n/2$

$T(n/4) = 2T(n/8) + \log n/4$

$n = 2^k$

$\log_2 n = k$

$T(n) = 2T(n/2) + \log n$

$T(n) = 2[2T(n/4) + \log n/2] + \log n$

$T(n) = 2^2 T(n/4) + 2\log n/2 + \log n$

$T(n) = 2^2[2T(n/8) + \log n/4] + 2\log n/2 + \log n$

$T(n) = 2^3 T(n/8) + 4\log n/4 + 2\log n/2 + \log n$

$\vdots$

$T(n) = 2^k T(n/2^k) + 2^{k-1}\log n/2^{k-1} + \ldots + 2\log n/2 + \log n$

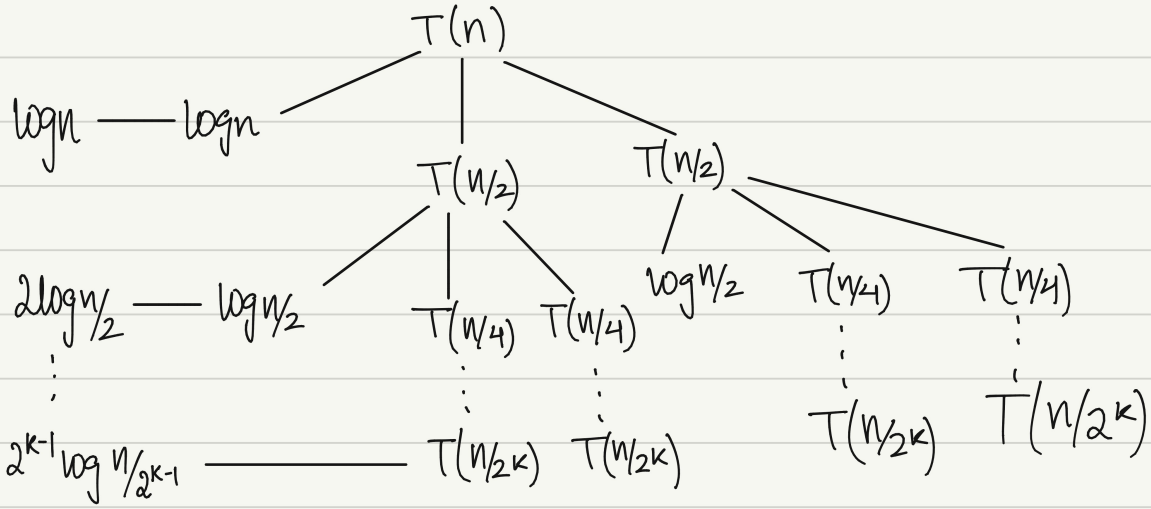
$T(n) = 2^{\log_2 n} T(n/n) + 2^{k-1}\log n/2^{k-1} + \ldots + 2\log n/2 + \log n$

$T(n) = n + n$

$T(n) = 2n$

$\boxed{O(n)} \rightarrow$ Ans.

Recurrence Tree Method:



$T(n) = \log n + 2\log n/2 + 4\log n/4 + \ldots + 2^{k-1}\log n/2^{k-1}$

$T(n) = n$

$\boxed{O(n)} \rightarrow$ Ans.

Q7]

i) $T(n) = 4T(n/2) + n^2$

$a = 4, b = 2$

$\log_b a = \log_2 4 = 2$

$n^2 \log^0 n = n^K \log^P n$

$K = 2, P = 0$

$\log_b a = K$ & $P > -1 \longrightarrow O(n^K \log^{P+1} n)$

$O(n^2 \log n)$ → Ans.

ii) $T(n) = 2T(n/4) + \sqrt{n}$

$a = 2, b = 4$

$\log_b a = \log_4 2 = 0.5$

$n^{1/2} \log^0 n = n^K \log^P n$

$K = 1/2, P = 0$

$\log_b a = K$ & $P > -1 \longrightarrow O(n^K \log^{P+1} n)$

$O(n^{1/2} \log n)$ → Ans.

iii) $T(n) = 4T(n/2) + n^2 \log n$

$a = 4, b = 2$

$\log_b a = \log_2 4 = 2$

$n^2 \log^1 n = n^K \log^P n$

$K = 2, P = 1$

$\log_b a = K$ & $P > -1 \longrightarrow O(n^K \log^{P+1} n)$

$O(n^2 \log^2 n)$ → Ans.

Q8]

$T(n) = 4T(n/2) + n^2$

$T(n/2) = 4T(n/4) + n^2/4$

$T(n/4) = 4T(n/8) + n^2/16$

$n = 2^k$

$\log_2 n = k$

$T(n) = 4T(n/2) + n^2$

$T(n) = 4\left[4T(n/4) + n^2/4\right] + n^2$

$T(n) = 4^2 T(n/4) + 2n^2$

$T(n) = 4^2\left[4T(n/8) + n^2/16\right] + 2n^2$

$T(n) = 4^3 T(n/8) + 3n^2$

$\vdots$

$T(n) = 4^k T(n/2^k) + kn^2$

$T(n) = 4^{\log_2 n} T(n/n) + n^2 \log_2 n$

$T(n) = n^2 + n^2 \log_2 n$

$\boxed{O(n^2 \log n)}$

$\therefore$ therefore, both guesses are incorrect.