CS-2009 Design and Analysis of Algorithms

Assignment One

22L-6552

Section: 4H

22L-6552 4H

Q1) a)

For View Data function:

Outer Loop: N iterations
Inner Loop: $Log_2(N)$ iterations

⇒ Overall: $NLog(N)$ iterations

Recurrence:

$$T(N) = T(N-1) + T(N-1) + \Theta(NLog(N))$$

$$T(N) = 2T(N-1) + \Theta(NLog(N))$$

Q1) b) For Solve function:

Outer Loop: N iterations

Inner Loop: $1 + 2 + 4 + 5 - - - + N$

$$= \frac{N}{2}(N+1) \rightarrow \text{Formula for sum of arithematic series.}$$

$$= \frac{N^2}{2} + \frac{N}{2}$$

⇒ Overall: $N + \frac{N^2}{2} + \frac{N}{2}$

$$\Rightarrow \Theta(N^2)$$

Recurrence:

$$T(N) = 16\,T\left(\frac{N}{2}\right) + O(N^2)$$

Q1) c) The FOR Loop runs N times

Recurrence:

$$T(N) = T\left(\frac{2N}{3}\right) + T\left(\frac{N}{5}\right) + O(N)$$

Q1) d) Code other than the recursive calls takes constant time.

Recurrence:

$$T(N) = 3T\left(\frac{2N}{3}\right) + O(1)$$

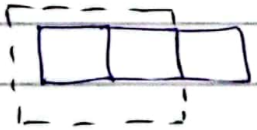
The algorithm compares the first and the last index positions of the array and swaps them if required. It then makes a recursive call for the first $\frac{2}{3}^{rd}$ part of the array which sorts the first $\frac{2}{3}^{rd}$ part of the array.

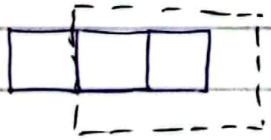
Then it makes a recursive call to sort last $\frac{2}{3}^{rd}$ part of the array. Now the first $\frac{2}{3}^{rd}$ part and the last $\frac{2}{3}^{rd}$ part of the array

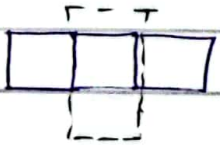
22L-6552 4H

is sorted but we are uncertain about the central $\frac{1}{3}^{rd}$ part of the ~~arr~~ array.

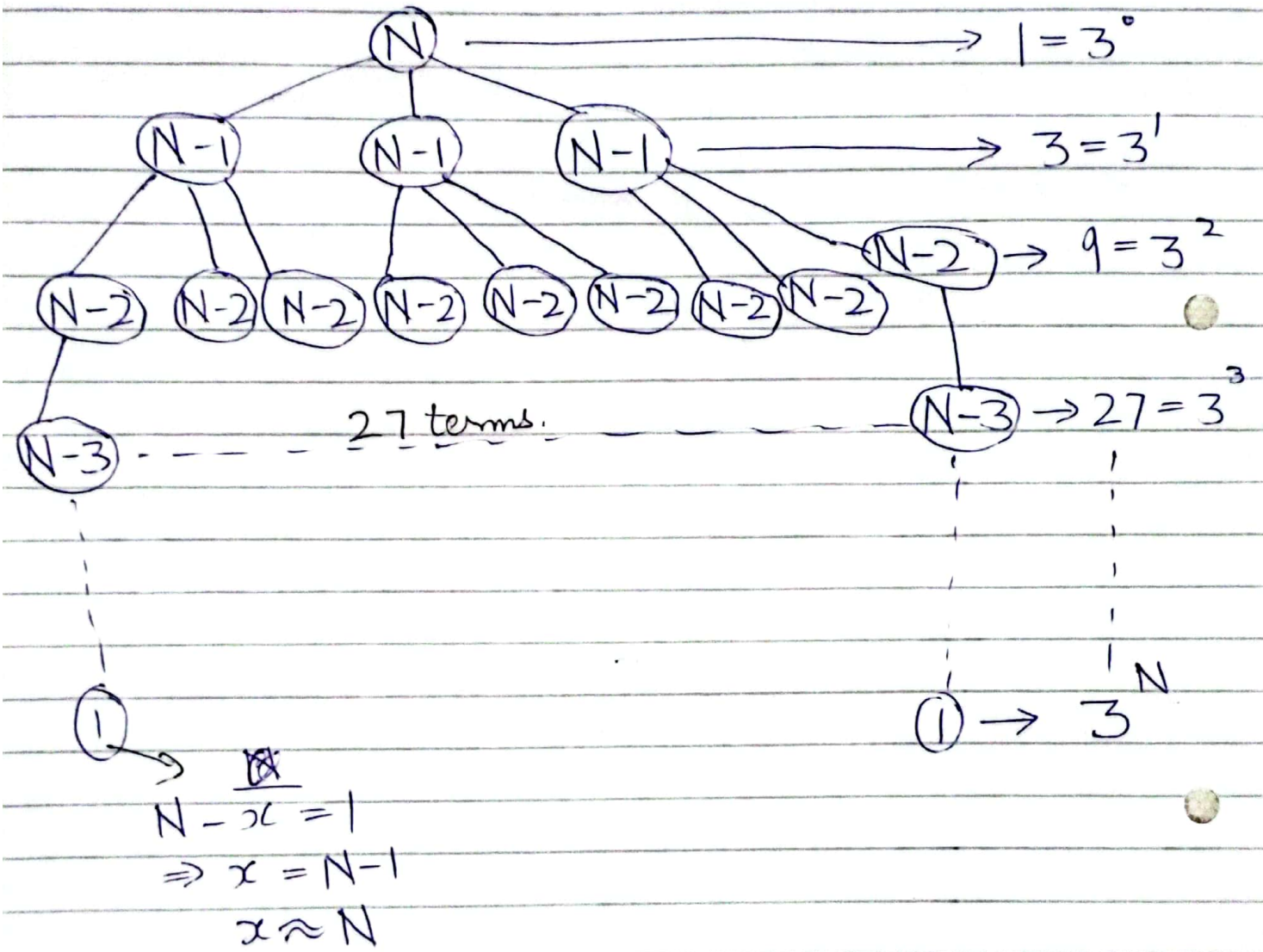
So we make a call again for the first $\frac{2}{3}^{rd}$ part of the array so that if anything is wrong in the middle $\frac{1}{3}^{rd}$ part due to the first two calls, it can be rectified.

First call: 

Second call: 

Third call:  to cater for 

Q2) a) a) $T(N) = 3T(N-1) + \Theta(1)$



$N - x = 1$

$\Rightarrow x = N - 1$

$x \approx N$

A geometric series is formed with N terms.

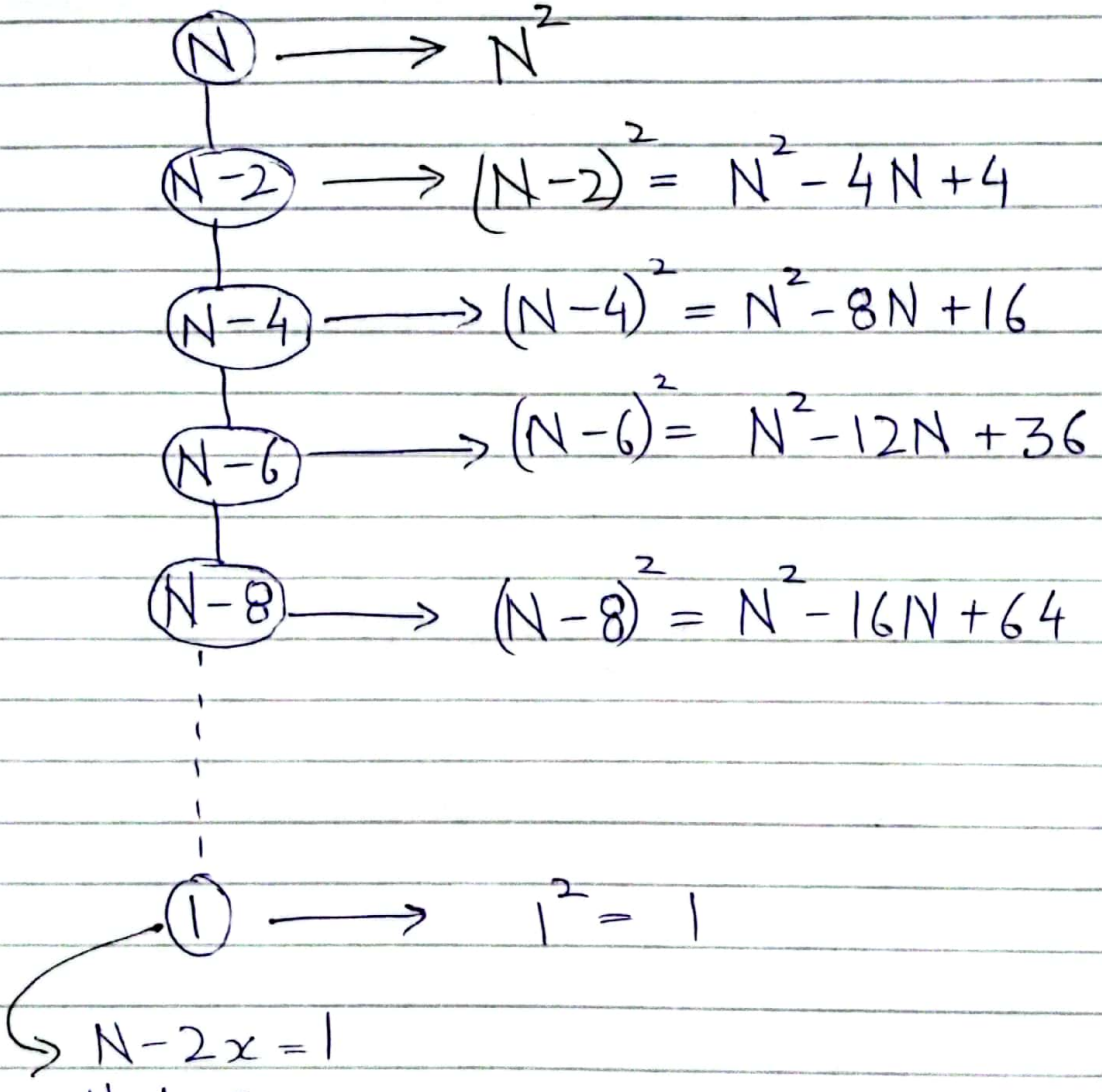
Sum

$$\rightarrow S = \frac{a(r^n - 1)}{r - 1} = \frac{1(3^N - 1)}{3 - 1} = \frac{3^N - 1}{2} = \frac{3^N}{2} - \frac{1}{2}$$

$$\Rightarrow T(N) = \frac{3^N}{2} - \frac{1}{2} \Rightarrow \Theta(3^N)$$

22L-6552 4H

(Q2) a) b) $T(N) = T(N-2) + \Theta(N^2)$



$N - 2x = 1$

$N - 1 = 2x$

$x = \frac{N-1}{2} \approx \frac{N}{2}$ Levels

Overestimation: Assumming that the cost at each level is N^2

$T(N)$ = Cost at each level * height of tree

$= N^2 * \frac{N}{2} = \frac{N^3}{2} \Rightarrow O(N^3)$

Another approach is to use some formula.
If N is even then each sub problem will also be even since we decrement by 2 and so we have a formula for sum of squares of even numbers

↳ Sum of squares of even numbers $= \frac{2n(n+1)(2n+1)}{3}$

$$\Rightarrow T(N) = \frac{2N(N+1)(2N+1)}{3}$$

$$\Rightarrow O(N^3)$$

If N is odd then each sub problem will also be odd since we decrement by 2 and so we have a formula for sum of squares of odd numbers

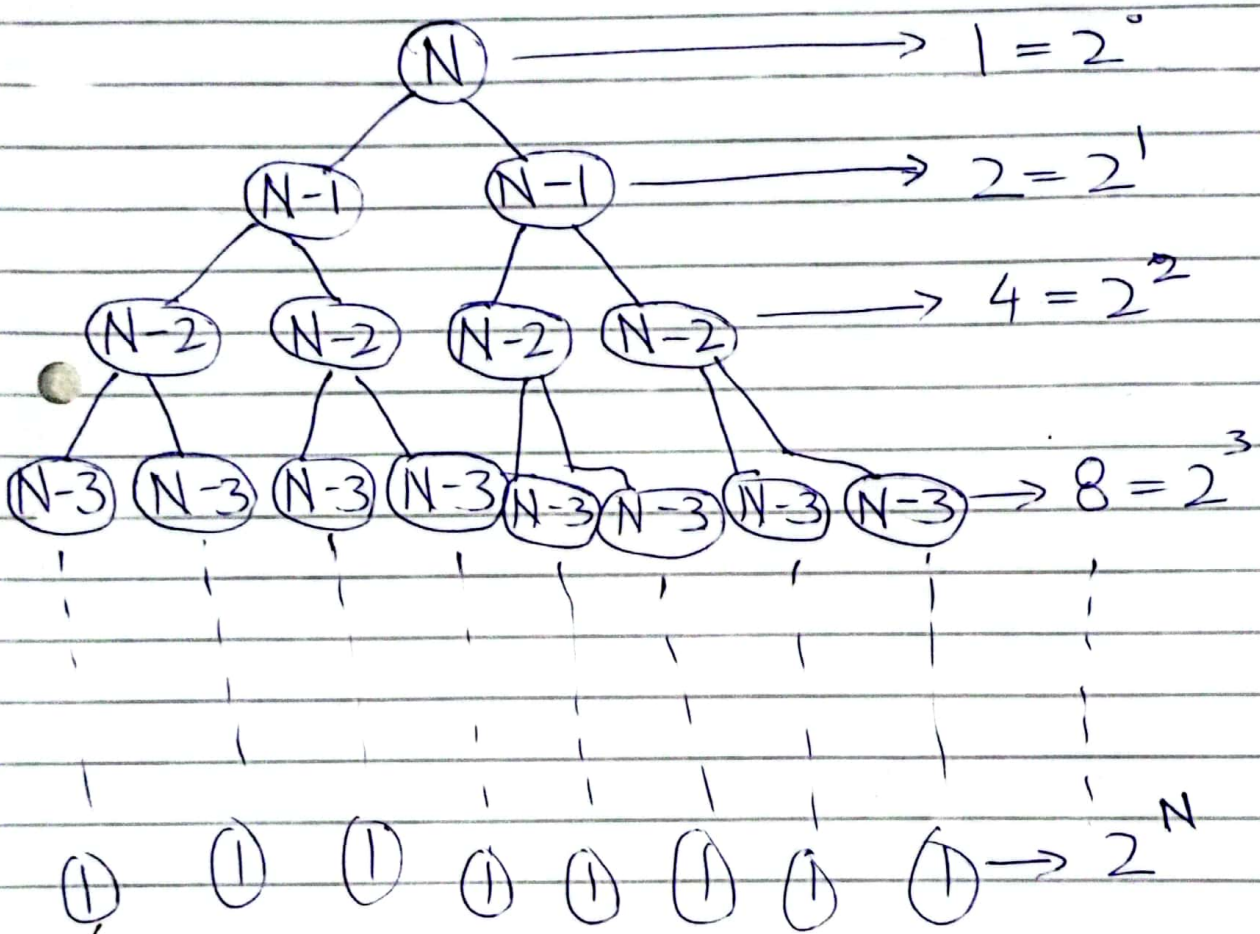
↳ Sum of squares of odd numbers $= \frac{n(2n+1)(2n-1)}{3}$

$$T(N) = \frac{N(2N+1)(2N-1)}{3}$$

$$\Rightarrow O(N^3)$$

22L-6552 4H

Q2) a) c) $T(N) = 2T(N-1) + \Theta(1)$



$N - x = 1$

$\Rightarrow x = N - 1$

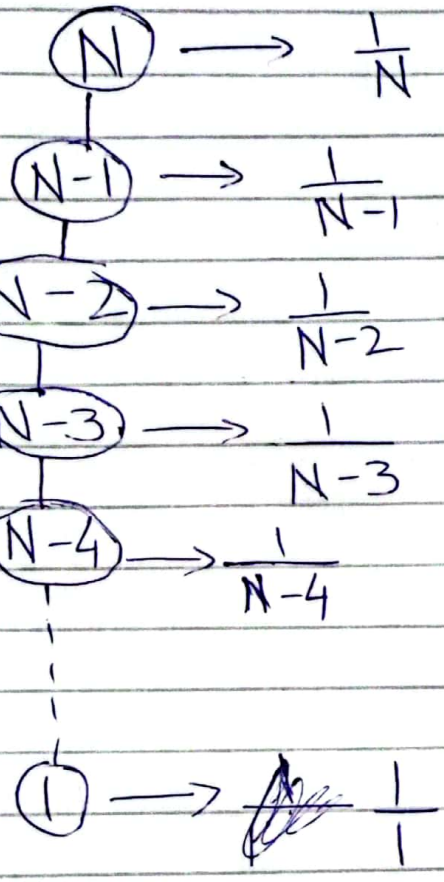
$x \approx N$

A geometric series is formed with N terms

$$\text{Sum} = \frac{a(r^n - 1)}{r - 1} = \frac{1(2^N - 1)}{2 - 1} = 2^N - 1$$

$$T(N) = 2^N - 1 \Rightarrow O(2^N)$$

Q2) a) d) $T(N) = T(N-1) + O\left(\frac{1}{N}\right)$



A Harmonic series is formed:

$$H_n = \frac{1}{1} + \frac{1}{2} + \frac{1}{3} + \frac{1}{4} + \frac{1}{5} + \cdots \frac{1}{N}$$

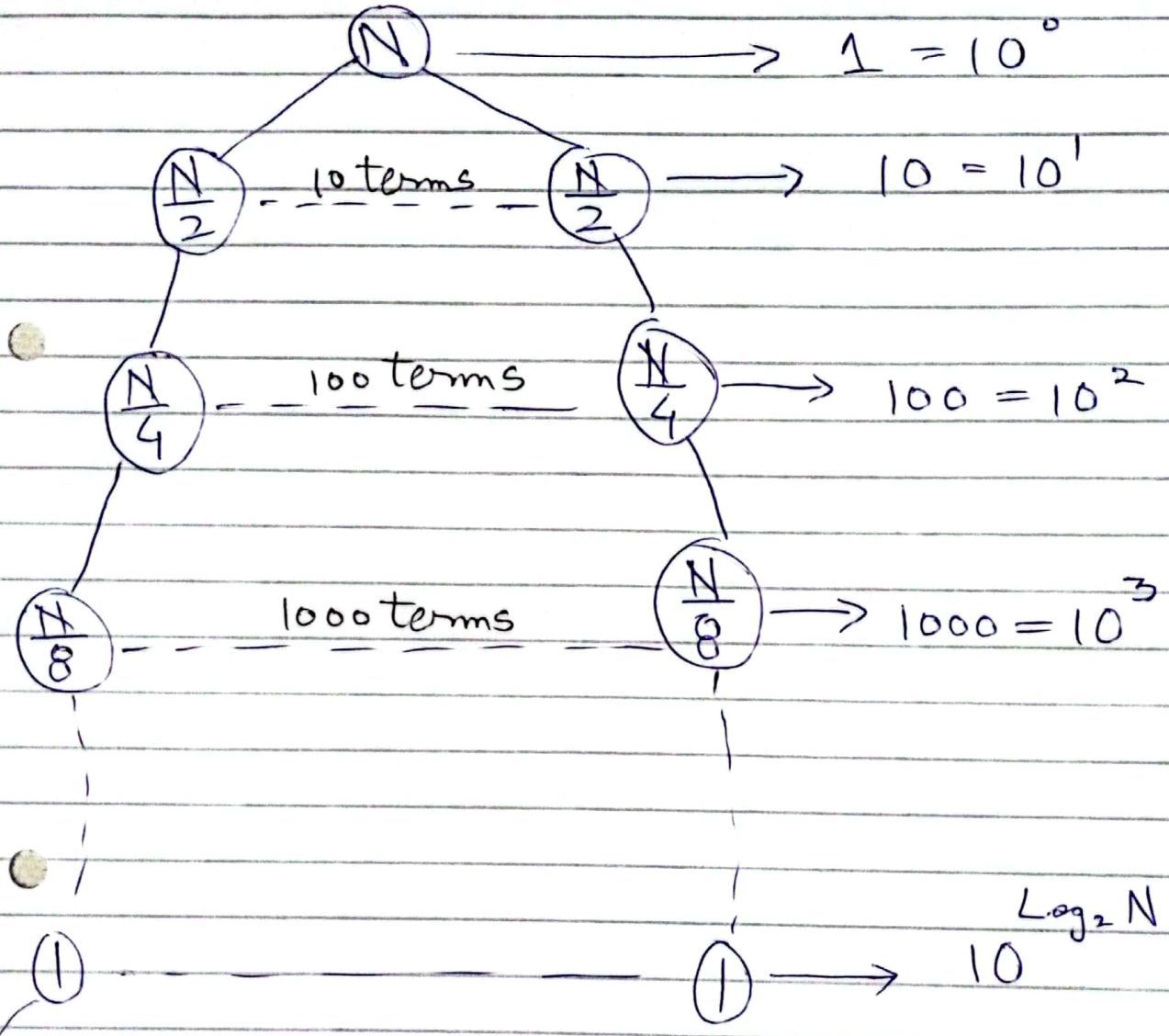
$$= \sum_{K=1}^{N} \frac{1}{K}$$

$= Ln(N) + O(1)$ → Expression used from Book & page 1142.

$\Rightarrow O(Log(N))$

22L-6552 4H

Q2) b) f) $T(N) = 10T(\frac{N}{2}) + \Theta(1)$



$\frac{N}{2^K} = 1$

$N = 2^K$

$Log_2 N = Log_2 2^K$

$Log_2 N = K Log_2 2$

$K = Log_2 N$

A geometric series is formed with $Log_2 N$ terms

$$Sum = \frac{a(r^n - 1)}{r - 1} = \frac{1(10^{Log_2 N} - 1)}{10 - 1} = \frac{10^{Log_2 N} - 1}{9}$$

$$T(N) = \frac{10^{Log_2 N} - 1}{9}$$

$$\Rightarrow O(10^{Log_2 N})$$

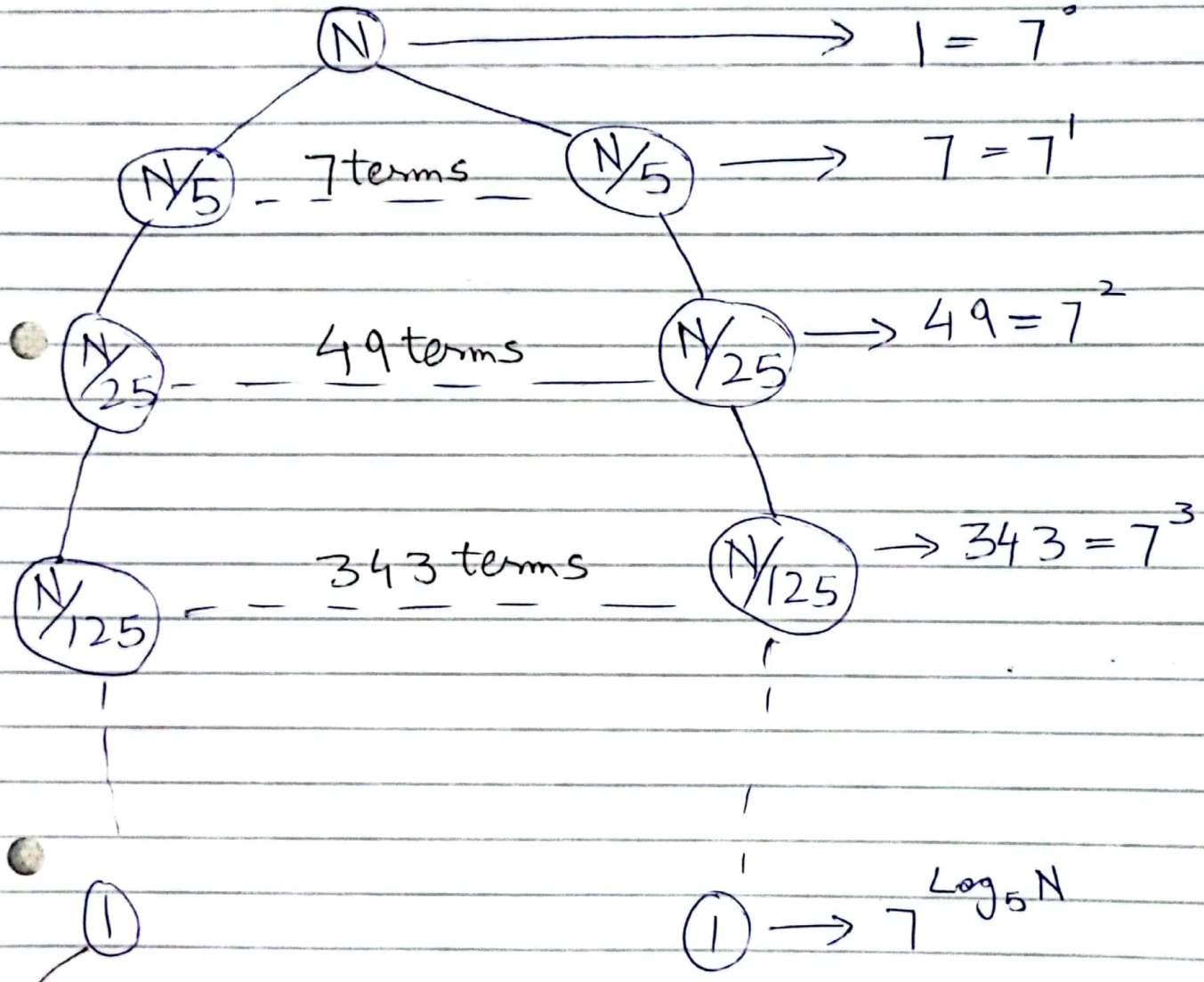
$$\Rightarrow O(N^{Log_2(10)})$$

$$\Rightarrow O(N^{3.32})$$ We can round/ciel and still we will get ~~an~~ on upper bound

So $O(N^4)$

22L-6552 4H

Q2) b) K) $T(N) = 7T\left(\frac{N}{5}\right) + \Theta(1)$



$\frac{N}{5^K} = 1$

$N = 5^K$

$\text{Log}_5 N = \text{Log}_5 5^K$

$\text{Log}_5 N = K \text{Log}_5(5)$

$\text{Log}_5 N = K$

A geometric series is formed with $\log_5 N$ terms

$$\text{Sum} = \frac{a(r^n - 1)}{r - 1} = \frac{1\left(7^{\log_5 N} - 1\right)}{7 - 1}$$

$$= \frac{7^{\log_5 N} - 1}{6}$$

$$T(N) = \frac{7^{\log_5 N} - 1}{6}$$

$$\Rightarrow O\left(7^{\log_5 N}\right)$$

$$\Rightarrow O\left(N^{\log_5 (7)}\right)$$

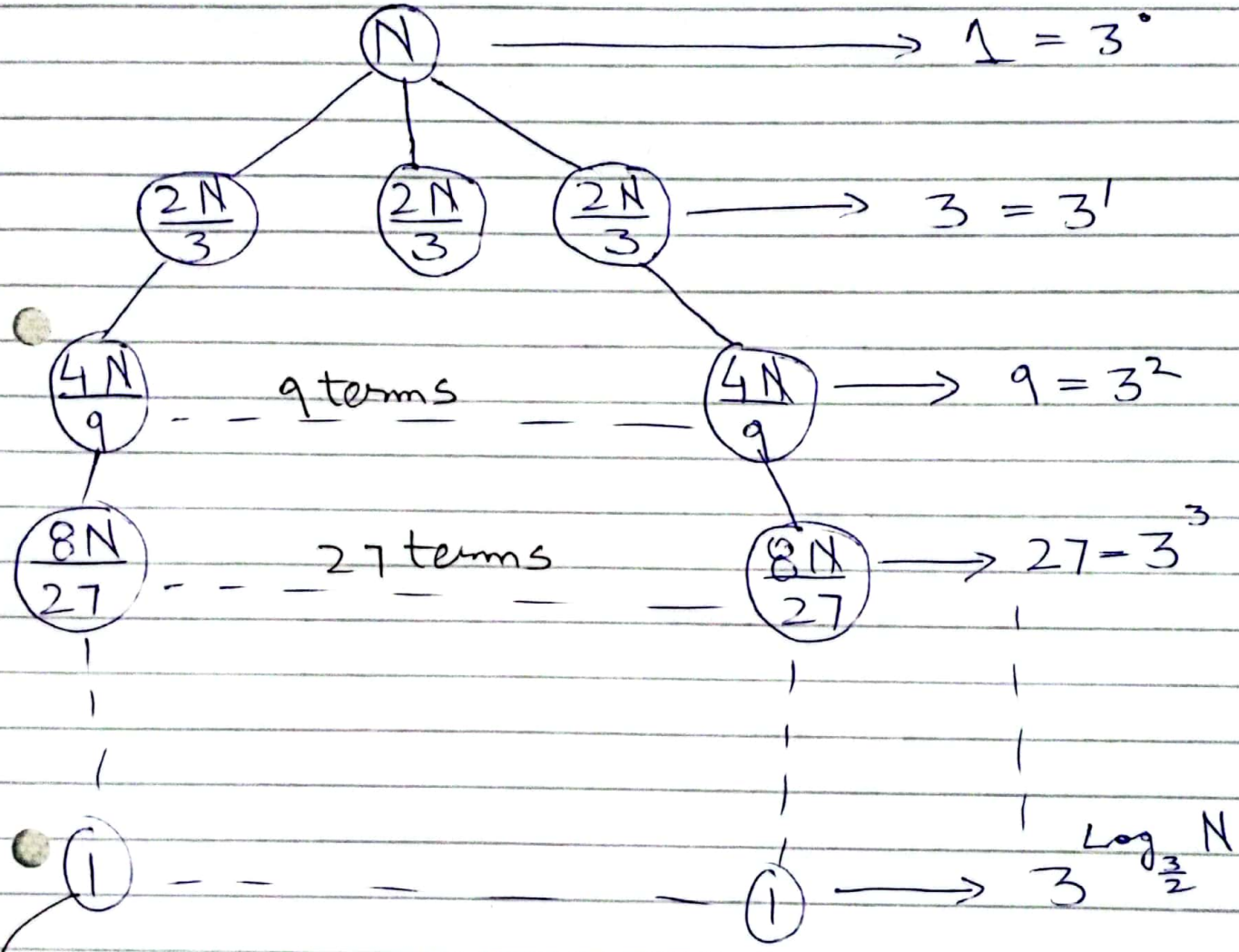
$$\Rightarrow O\left(N^{1.21}\right)$$

We can round/ciel and still we will get an upper bound.

So $O(N^2)$

22L-6552 4H

Q2) b) m) $T(N) = 3T\left(\frac{2N}{3}\right) + O(1)$



$\frac{N}{(3/2)^K} = 1$

$N = \left(\frac{3}{2}\right)^K$

$\text{Log}_{\frac{3}{2}} N = \text{Log}_{\frac{3}{2}} \left(\frac{3}{2}\right)^K$

$\text{Log}_{\frac{3}{2}}(N) = K \text{Log}_{\frac{3}{2}}\left(\frac{3}{2}\right)$

$K = \text{Log}_{\frac{3}{2}}(N)$

A geometric series is formed with $\text{Log}_{\frac{3}{2}} N$ terms

$$\text{Sum} = \frac{a(r^n - 1)}{r - 1} = \frac{1\left(3^{\text{Log}_{\frac{3}{2}}(N)} - 1\right)}{3 - 1}$$

$$= \frac{3^{\text{Log}_{\frac{3}{2}}(N)} - 1}{2}$$

$$T(N) = \frac{3^{\text{Log}_{\frac{3}{2}}(N)} - 1}{2}$$

$$\Rightarrow O\left(3^{\text{Log}_{\frac{3}{2}} N}\right)$$

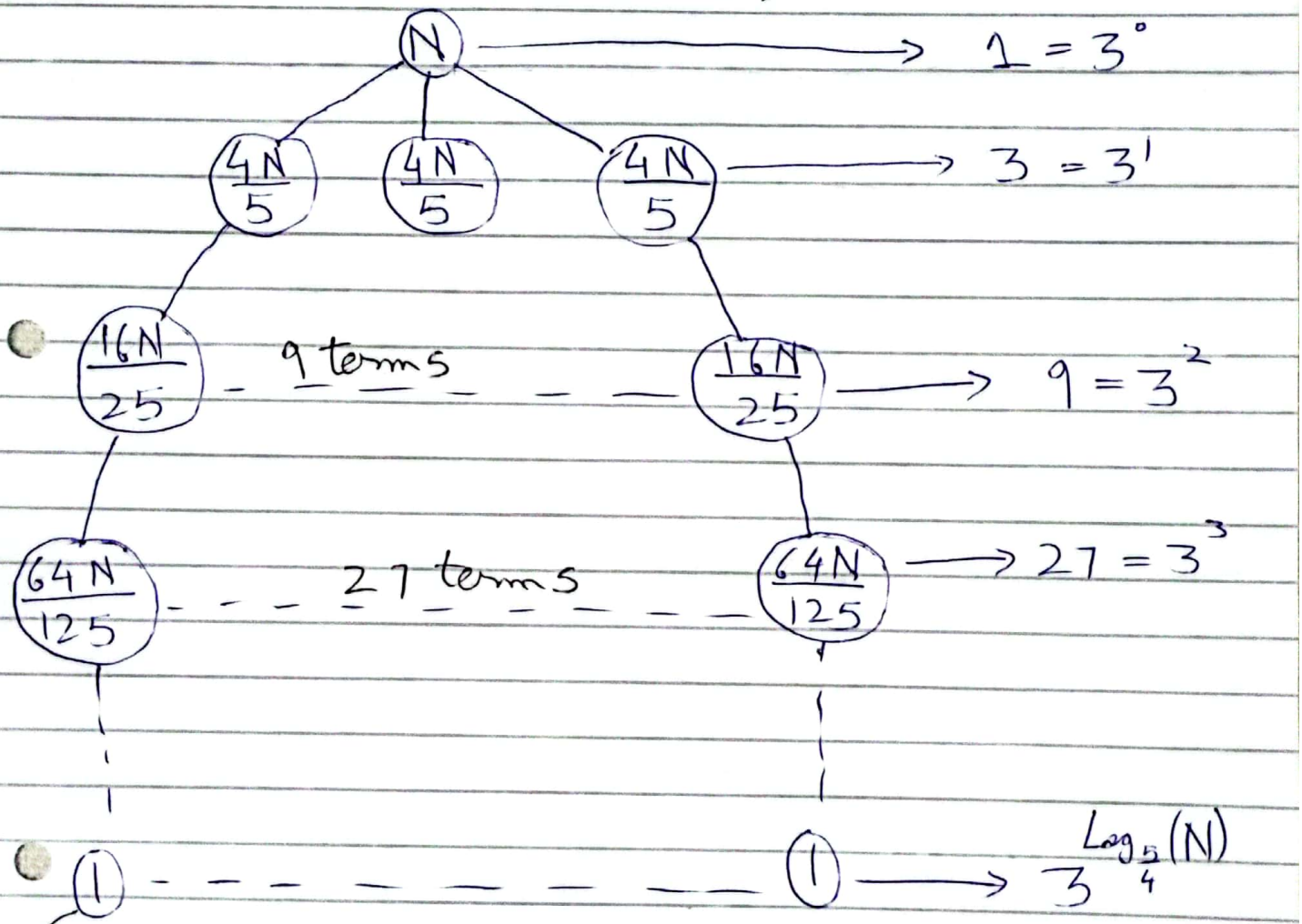
$$\Rightarrow O\left(N^{\text{Log}_{\frac{3}{2}}(3)}\right)$$

$$\Rightarrow O\left(N^{2.71}\right)$$ We can round/ ciel and still we will get an upper bound

So $O(N^3)$

22L-6552 4H

Q2) b) n) $T(N) = 3T\left(\frac{4N}{5}\right) + \Theta(1)$



$\frac{N}{(5/4)^K} = 1$

$N = \left(\frac{5}{4}\right)^K$

$\text{Log}_{\frac{5}{4}} N = \text{Log}_{\frac{5}{4}} \left(\frac{5}{4}\right)^K$

$\text{Log}_{\frac{5}{4}} N = K \, \text{Log}_{\frac{5}{4}} \left(\frac{5}{4}\right)$

$\Rightarrow K = \text{Log}_{\frac{5}{4}} (N)$

A geometric series is formed with $\log_{\frac{5}{4}} N$ terms

$$\text{Sum} = \frac{a(r^n - 1)}{r - 1} = \frac{1(3^{\log_{5/4} N} - 1)}{3 - 1}$$

$$= \frac{3^{\log_{\frac{5}{4}} N} - 1}{2}$$

$$T(N) = \frac{3^{\log_{\frac{5}{4}} N} - 1}{2}$$

$$\Rightarrow O(3^{\log_{\frac{5}{4}} N})$$

$$\Rightarrow O(N^{\log_{\frac{5}{4}}(3)})$$

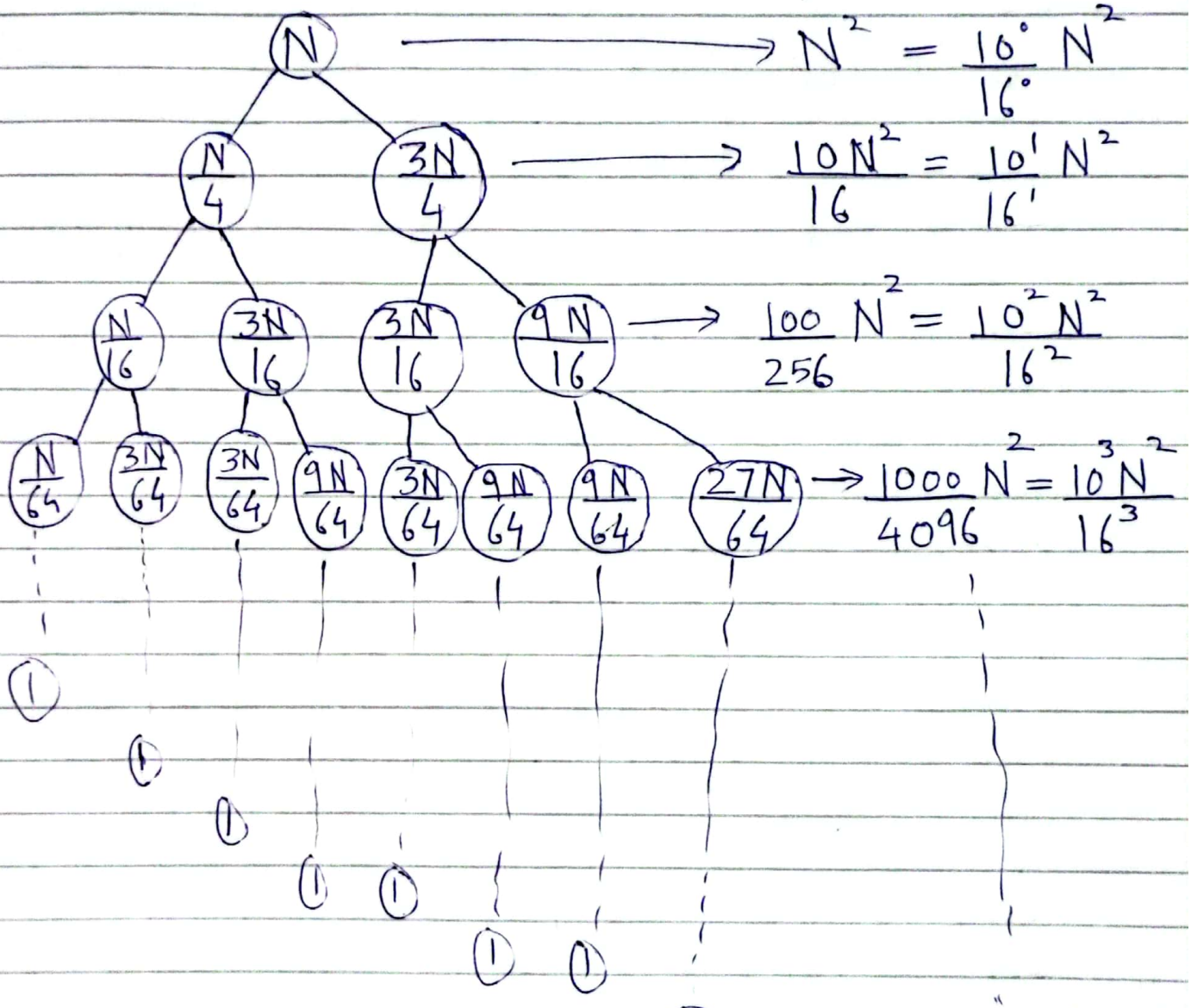
$$\Rightarrow O(N^{4.92})$$

We can round/ciel and still we will get an upper bound

So $O(N^5)$

22L-6552 4H

Q2) c) a) $T(N) = T\left(\frac{N}{4}\right) + T\left(\frac{3N}{4}\right) + O(N^2)$



$N^2 = \frac{10^0}{16^0} N^2$

$\frac{10N^2}{16} = \frac{10^1}{16^1} N^2$

$\frac{100}{256} N^2 = \frac{10^2 N^2}{16^2}$

$\frac{1000}{4096} N^2 = \frac{10^3 N^2}{16^3}$

$\frac{N}{4^k} = 1$

$N = \frac{4}{3}^k$

$\text{Log}_{\frac{4}{3}} N = \text{Log}_{\frac{4}{3}} K$

$\Rightarrow K = \text{Log}_{\frac{4}{3}}(N)$

$\frac{10^{\text{Log}_{\frac{4}{3}}(N)}}{16^{\text{Log}_{\frac{4}{3}}(N)}} N^2$

approximately

$$\text{Sum} = N^2 + \frac{10}{16}N^2 + \left(\frac{10}{16}\right)^2 N^2 + \left(\frac{10}{16}\right)^3 + \left(\frac{10}{16}\right)^4 + \cdots + \left(\frac{10}{16}\right)^{\log_{\frac{4}{3}} N} N^2$$

$$= N^2\left(1 + \frac{10}{16} + \left(\frac{10}{16}\right)^2 + \left(\frac{10}{16}\right)^3 + \left(\frac{10}{16}\right)^4 + \cdots + \left(\frac{10}{16}\right)^{\log_{\frac{4}{3}} N}\right)$$

→ This is a geometric series with $r < 1$ i.e $\frac{10}{16} = \frac{5}{8} < 1$

We can compute the sum to infinity using the formula: $\frac{a}{1-r} = \frac{1}{1-(5/8)} = \frac{8}{3}$

$$\Rightarrow \text{Sum} = N^2\left(\frac{8}{3}\right)$$

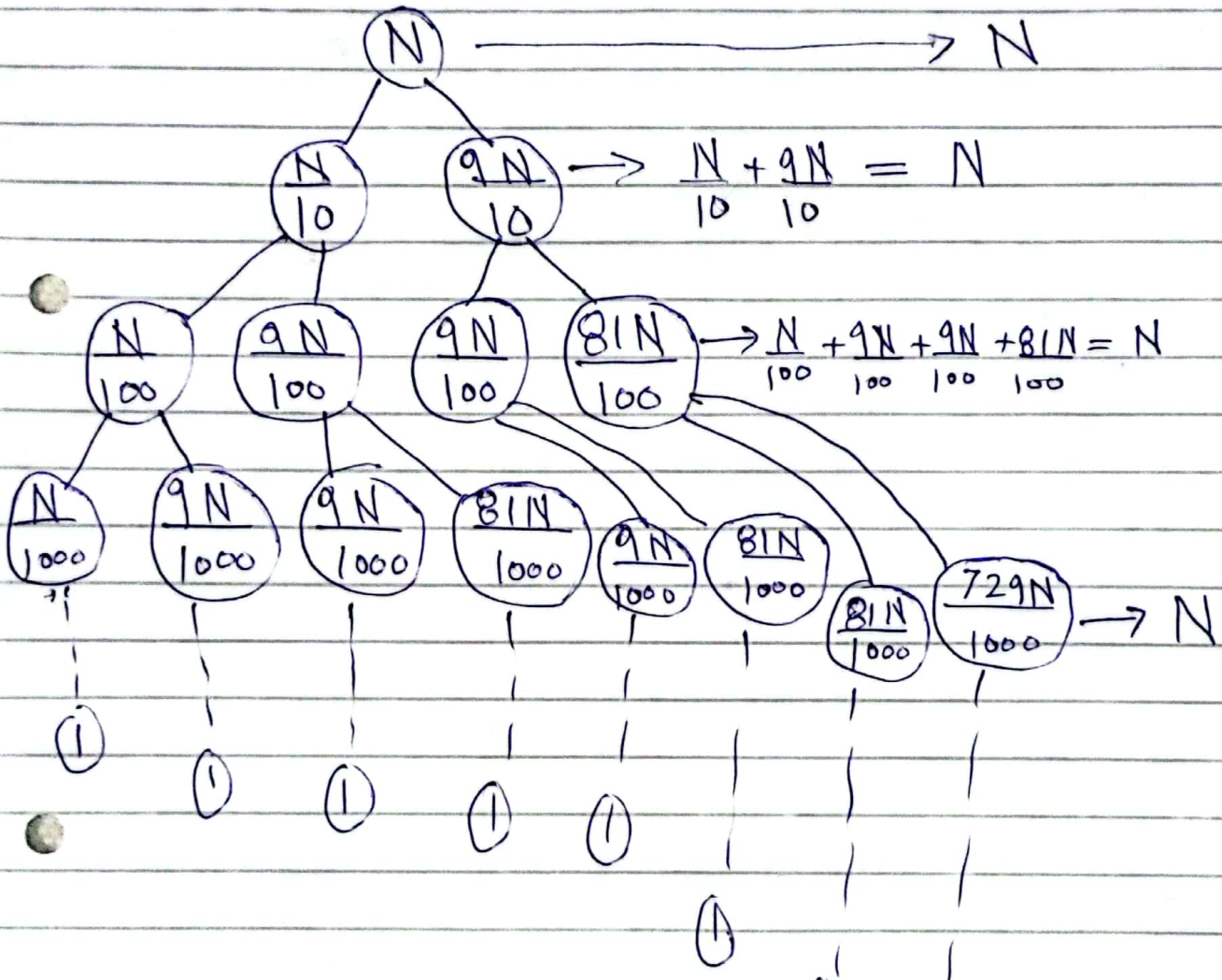
$$= \frac{8}{3}N^2$$

$$T(N) = \frac{8}{3}N^2$$

$$\Rightarrow O(N^2)$$

22L-6552 4H

Q2) c) p) $T(N) = T\left(\frac{N}{10}\right) + T\left(\frac{9N}{10}\right) + O(N)$



$T(N)$ = Cost at each level × Total Levels

$= N \times \text{Log}_{\frac{10}{9}}(N)$

$= N\text{Log}_{\frac{10}{9}}(N)$

$\Rightarrow O(N\text{Log}N)$

$1 \rightarrow N$ approximately.

$\frac{N}{\frac{10}{9}^K} = 1$ | $\text{Log}_{\frac{10}{9}} N = \text{Log}\left(\frac{10}{9}\right)^K$

$N = \frac{10}{9}^K$ | $\Rightarrow K = \text{Log}_{\frac{10}{9}}(N)$

Q3) a) $T(N) = \frac{1}{8}N^3 - 5N^2$

$$c_1 \times N^3 <= \frac{1}{8}N^3 - 5N^2 <= c_2 \times N^3$$

For upper bound:

$c_2 = 2 \Rightarrow \frac{1}{8}N^3 - 5N^2 <= 2N^3$

$n = 1 \Rightarrow \frac{-39}{8} <= 2 \rightarrow$ true

$n = 2 \Rightarrow -19 <= 16 \rightarrow$ true

$\Rightarrow n_0 = 1$

So $c_2 = 2$, $n_0 = 1$ and $g(N) = N^3$

For lower bound:

$c_1 = \frac{1}{100} \Rightarrow \frac{1}{8}N^3 - 5N^2 >= \frac{N^3}{100}$

$n = 1 \Rightarrow \frac{-39}{8} >= \frac{1}{100} \rightarrow$ false

$n = 2 \Rightarrow -19 >= \frac{2}{25} \rightarrow$ false

$n = 43 \Rightarrow 693.375 >= 795.07 \rightarrow$ false

$n = 44 \Rightarrow 968 >= 851.84 \rightarrow$ true

$n = 45 \Rightarrow 1265.625 >= 911.25 \rightarrow$ true

22L-6552 4H

So $c_1 = \frac{1}{100}$, $n_0 = 44$ and $g(N) = N^3$

Q3) b) $T(N) = 5\sqrt{N} + 3N^2 \log N + \frac{N}{(\log N)^2}$

$g(N) = N^2 \log N$

$c_1 \times N^2 \log(N) <= 5\sqrt{N} + 3N^2 \log N + \frac{N}{(\log N)^2} <= c_2 \times N^2 \log N$

For upper bound:

$c_2 = 10 \Rightarrow 5\sqrt{N} + 3N^2 \log N + \frac{N}{(\log N)^2} <= 10N^2 \log N$

$n = 1 \Rightarrow$ left side undefined

$n = 2 \Rightarrow \quad 32.75 <= 12.04 \rightarrow$ false

$n = 3 \Rightarrow \quad 34.72 <= 42.94 \rightarrow$ true

$n = 4 \Rightarrow \quad 49.93 <= 96.33 \rightarrow$ true

So $c_2 = 10$, $n_0 = 3$, $g(N) = N^2 \log(N)$

For lower bound :

$$C_1 = \frac{5}{2} \Rightarrow 5\sqrt{N} + 3N\text{Log}N + \frac{N}{(\text{Log}N)^2} >= \frac{5}{2} N^2 \text{Log}(N)$$

$n = 1 \Rightarrow$ left side undefined

~~$n = 2$ $32.68 <= 12.04 \rightarrow$~~

$n = 2$ $32.75 >= 3.01 \rightarrow$ ~~false~~ true

$n = 3$ $34.72 >= 10.74 \rightarrow$ true

So $C_1 = \frac{5}{2}$, $n_0 = 2$, $g(N) = N^2 \text{Log}(N)$

22L-6552 4H

Q4)

```
Merge_Sort(Arr[], left, right)
{
    if( left < right)
    {
        M = (left + right)/2
        Merge_Sort(Arr, left, M)
        Merge_Sort(Arr, M+1, right)
        Merge(Arr, left, M, right)
    }
}
```

```
Merge(arr[], start, mid, end)
{
    i = start  // pointer to left sub-array
    j = mid + 1  // pointer to right sub-array

    if(arr[mid] <= arr[j])  // case when element
    {                          at position m and
        return                 m+1 is already sorted
    }

    while(i <= mid && j <= end)
    {
        if(arr[i] <= arr[j])
        {
            i++  // keep moving ahead if element
        }            in left sub-array is smaller than
                     the element in the right
                       sub-array
        else
        {
            value = arr[j]  // save a value in the
            index = j           right sub-array

            while(index != i)  // move elements to the
            {                     right by one
                arr[index] = arr[index-1]    position
                index--
            }
            arr[i] = value  // put the saved value
                               at the start
```

22L-6552 4H

```
        i++  // move to next position in left sub-array
        mid++ // left sub-array will grow by one unit
        j++  // move to next position in right sub-array
    } // end of else portion

} // end of outer while loop

} // end of Merge function
```

Recurrence for the Merge Sort Algorithm

$$T(N) = T(N/2) + T(N/2) + \underbrace{O(N^2)}_{\text{Due to nested loops in the merge function}}$$

$$T(N) = 2T\left(\frac{N}{2}\right) + O(N^2)$$