★ Recursion Tree Method

So $T(n) = O(n.\log_2 n)$

Space Complex → $O(n)$

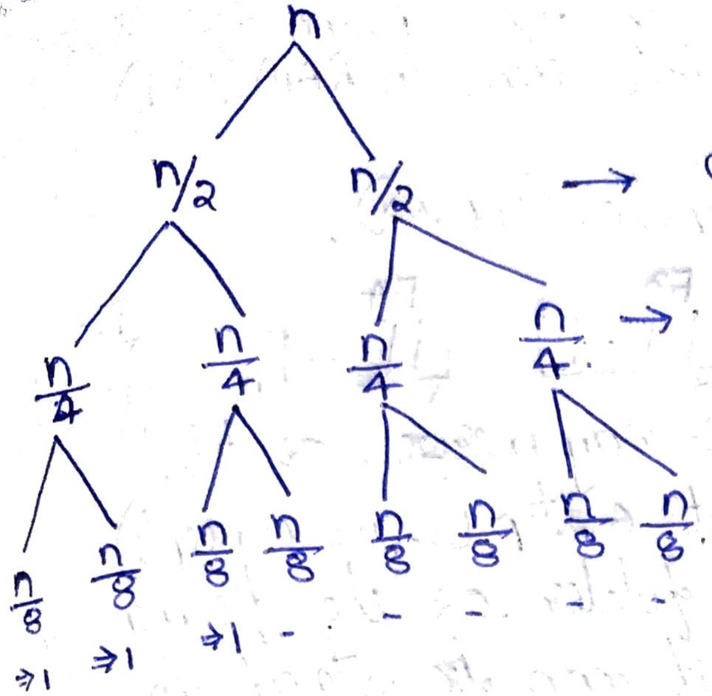
Merge_sort

Time → Worst → $O(n^2)$

Time → Best → $O(n)$.

Space → $O(1)$

Insertion sort



$\rightarrow C.(n)$

$\rightarrow C.(n/2) + C.(n/2) = C.(n)$

$\rightarrow C.\left(\frac{n}{4}\right) + C\left(\frac{n}{4}\right) + C\left(\frac{n}{4}\right) + C\left(\frac{n}{4}\right) = C(n)$

If n=8   Total 3 C(n)

↓

Height of tree

Ex if n = 16 ⇒ Height of tree ⇒ 4 levels.

n = 32 ⇒ Height of tree = 5 levels.

n = 8 ⇒ Height of tree = 3 levels.

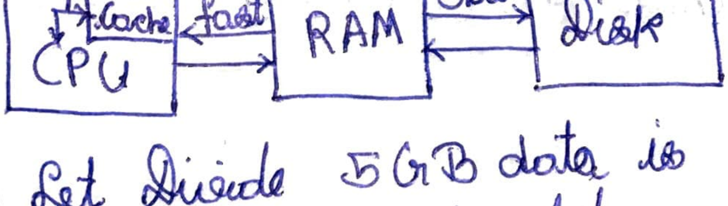
So by observing above

$$\boxed{\begin{array}{c} \log_2 n = x \\ \hline 2^x = n \end{array}}$$

★ External Merge - Sort.

1) Let we assume we have 5GB Data to be sorted

So given 1GB RAM.



Step

1) So Let Divide 5GB data is 5 parts each containing 1GB data file seperately.

So now we can sort 1GB data in 1GB RAM.

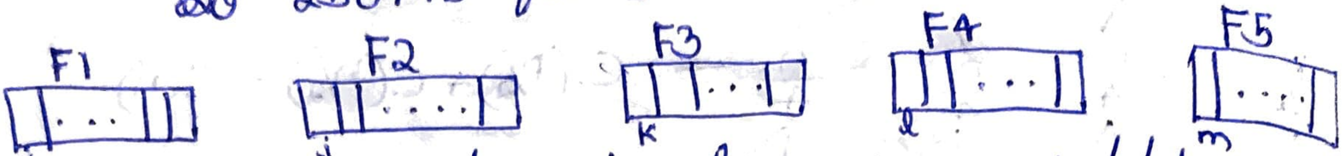
5 GB Data
- 1GB → Folder 1
- 1GB → Folder 2
- 1GB → Folder 3
- 1GB → Folder 4
- 1GB → Folder 5

Can be sorted by any Insertion or Merge sort

2) Step. Now take 150MB of data from the sorted 1 GB data each. and place them in RAM (1GB)

So 150 × 5 = 750MB.

So 250MB free space. (approx).



So there we perform K-way merge - as told in Pseudo code at last 6 step (same concept) by comparing in all folder. as the free memory of 250MB is filled it can be stored in the Disk to free the memory in RAM for the next process this process continues until the whole 5GB data is sorted

★ But the 250MB sorted will be stored in Disk in multiple folders.

So the Output files are in a sorted form.

So finally sorted 5GB in (1GB) RAM by using External Merge or (K Way Merge).

So External ~~Mer~~ Sort →. Sorting data when all of the data doesn't fit into RAM.

Merge Sort was invented in 1948 by. John von Neuman

Q) How many Comparisons are made in the while loop?

```
int j, n;                n>0.
  j = 1;
while (j <= n)
    j = j * 2;
```

so let $n = 5$.

so ⇒ $j = 1, 2, 4, 8$ ⇒ 4 ✓.

a) $\lceil \log_2 n \rceil + 2 \Rightarrow 3 + 2 = \underline{5}$ ✗

b) n ✗

c) $\lceil \log_2 n \rceil \Rightarrow 3$ ✗.

✓ d) $\lfloor \log_2 n \rfloor + 2 \Rightarrow 2 + 2 = \underline{4}$.

Q)
```
Sum = 0;
for (i=1; i<=n; i++).
{ for (j=1; j<n; j=j*2).
      { Sum = Sum + j;  → # times this line is executed?
      }.
}.
```

⇒ so $i = 1, 2, 3, 4, \ldots\ldots n$.

$j =$ let take $n = 5$.

$j = \overset{2^0}{1}, \overset{2^1}{2}, \overset{2^2}{4} \Rightarrow 3$ times.

so. $1 + \lfloor \log_2 n \rfloor = K$.

1 is constant so $\log_2 n$

⇒ so $O(n \log_2 n)$.

Q). Assume that a mergesort algorithm in the worst case takes 30 seconds for an input of size 64. Which of the following most closely approximates the maximum input size of a problem that can be solved in 6 minutes?

A) 256

B) 512.

C) 1024.

D) 2018.

$\Rightarrow$ so $n = 64$. $\rightarrow$ 30 sec.

$c \cdot n \log_2 n \rightarrow 30$ if $n = 64$.

$$\boxed{c = \frac{30}{64 \times 6}}$$

so to find size for 6 min.

so $6 \times 60 \Rightarrow 360$ sec.

so compare all the above option such that $\boxed{m \log_2 m = 4608}$

$c \times m \log_2 m = 360$.

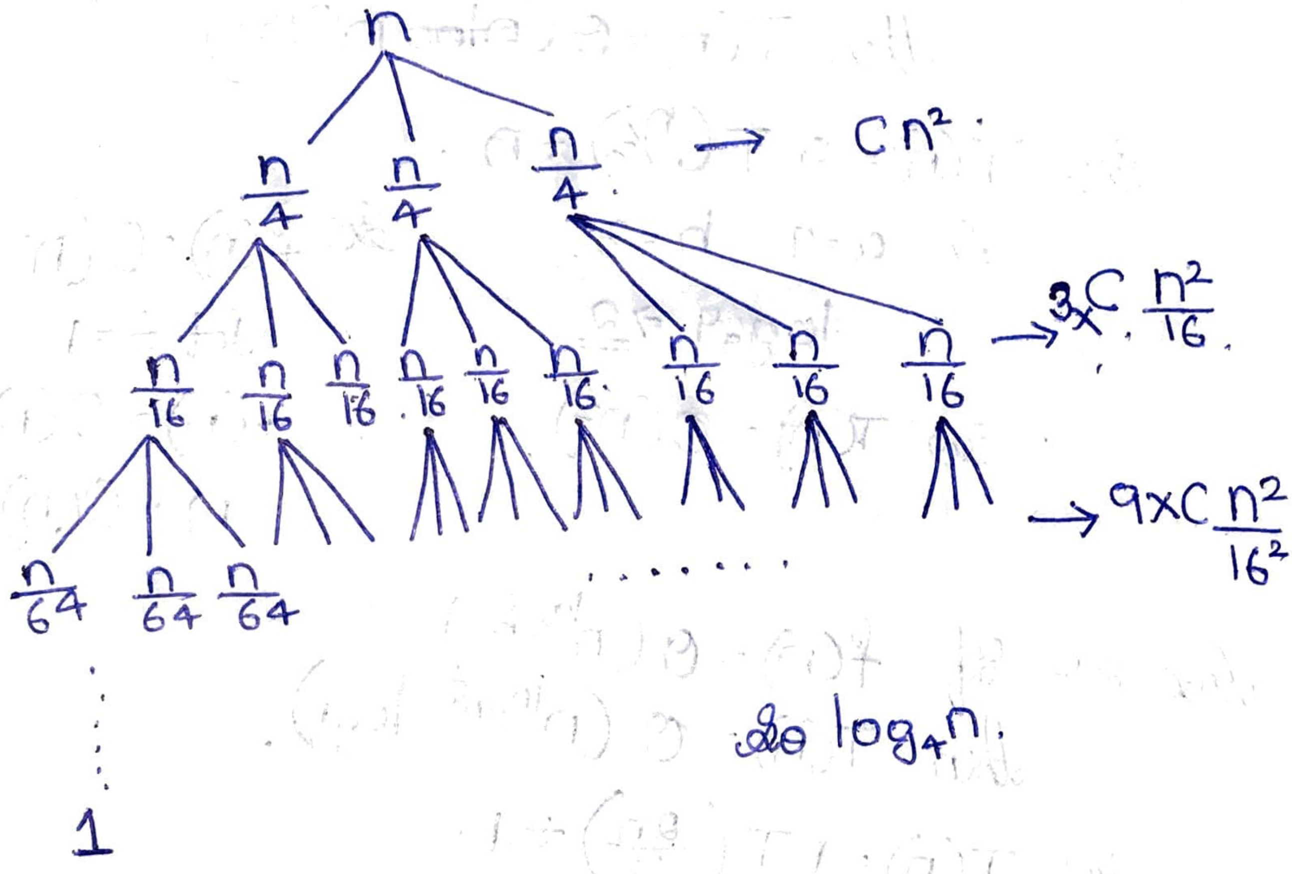
$m \log_2 m = \frac{\cancel{360}^{12} \times 6 \times 64}{\cancel{30}}$

$m \log_2 m = 4608$

★ Recursion Tree method

so Ex:-

$T(n) = 3T\left(\left\lfloor \frac{n}{4} \right\rfloor\right) + cn^2$



so Ex:- Lets take $n = 64$.

$$64 \downarrow 16 \downarrow 4 \downarrow 1$$

so $cn^2 + \frac{3}{16}cn^2 + \frac{9}{16^2}cn^2 + \left(\frac{3}{16}\right)^3 cn^2 + \left(\frac{3}{16}\right)^4 cn^2 + \dots$

$$= cn^2 \underbrace{\left[1 + \frac{3}{16} + \left(\frac{3}{16}\right)^2 + \left(\frac{3}{16}\right)^3 + \dots\right]}_{\text{geometric progression}}$$

$$a + ar + ar^2 + ar^3 + \dots = \sum_{k=0}^{\infty} ar^k = \frac{a}{1-r}, \ |r| < 1$$

$$1 + r + r^2 + r^3 + \dots = \frac{1}{1-r}$$

$$\Rightarrow cn^2 \left\{\frac{1}{1 - \left(\frac{3}{16}\right)}\right\} \Rightarrow cn^2 \left\{\frac{16}{13}\right\}$$ so $T(n) = O(n^2)$