Date : ..../......../........

Page : ....

1

We can use binary search algorithm to search the entire array and find the position
To do so, we need to find the mid element compare it with our element if it is equal we have the index else we would either search the left or right part of array

1 There will be a total n cases to be considered as our key can be present in one of the n positions.

3 ~~Since~~ there can be only $\log_2 n$ maximum comparisions.
At each ith step ~~for each position i~~ the probability that we have match will be

$i2^{i-1}$ → no of indices which will have i comparisions.

i ← no of comparisions

n → total no of indices ~~comparisions~~ possible.

so $\sum_{i=1}^{\log_2 n} \frac{i2^{i-1}}{2^m-1}$

$$S_n = \frac{1}{2^m-1} + \frac{2.2}{2^m-1} + \frac{3.2^2}{2^m-1} + \ldots$$

$$2S_n = \frac{2}{2^m-1} + \frac{2\ 2^2}{2^m-1} + \ldots$$

$$-S_n = \frac{1}{2^m-1} + \frac{2}{2^m-1} + \frac{2^2}{2^m-1} + \ldots - \frac{2.\log_2 n^{\log_2 n}}{2^m-1}$$

$$-S_n = \frac{1}{2^m-1}\left(\frac{1(1-2)^{\log_2 n+1}}{1-2}\right)$$

iNKWORM

Date : ..../......../........

Page : ....

$$S_n = -\left[\left(\frac{1}{2^m-1}\right)\left(\frac{\cancel{2}^{\log_2 n+1}-1}{2-1}\right)\right] - \frac{2^{\log_2 n}\log_2 n}{2^m-1}$$

$$S_n = \frac{2^{\log_2 n}\log_2 n}{2^m-1} + \frac{(1-2^{\log_2 n+1})}{2^m-1}$$

$$= \frac{n\log_2 n}{n} + \frac{(1-2\cdot 2^{\log_2 n})}{n}$$

$$= \log_2 n + \frac{1}{n} - \left(\frac{2\cdot n}{n}\right).$$

$$= O(\log_2 n)$$

$$= O(m)$$