PW – Java & DSA

Date: 16/12/2022

Time & Space Complexity and Big O Notation | Lecture 25.

1. Time Complexity: No of operations as a function of Input.

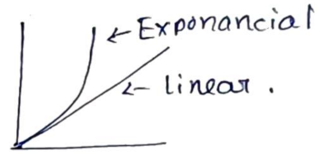
Asymptopic Analysis. (also known as)

- Worst case ~~case~~ time complexity → Big O, O(n) (In genaral)
- Best case time complexity → Big omega, $\Omega(1)$
- ~~Avarage~~ Avarage case time complexity → Big theta $\Theta(n)$

- ~~[illegible]~~ In 1 second → $10^8 - 10^9$ operations can be perfomed on an avarage.

1. O(1) → constant time complexity.

(*)



- TLE means time limit Exceded.

- Time complexity for traversing an array of length N is → Big O(n).

- Time complexity when traversing 2 indivisual array of length M & N respectively → O(m+n).

- Nested loops time complexity → $O(n^2)$. [where $i<n$ & $j<n$]

  $O(n \times \sqrt{n})$ [where $i<n$ & $j<\sqrt{n}$]

Date: 18/12/2022

- Time complexity for traversing the Array and multiply the increment pointer by 2. ⇒ O(log n)

$$\boxed{\begin{aligned} &\log_a x^n = n\log_a x \\ &\log_b b = 1 \\ &\log_a b = \frac{\log_c b}{\log_c a} \end{aligned}}$$

```
int val = 0;
for (int i = 1; i <= N; i *= k) {
  val++;
}
```

multiply the increment pointer by k.

Time complexity = $O(\log_K N)$

Space Complexity: extra memory / space used by an algorithm with respect to the input size. $\propto$ input size.

- Input & Output not considered as space. (clarify that)
- Asymptopic Analysis.
- Space complexity for an array of length N. $\rightarrow O(n)$
- Space complexity for a 2-d matrix of N rows and M columns $\rightarrow O(m \times n)$