Intro

Tuesday, 18 June 2024 12:37 AM

– Time & space complexity is used to measure the efficiency of a programme

→ Time :

Amount of time taken by an algorithm by a function of its input size.

→ Space :

Amount of space taken by an algorithm by a function of its input size

How to represent

→ By using asymptotic notations

1. Big-O Notation $O(n)$: Worst case complexity
2. Omega notation $\Omega(n)$: Best case complexity
3. Theta notation $\Theta(n)$: Average case complexity.

Types of O(notation)

→ Constant $O(1)$: Remains constant irrespective of input size.

→ linear $O(n)$: simple for loop till n

→ Quadratic $O(n^2)$: 2 nested for loops

→ logarithmic $O(n \log n)$: input size reduces by half every iteration

JS Objects Big-O

Insert – $O(1)$

Remove – $O(1)$

Access – $O(1)$

Search for a value – $O(n)$

Object.keys
Object.values
Object.entries
} → $O(n)$

Arrays – Big-O

Insert/remove at end – $O(1)$

Insert/remove at beginning – $O(n)$: indexes have to be reset.

Access – $O(1)$

Search – $O(n)$

push/pop – $O(1)$

shift/unshift/concat/slice/splice – $O(n)$

foreach/map/filter/reduce – $O(n)$

sort – $O(n \log n)$

When we execute code in coding platforms:

it takes roughly 1s to run 10^8 operations

1s — 10^8 ops
2s — 2×10^8 ops
3s — 3×10^8 ops