**Complexity(notion of order of complexity)/Complexity of algorithms**:--2 marks or 5 marks

Def--Algorithmic complexity is concerned about how fast or slow particular algorithm performs and how much space a particular algorithm takes.

Time complexity: It represents the amount of time required by the algorithm to run to completion.

Space complexity: It represents the amount of space required by the application and its life cycle.—not an issue nowadays.

Usually time required by an algorithm falls under 3 category.

Best Case: minimum time required for program execution,

Average Case: Average time required for “ “

Worst Case: maximum time required for program execution.

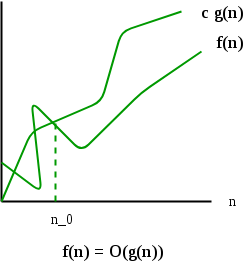
**Asymptotic notation ( Notion of order of Complexity)**

1. Big-O Notation (Ο) –worst case

2. Omega Notation (Ω)-best case

3. Theta Notation (θ)-average case

**Big O notation** specifically describes worst case scenario. It represents the upper bound running time complexity of an algorithm.

**This means that function can do better but not worse.**

**Omega notation** specifically describes best case scenario. It represents the lower bound running time complexity of an algorithm. This means that the function can never do better than the specified value but it may do worse.

BigOmega

**Theta notation** specifically describes average case scenario. It represents both upper bound and lower bound of an algorithm.thetanotation

f(n)= θ(g(n))

Reading:\_

**Complexity of Linear search: O(n)**

**Complexity of bubble sort : O(n2)**

**Complexity of selection sort: O(n2)**

**Complexity of Insertion sort: O(n2)**

**Complexity of Binary search: O(log(n))**