1.How will you measure the running time of an algorithm?

T(n) = Running time of an algorithm
Identify basic operation:

C(n)=basic operation count

Coperation = Time taken by basic operation for its execution

T(n)≅Coperation . C(n).

2. What are the steps involved in the analysis framework?

Steps:

i)Finding time and space efficiency(complexity)

ii)Measuring an input's size

iii)Measuring running time and Units for Measuring Running lime

iv)Orders of growth

v)worst,best,average case efficiencies

3. What is a basic operation?

The most time-consuming operation in the algorithm is called as basic operation. For example, most sorting algorithms work by comparing elements (keys) of a list being sorted with each other; for such algorithms, the basic operation is a key comparison.

4.Arrange the following functions based on its order of decay n!, nlogn,15logn, n^3 , 100n^2 +n, 2^n

n!, 2^n, n^3, 100n^2+n, nlogn, 15logn

5.Arrange the following functions based on its order of growth n!, nlogn, 15logn, n^3 , 100n^2 +n, 2^n

15logn, nlogn, 100n^2+n, n^3, 2^n, n!

6. Give formal and informal definition of Big Theta.

Formal:

*t(n) ∈ θ(g(n)) iff t(n) is bounded above and below by some constant multiple of g(n).

c_2.g(n)≤t(n)≤ c_1.g(n); $n_0 = max(n_1, n_2)$

Informal:

*Set of all functions whose order of growth is same as g(n)

(eg): $10n^2$-5∈ $\theta(n^2)$

7. Give formal and informal definition of Big Oh.

Formal:

t(n) ∈ $O(g(n))$ iff order of growth of t(n) is less than or equal to some constant multiple of g(n).

(i.e): t(n) ≤ c.g(n) for some position constant c, n≥ n_0

Informal:

Set of all functions whose order of growth is less than or same as order of growth of g(n).

Eg: $10n^2$+n ∈ O(n^2)

8. Give formal and informal definition of Big Omega.

Formal: t(n) ∈ Ω (g(n)) iff order of t(n) is greater than or equal to and bounded below some constant multiple of g(n).

(i.e): t(n) ≥ c.g(n) for some positive constant c and n≥ n_0

Informal: set of all function whose order of growth is greater or same as g(n).

Eg: $10n^2$+n ∈ $\Omega(n^2)$

9. Define best, average and worst case efficiency.

Best case: The best-case efficiency of an algorithm is its efficiency for the best-case input

of size n, which is an input (or inputs) of size n for which the algorithm runs the

fastest among all possible inputs of that size

Eg: linear search

Average case: "average" over inputs of size n Number of times the basic operation will be executed on typical input.In average case, some assumptions about possible inputs of

size n.

Eg: Linear search

Worst case: maximum over inputs of size n. In worst case, the case that causes a maximum number of operations to be executed.

Eg: Linear search with time complexity O(n).

9. Identify true or false $5n^2 - 6n \in O(n^2)$

True

10. Identify true or false n(n^2+1) ∈ Ω(n^2)

True

11.Find the time complexity of sum of n given numbers.

/////DON'T KNOW

1.Algorithm for linear search and its best,avg,worst cases:

//Algorithm linear search(a[0..n-1],key)

//Input: list of n numbers and key to be search

//Output: Returns position of key in a list if successful search, otherwise return -1.

```
for i=0 to n-1 do
   if a[i] == key
      return i
return -1
```

Best case:

key is present in 1^{st} position

Number of comparison = 1(best case)

T(n) $\in \Omega$ (1)

Average case:

Probability of successful search = p

Probability of unsuccessful search = 1-p

T_{avg}(n)=$\frac{(1.p + 2.p + \ldots.+n.p)}{n}$ + n(1-p)

$= \frac{p(1+2+3\ldots n)}{n}$+n(1-p)

$= p\,\frac{(n+1)}{2}$+n(1-p)

Case i): For successful search: p=1	case ii):for unsuccessful search: p=0
T(n) $= \frac{n+1}{2}+0$ $= \frac{n+1}{2}$	T(n) $= 0*\frac{n+1}{2}$+n(1-0) = n

Worst case:

Algorithm runs slowest among all possible inputs of size n.

T(n) $\in$ O(n)

2.Discuss about asymptotic notations in detail.

BIG OH(O):

Formal:

t(n) ∈ $O(g(n))$ iff order of growth of t(n) is less than or equal to some constant multiple of g(n).

(i.e): t(n) ≤ c.g(n) for some position constant c, n≥ n_0

Informal:

Set of all functions whose order of growth is less than or same as order of growth of g(n).

Eg: $10n^2$+n ∈ O(n^2)

BIG OMEGA(Ω):

Formal: t(n) ∈ Ω (g(n)) iff order of t(n) is greater than or equal to and bounded below some constant multiple of g(n).

(i.e): t(n) ≥ c.g(n) for some positive constant c and n≥ n_0

Informal: set of all function whose order of growth is greater or same as g(n).

Eg: $10n^2$+n ∈ $\Omega(n^2)$

BIG THETA(θ):
Formal:

*t(n) ∈ θ(g(n)) iff t(n) is bounded above and below by some constant multiple of g(n).

c_2.g(n)≤t(n)≤ c_1.g(n); $n_0 = max(n_1, n_2)$

Informal:

*Set of all functions whose order of growth is same as g(n)

(eg): $10n^2$-5∈ $\theta(n^2)$

3. Prove the following assertions:

i) n^2 +5n-10 ∈ O(n^2)

ii)n^3 +n^2 ∈ ϴ(n^3)

iii) 100n^2 -n-10 ∈ Ω(n^2)

//SOLVE IT YOURSELF