Intermediate DSA
Introduction To Problem Solving

14:08

Agenda :

1. Count the Factors
2. Optimisation for counting the Factors
3. Definition of AP & GP
4. How to compare two Algorithms.

1. Count Of Factors

Brute Force Approach

Consider a number n= 12
Min factor = 1
Max Factor = 12, rest all factor will come in between min and max. So we can start iterating from min till max and whichever is able to perfectly divide the number will be our factors.

Pseudo :

```
Int count=0;
For(int i=1; i<n; i++){
        If(n%i == 0){
                Count++;
        }
}
Return count;
```

Note :

- Here total number of iterations made for calculating number of factors of number n is n-iterations.
- **Servers have the capability of running ~10^8 Iterations in 1 sec.**

N	Iterations	Execution Time
10^8	10^8 iterations	1 sec
10^9	10^9 iterations	10 sec
10^18	10^18 iterations	317 years

- So if number is as big as 10^8, it will take 317 years to get the count of factors, which is illogical. **So we need to optimize the solution.**

2. Optimization for Counting of Factors

Optimization :-

if $i*j = N$; then i & j are the factors

ie i, N/i are the factors.

For example, N = 24
Comment

i	N / i
1	24
2	12
3	8
4	6
6	4
8	3
12	2
24	1

→ we can see if we just iterate till 6. i.e sqrt(24)
we can get all the factors.

We need to only iterate till - sqrt(n)

Pseudocode :

```
function countfactors (N):
    fac = 0
    for i = 1 till sqrt(N):
        if N % i == 0:
            if i == N / i:
                fac = fac + 1
            else:
                fac = fac + 2
    return fac
```

N	Iterations	Execution Time
10^18	10^9 iterations	10 secs

Note : Now a lot of questions can be solved using this logic like prime numbers, count of factors etc.

3. GP (geometric progression)

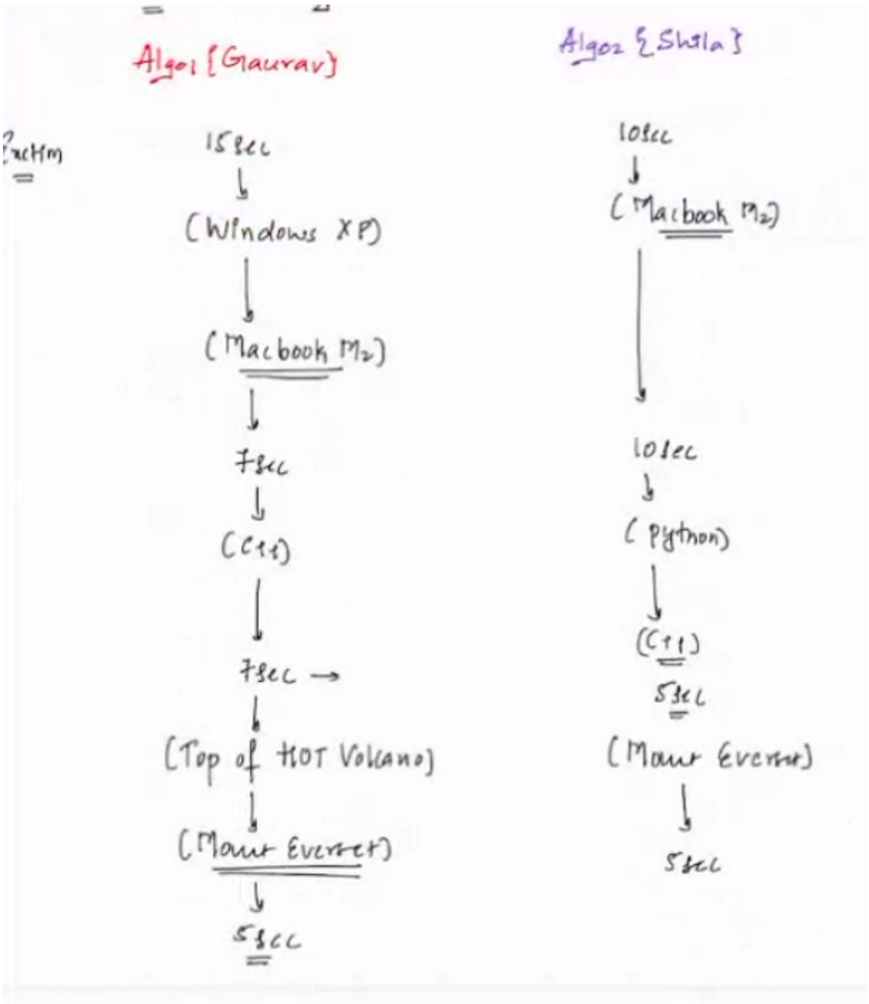
Generic Notation:

a, a * r, a * r^2, …

$\frac{a*(r^n-1)}{(r-1)}$ for r!=1

Sum of first N terms of GP:

4. How to Compare Two Algorithms



Conclusion

We can't evaluate algorithm's performance using execution time as it depends on a lot of factors like operating system, place of execution, language, etc.

But we can **calculate no of iterations,** as it is same on all the platforms.