Yash Raj → 0%

2017 : Birla Pilani hyd. CS
2017 : Directi / Media.net ⇒ 3.5 years
2021 : Resigned
2022 : → Reelhub,

1 - 1.5 years teaching DSA.

⇒ 2014 - 2nd year. [ACM ICPC ↓]
2015 - 20
2016 - 8

⇒ Directi ⇒ Hiring Committee
↓
2 years (50+)

⇒ 1.5 years ⇒ 5-6 Batches

Q1 Given a number n, find how many factors does n have.

Approach 1 :

```
c = 0
for (int i = 1 ; i <= n ; i++) {
    if (n % i == 0) {
        c++
    }
}
```

Iterations : n $\quad$ $n = 10^9$

Total instructions : $5n$ $\quad$ Processor = 10^9 instruction per second

5×10^9

$\simeq$ 5 sec

$n = 10^{18}$

Total instruction $= 5 \times 10^{18}$

Time taken $= \frac{5 \times 10^{18}}{10^9} \Rightarrow 5 \times 10^9$ seconds

$\Rightarrow$ 158 years

Approach 2

$n = 24$

i	n/i
1	24
2	12
3	8
4	6
6 >	4
8 >	3
12 >	2
24 >	1

$n = 100$

i	n/i
1	100
2	50
4	25
5	20
10	10
20 >	5
25 >	4
50 >	2
100 >	1

$$i \leq \frac{n}{i} \Rightarrow i^2 \leq n \Rightarrow i \leq \sqrt{n}$$

```
c = 0          int size = sqrt(n)

for (int i = 1 ; i <= size ; i++) {
    if (n % i == 0) {
        if (i != n/i)
            c = c + 2
        else
            c = c + 1
    }
}
```

Iteration : $\sqrt{n}$

Instructions : $5 \times \sqrt{n}$

$n = 10^{18} \Rightarrow 5 \times \sqrt{10^{18}} = 5 \times 10^9$

How much time will the proc take $\Rightarrow \frac{5 \times 10^9}{10^9} \Rightarrow 5 \text{ sec}$

Problem given to Harshit & Mayank
Everything is same in both problem.

	Harshit Algo1	Mayank Algo2
Time taken	20 sec [Windows XP]	15 sec [Macbook pro m2 chip]
Time taken	12 sec [C++]	15 sec [Python]
Time taken	12 sec [Volcano]	10 sec [Antartica]

Time taken can never be a metric to compare algos.

What can be the correct metric : No of iterations / instruction.

Q1
```
for (int i=0; i<=100; i++) {
    .....
}
```

$O(1)$

$\# [a, b] = b - a + 1$

$[0, 100] = 100 - 0 + 1 \Rightarrow 101$

Q2
```
for (int i=1; i<=N; i++) {
    S=S+i
}
```

$O(n)$

$[1, N] = N - 1 + 1 = N$

Q3
```
for (int i=1; i<=N; i=i+2) {
    ....
}
```

$n \Rightarrow 8$

$i = 1, 3, 5, 7 = 4$

$n \Rightarrow 9$

$i = 1, 3, 5, 7, 9$

$\left(\frac{n}{2}\right) / \left(\frac{n+1}{2}\right)$ → Integer Division ← $\left(\frac{n+1}{2}\right) \Rightarrow O(N)$

Q4 `for (i = 1; i*i <= N; i++) {`

`...`

`}`

⇒ sqrt(N)

$\sqrt{N}$ ⇒ $O(\sqrt{n})$

Q5

```
i = N
while (i > 1) {
    i = i/2;
}
```

⇒ $N, \frac{N}{2^1}, \frac{N}{2^2}, \frac{N}{2^3}, \ldots \frac{N}{2^k} \ldots 1$

$\frac{N}{2^k} \Rightarrow 1$

$N = 2^k$

$\log_2 N = k$

$O(\log N)$

$\log_2 N$ is the number we need to divide N by 2 so that it reaches 1.

Q
```
for (int i=0 ; i <= n ; i=i*2) {
}
```

$i = 0, 0, 0, 0 \ldots$

⇒ Infinite iterations

Q
```
for (int i=1 ; i <= n ; i=i*2) {
}
```

$i = 1, 2, 4, 8 \ldots N$

$N, \frac{N}{2^1}, \frac{N}{2^2}, \frac{N}{2^3}, \ldots \frac{N}{2^k} \ldots 1$

⇒ $\log_2 N = O(\log N)$

$\log_2 N$ is the number of time we need to multiply 1 by 2 so that it reaches N.

Q

```
for (int i = 1 ; i <= 10 ; i++) {
    for (int j = 1 ; j <= n ; j++) {

    }
}
```

i	j	no of iterations in second loop
1	[1, n]	n
2	[1, n]	n
3	[1, n]	n
⋮		
10	[1, n]	n

10 → 10n

Total iterations ⇒ 10n

⇒ O(n)

Q

```
for (int i = 1 ; i <= n ; i++) {
    for (int j = 1 ; j <= n ; i++) {

    }
}
```

iterations = n^2

⇒ $O(n^2)$

Q

```
for (int i = 1 ; i ≤ n ; i++) {
    for (int j = 1 ; j ≤ n ; j = j*2) {
        ==
    }
}
```

i	j	total
1	[1, n]	$\log_2 n$
2	[1, n]	$\log_2 n$
⋮		
n	[1, n]	$\log_2 n$

$O(n \log n)$ ← $n \log n$

Q How many elements in range [3, 10]

$[a, b] = b - a + 1$

$[3, 10] = 10 - 3 + 1 \Rightarrow 8$

[3, 4, 5, 6, 7, 8, 9, 10]

10:32 pm

Akshat | Ravi

$\frac{n}{10}$ | $100 \log N$

$n \leq 3500$ — $n/10$ is better

$n > 3500$ — $100 \log N$ is better

IPL → hotstar $\xrightarrow{max}$ 25 million. $\simeq$ 2.5 crore.

Popular Youtube videos $\simeq$ 1 Billion

$\Rightarrow$ for larger numbers Ravi's algo is better.

Asymtotic Analysis

↳ Analysis of an algorithm for large values of n / input.

$n \to \infty$

Bigo | Omega | Theta

Big O

How to calculate Big O

1) Calculate number of iteration

2) Ignore lower order terms

3) Ignore Constants.

Q : iterations = $4n^2 + 3n + 2$

Ignore lower order $\Rightarrow$ $4n^2$.
Ignore Constants $\Rightarrow$ n^2

$$O(n^2)$$

Q iterations $\Rightarrow$ $n^2 + n\log n + 3n$

$\Rightarrow$ $O(n^2)$

I Why we are ignoring lower order terms.

Ex. iterations $= n^2 + 100n$

n	iterations	contribution of lower order
10	$100 + 1000$	90%
100	$10000 + 10000$	50%
10^5	$10^{10} + 10^7$	$10^{-3} \simeq 0.1\%$

II Why ignore Constants

Akshat	Ravi
$10^3 n$	n^2

$n = 1000$ Akshar $= 10^6$

Ravi $= 10^6$

$n > 1000$, Akshat's algo would be better

How to avoid TLEs. [Time limit Exceeded]

Every Online Judge $\simeq 10^9$ instruction per second.

Case 1: Per iteration $\simeq$ 10 instruction

Maximum iteration $\Rightarrow 10^8$

Case 2: Per iteration $\simeq$ 100 instruction

Maximum iterations $\Rightarrow 10^7$

$\Rightarrow$ $[10^7, 10^8]$

Constraints

Ex1 $1 \leq n \leq 10^6$

(i) $O(n^2) \simeq 10^{12}$

(ii) $O(n\sqrt{n}) \simeq 10^9$

(iii) $O(n \log n) \simeq 10^6 \times 20$

2×10^7

(iv) $O(n) \simeq 10^6$

Ex2 $n = 10^3$ (i) $O(n^3) \simeq 10^9$

(ii) $O(n^2) \simeq 10^6$

Ex3 $n = 15$ (i) $O(2^n) < 10^8$

Space Complexity

I
```
void func() {
    int x;      4 bytes
    long p;     8 bytes
}
```
⇒ 12 bytes

⇒ $O(1)$

II
```
void func(int n) {
    int x;
    long p;
    int arr[n];
}
```
⇒ $12 + 4n$

SC ⇒ $O(n)$

III

```
void func (int arr[n], int n) {
    int x = 4;
    dog p = 1;
}
```

3

Total Space = 12 bytes + (4n + 4)

SC ⇒ O(1) Input

While computing the space complexity, we will always ignore the input space.

This course is Language Agnostic