Time Complexity 1

Good Evening

Enjoy the Song

Starting at 9:05pm

Today's content

- → Count of factors
- → Comparison of Algorithms
- → No. of Iterations & Big O
- → Space complexity
- → TLE
- → Bitwise operators & properties

FAQ's

(a) Notes will be uploaded right after the session

(b) Assignments will be unlocked once the session ends.

(c) There is no deadline for assignments
↳ I would highly appreciate if you guys do it on time. (Try to avoid backlogs)

(d) During doubt session, attendance is not counted
↳ recorded : Yes
↳ Notes : Yes

(e) Language Independent pseudocodes

Important things

→ Join before 9:05 pm

→ Question → Please put every question in question bar.

→ Answers → Privately

→ Quizzes → No discussions in chat

01. Sum of first N natural numbers

$$S = 1 + 2 + 3 + \dots + (N-1) + N$$

$$S = N + (N-1) + (N-2) + \dots + 2 + 1$$

$$2S = N * (N+1)$$

$$S = \frac{N * (N+1)}{2}$$

02. Numbers in Range [3 8]

No. = 3 4 5 6 7 8

Ans = 6

$$[L \quad R] = R - L + 1$$

$$(L \quad R) = R - L + 1 - 2$$

$$= R - L - 1$$

03. No. of factors $N = 24 \longrightarrow 1, 2, 3, 4, 6, 8, 12, 24$

Ans = 8

Factor $\longrightarrow$ n % factor = 0

Count no. of factors

01. Given N, return no. of factors of N?

(a) N = 10 → {1, 2, 5, 10} Ans = 4

(b) N = 12 → {1, 2, 3, 4, 6, 12} Ans = 6

BF Idea → Iterate from 1 to N & check every no. if it is giving remainder 0 or not

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

Assumption → 10^8 iterations → 1 sec

$N = 10^9$

$$1 \text{ iteration} = \frac{1}{10^8} \text{ sec}$$

$$10^9 \text{ iterations} = \frac{1}{10^8} \times 10^9 \text{ sec} = 10 \text{ sec}$$

$N = 10^{18}$

10^{18} iterations = 317 years

Optimization

Optimised Idea

01. $i * j = N$

(i is a factor, $\frac{N}{i}$ is also a factor)

02. $a \leq 50$

max $a = 50$

Min $i = 1$

Max $i \leq \frac{N}{i}$

$i^2 \leq N$

max $i \leq \sqrt{N}$

N = 24

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

Part A

i < N/i

N = 36

i	N/i
(1)	36
2	18
3	12
4	9
(6)	6
9	4
12	3
18	2
36	1

i = 1

i = N/i

```
countfactors (int N)
    int c=0
    for (i=1; i*i<=N; i++){
        if (N % i ==0){
            if (i==N/i) {c++;}
            else {c=c+2};
        }
    }
    return c;
}
```

No. of iteration : $\sqrt{N}$

$N = 10^{18}$ → No. of iterations

$\sqrt{N} = \sqrt{10^{18}} = 10^9$

10^9 iterations = 10 sec

$N = 10^{18}$ → 317 years

→ 10 sec

* Comparison of Algo

Q Given $N = 10^4$ elements, sort them in increasing order.

arr[5] = {3 2 6 8 1} —sort→ {1 2 3 6 8}

→ 2 people have submitted their Algos

Note :- Both are executed on same input

Execution time

Vibhas

15 sec

{Window xp}

↓ Macbook

10 sec {python}

↓ C++

7 sec {Normal temp}

↓

7 sec

Anish

10 sec

{Macbook Pro}

↓

10 sec {C++}

↓

10 sec {Volcano}

↓ Normal temp

5 sec

Issues with execution time → It depends upon lot of external factor

```
void fun (int n)
{
    for (i=0; i<=n; i++){        → no. of iteration = n+1 iterations
        sum = sum+i;
    }
}
```

* Comparison of two algorithms should be done on the no. of iterations

* No. of iterations

```
void fun (int n){
    int i=n
    while (i>1){
        print (Hi);
        i = i/2
    }
}
```

Iterations	Value of i after each iteration
1	$i = n/2 = \frac{n}{2^1}$
2	$i = \frac{n}{4} = \frac{n}{2^2}$
3	$i = \frac{n}{8} = \frac{n}{2^3}$
4	$i = n/16 = n/2^4$
⋮	
k	$i = n/2^k$

$i = 1$, the iteration will stop

$$1 = \frac{n}{2^k}$$

$$n = 2^k$$

$$\log_2 n = \log_2 2^k$$

$$\log_2 n = k$$

$TC = O(\log_2 n)$

02.

```
void fun (int n)
{
    for (int i=1 ; i≤N ; i++){
        for (int j=1 ; j ≤N ; j++){
            print ("Hi");
        }
    }
}
```

No. of iterations = N^2

i	j	No. of iteration
1	[1 N]	N
2	[1 N]	N
3	[1 N]	N
⋮	⋮	⋮
N	[1 N]	N

Total iterations = N^2

$TC = O(N^2)$

```
void fun (int n)
{
    for (int i=0; i<N ; i ++){
        for (int j=0 ; j≤i  ; j++ ){
            print ("Hi");
        }
    }
}
```

i	j	No. of iteration
0	[0 0]	1
1	[0 1]	2
2	[0 2]	3
⋮		⋮
n-1	[0 n-1]	n

$\frac{n^2}{2} + \frac{n}{2}$

$\frac{n * (n+1)}{2}$

TC : $O(n^2)$

Big O

$f(x) = x^2 + 3x - 2$

$O(f(n))$ = Rate of growth of function w.r.t N

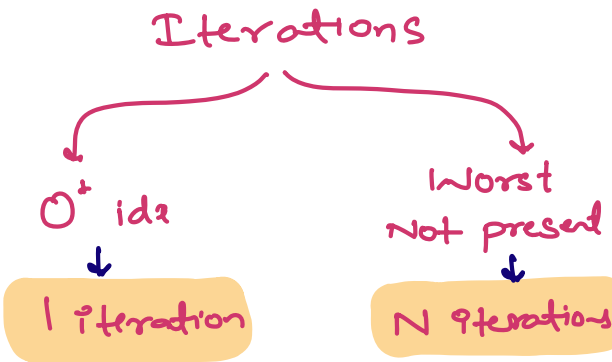
01 $f(N) = 3n^3 + 4n^2 + 2$ $\quad O(f(N)) = O(n^3)$

02 $f(N) = 3n^2 + 4\log N$ $\quad O(f(N)) = O(n^2)$

03. $f(N) = \sqrt{N} - 2\log N$ $\quad O(f(N)) = O(\sqrt{n})$

TC = O(# Iterations)

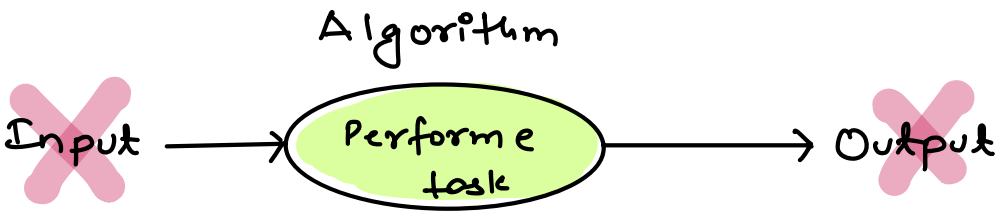
```
boolean search (int ar, int k)
{
    for( i=0; i<N; i++) {
        if (ar[i]==k) return true
    }
    return false
}
```



Big (O) → Always consider the worst case scenario.

10:02 pm → 10:10 pm

* Space complexity



int ⟶ 4B long ⟶ 8B

```
func (int n){                          // memory constant
    int x = n;          // 4B
    int y = x + x;      // 4B
    long z = x + y;     // 8B
}
```

Memory = 4B + 4B + 8B

= 16B

SC = O(1)

```
func (int n){
    int x = n;          // 4B
    int y = x + x;      // 4B
    long z = x + y;     // 8B
    int []ar = new int[n];   // 4N
}
```

Total space = 16B + 4N

≈ 4N

SC = O(n)

```
func (int n){
    int x = n;
    int y = x + x;
    long z = x + y;
    int []ar = new int[n];
    long [][] mat = new long[n][n];
}
```

SC : $O(N^2)$

TLE { Time Limit Exceeded }

Online Editor ⟶ Online Servers

Process speed = 1 GHz

$= 10^9$ ____ /sec

$\Rightarrow 10^9$ instructions / sec

Approximately, 1 iteration = 10 instructions

At max, 10^9 instructions

$= 10 * 10^8$ instructions

$= 10^8$ iterations

1 iteration = 100 instructions

Our code can atmax have 10^7 iterations

Code iteration = $10^7 - 10^8$ iterations

$O(N^2)$

$N = 10^5$ → 10^{10} iterations

$N = 10^3$ → 10^6 iteration

$N = 10^4$ → 10^8 iteration → Either it will work OR it will not work

* Bitwise operators

&	|	^	~	<<	>>
AND	OR	XOR	NOT	Left	Right

0 → unset bit / False

1 → set bit / True

→ Addition without carry

A	B	A & B	A | B	A ^ B	~A	~B
0	1	0	1	1	1	0
0	0	0	0	0	1	1
1	1	1	1	0	0	0
1	0	0	1	1	0	1

A & B = Both the bits are set

A | B = Only one bit has to be set

A ^ B = same same puppy shame

diff bits = 1

same bits = 0

A = 20 A = 0 1 0 1 0 0

B = 45 B = 1 0 1 1 0 1

Print (A&B) = 0 0 0 1 0 0 → 4

print (A|B) = 1 1 1 1 0 1

print (A^B) = 1 1 1 0 0 1

Properties of Bitwise Operators

01. A & 1 =

A=10 = 1010

A=11 = 1011

```
A = 1010
&
1 = 0001
-----------
0 = 0000
```

```
A = 1011
&
1 = 0001
-----------
1 = 0001
```

A & 1 → 0 { A last bit = 0 } → A is even

A & 1 → 1 { A last bit = 1 } → A is odd

02 A & 0 = 0

03. A & A = A

04. A | 0 = A

05 A ^ A = 0

06. A ^ 0 = A

```
A=10   1010
       1010
-------------
       0000
```

```
A  1010
^0 0000
--------
```

Q = All the numbers in array has duplicates except one ele. Find the unique ele.

arr = {1, 3, 1, 4, 3} Ans = 4

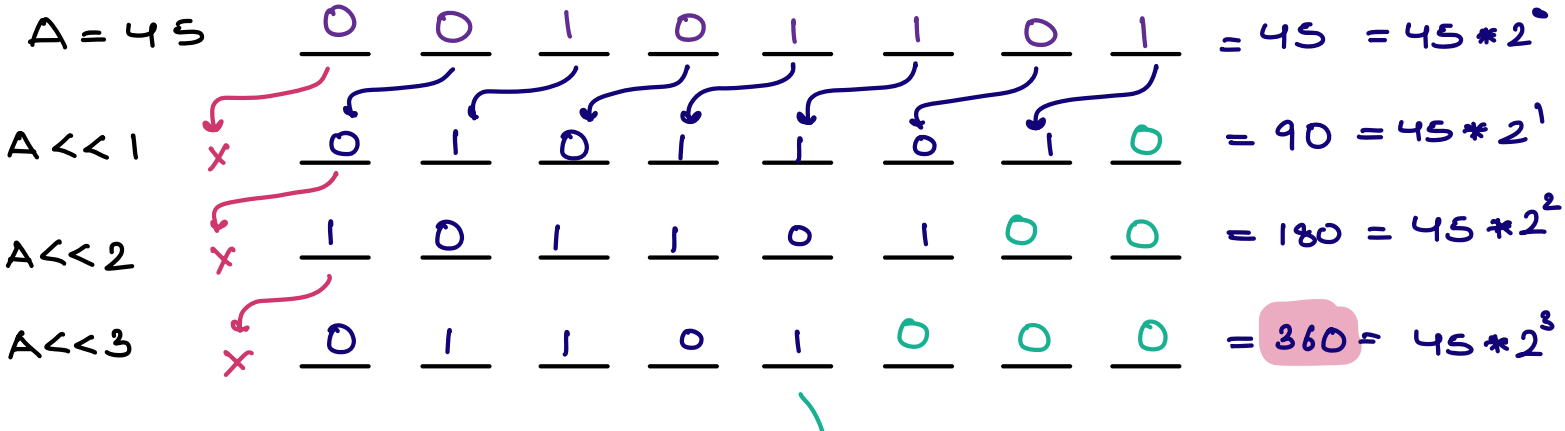
Ans → XOR of complete array.

```
int val = 0
for (i=0; i<n; i++){
    val = val ^ ar[i];
}
return val;
```

TC: O(N)
SC: O(1)

Left Shift Operator

int = 8 bits ⟶ Just for understanding



→ Ans = 104 & not 360

Overflow

$a << n = a * 2^n$

$1 << n = 2^n$

Right Shift Operator

A = 20	0	0	0	1	0	1	0	0	= 20
A >> 1	0	0	0	0	1	0	1	0	x = 10 = $\frac{20}{2}$
A >> 2	0	0	0	0	0	1	0	1	x = 5 = $\frac{20}{2^2}$
	0	0	0	0	0	0	1	0	x ⇒ 2 = $\frac{20}{2^3}$
	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0	

$a >> n = \frac{a}{2^n}$

No overflow

Doubts

arr = {1, 3, 1, 4, 3}

1 = 0 0 1

3 = 0 1 1

4 = 1 0 0

val = 0 0 0

^ 0 0 1

0 0 1

^ 0 1 1

0 1 0

0 0 1

0 1 1

1 0 0

1 1 1

0 1 1

Ans = 4 = 1 0 0

Stucked at Question

→ Hints

→ Video solutions

→ Error in whatsapp

→ Google

→ TA

→ Instructor