Questions ✓

Answer → private ✓

Join by → 7:05 AM

Sum of first N natural numbers → $\frac{N*(N+1)}{2}$

$S = 1 + 2 + 3 + \ldots + N$

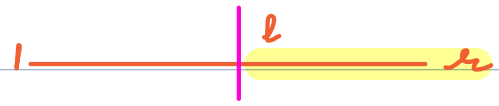
$S = N + (N-1) + (N-2) + \ldots + 1$

$2S = (N+1) + (N+1) + (N+1) \ldots$ N times

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

[3 8] → 3 4 5 6 7 8 Ans = 6

[l r]



r

r - l + 1

l - 1

r - (l - 1)

8 - 3 + 1 = 6

= r - l + 1

Q → Given an integer N, count # factors of N.

N = 5 → {1 5} Ans = 2

N = 20 → {1 2 4 5 10 20} Ans = 6

check if x is factor of N → N % x = 0

Smallest factor → 1

Largest factor → N

N = 24 → {1 2 3 4 6 8 12 24}

N = 10 → {1 2 5 10}

Bruteforce →

```
cnt = 0
for i → 1 to N {
    if (N % i == 0) cnt++
}
return cnt
```

iterations = N

Lets consider for a server it takes 1 sec to run 10^8 iterations.

$N = 10^8 \longrightarrow$ 1 sec

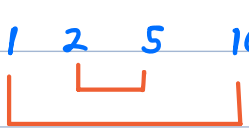
$N = 10^9 \longrightarrow 10 * 10^8$ iterations $\rightarrow$ 10 sec

$N = 10^{18} \longrightarrow 10^{10} * 10^8$ iterations $\rightarrow 10^{10}$ sec

$$\frac{10^{10}}{60 * 60 * 24 * 365} \approx 317 \text{ years}$$

Observations →

N = 24 → {1 2 3 4 6 8 12 24}



N = 10 → {1 2 5 10}

N = 24	a	b
	1	24
	2	12
	3	8
	4	6

N = 10	a	b
	1	10
	2	5

$N = a * b, \quad a <= b$

$b = N/a$

$a <= N/a$

$\Rightarrow a^2 <= N \Rightarrow \boxed{a <= \sqrt{N}}$

```
cnt = 0
for a → 1 to √N {
    if (N % a == 0) {
        // b = N/a
        cnt += 2
    }
} return cnt
```

N = 24 → 8

a = 1 2 3 4 5 (stop)

cnt = 0 2 4 6 8

N = 16 → 5 {1 2 4 8 16}

a → {1, 2, 4}

```
cnt = 0
for a → 1 to √N {
    if (N % a == 0) {
        if (a == N/a) cnt++
        else          cnt += 2
    }
} return cnt
```

1 * 16

2 * 8

4 * 4

N = 16 → 5 ✓

iterations = $\sqrt{N}$

$N = 10^{18} \longrightarrow \sqrt{10^{18}} = 10^9$ iterations → 10 sec

317 years → 10 sec

---

of iterations

1) `i = N  // > 0`
`while ( i > 1 ) {`
`  i /= 2`
`}`

$i = N \rightarrow \frac{N}{2} \rightarrow \frac{N}{2^2} \rightarrow \dots \boxed{\frac{N}{2^K} = 1}$

$\Rightarrow N = 2^K \Rightarrow K = \log_2(N)$

iterations = $\log_2(N)$

2)
```
for i → 1 to N {
    for j → 1 to N {
        print (i+j)
    }
}
```

iterations = N^2

3)
```
for (i = 0 ; i < N ; i++) {
    for (j = 0 ; j <= i ; j++) {
        print (i+j)
    }
}
```

i	j	# iterations
0	[0 0]	1
1	[0 1]	2
⋮		
(N-1)	[0 (N-1)]	N

$\frac{N * (N+1)}{2} \rightarrow N^2$

Big O → Rate of growth of function.

$O\left(\frac{N * (N+1)}{2}\right) \rightarrow O\left(\frac{N^2}{2} + \frac{N}{2}\right) \rightarrow O(N^2)$

How to calculate $TC = O(\#iterations)$?

1) calculate # iterations

2) Neglect lower order term (% contribution is very less for large value of N)

3) Ignore constant coefficient

1) $f(n) = \cancel{3n^2} + \cancel{4n} - \cancel{2}$

$O(f(n)) \rightarrow O(n^2)$

2) $f(N) = \sqrt{N} - 2\log(N) + 3N^2 + 5N\log(N) - 2 + N!$

$O(f(N)) \rightarrow O(N!)$

```
for i → 0 to (N-1) {
    if (A[i] == K)   return true
}
return false
```

i/p

SC = O(1)

#iterations

best: (A[0] == K) → 1

worst: N ✓

TC = O(N)

Space Complexity → Rate of growth of memory wrt i/p.

int → 4B long → 8B

```
// i/p → N
{ int x = 10          → 4B      Total = 12B
  long y = x * N      → 8B
}
```

SC = O(1)

2) // i/p → N

```
{ int x = N/10        → 4B          total = (4 + 4N) B
  int a[] = new int[N] → (4*N) B     SC = O(N)
}
```

Input → [Algorithm] → Output

Extra space apart from i/p & o/p.

TLE → (Time Limit Exceeded)

Processing speed of online editors → 1 GHz

10^9 instructions / sec

Allowed time limit = 1 sec

a+b,

a = 2, etc.

Let say, in 1 iteration → 10 to 100 instructions

10^9 instructions → 1 sec

10^7 * 100 instructions = 10^7 iterations

10^8 * 10 instructions = 10^8 iterations

Usually in 1 sec → 10^7 to 10^8 iterations

Constraints

TC = O(N²)

$1 <= N <= 10^3$ → $10^3 * 10^3 = 10^6$ iterations ✓

$1 <= N <= 10^5$ → $10^5 * 10^5 = 10^{10}$ iterations X (TLE)

Bitwise Operators

1 → set bit / true

0 → unset bit / false

A	B	A & B (AND)	A | B (OR)	A ^ B (XOR)
0	0	0	0	0
0	1	0	1	1
1	0	0	1	1
1	1	1	1	0

XOR → 1) add without carry

2) same same puppy shame ✓

2 → 1 0 (carry)

res = 1, ∀ bits = 1 | res = 1, any bit = 1

A	~A (Not)
1	0
0	1

3 2 1 0

12 → 1 1 0 0

9 → 1 0 0 1

12 & 9 → 1 1 0 0 & 1 0 0 1 = 1 0 0 0 → 8 (Ans)

12 ^ 9 → 1 1 0 0 ^ 1 0 0 1 = 0 1 0 1 → 5 (Ans)
