- Welcome to PS + DSA Module ⭐
- Manisha Pawar
- Graduated from MSIT Delhi (IT departement)
- Pepcoding (DSA Instuctor + content creator)
- 2+ years of Teaching Experience

→ Programming constructs

→ Problem solving (efficient)

Q.1 Count of factors

24 → 1, 2, 3, 4, 6, 8, 12, 24

10 → 1, 2, 5, 10

N , factors : 1 to N

N = 10

i → 1 2 3 4 5 6 7 8 9 10

count = ~~1~~ ~~2~~ ~~3~~ 4

itr : N

```
int   countfactors (int N) {
      int count = 0;

      for (int i = 1; i <= N; i++) {
            if (N % i == 0) {
                  // i is factor of N
                  count++;
            }
      }
      return count;
}
```

Execution time

→ value of N

→ system configuration

Assume

10^8 iteration per second

N	itrations (N)	time
10^8	10^8	1 sec
10^9	10^9	10 sec
10^{18}	10^{18}	10^{10} sec = 317 years

10^8 itr → 1 sec

1 itr → $\frac{1}{10^8}$ sec

10^9 itr → $\frac{1}{10^8} \times 10^9$

= 10 sec

1 itr → $\frac{1}{10^8}$ sec

10^{18} itr → $\frac{1}{10^8} \times 10^{18}$

= 10^{10} sec

Improvisation

$i * j = N$ (both i and j are factors of N)

$j = \frac{N}{i}$ (both i and N/i are factors of N)

N = 24

i	$\frac{N}{i}$
1	24
2	12
3	8
4	6
6	4
8	3
12	2
24	1

$i <= \frac{N}{i}$

$i * i <= N$

$i <= \sqrt{N}$

N = 100

i	$\frac{N}{i}$
1	100
2	50
4	25
5	20
10	10
20	5
25	4
50	2
100	1

Observations a) all the factor of N are present in the 1st part.

b) we are in 1st part till $i <= \sqrt{N}$.

```
int   countfactors (int  N) {
    int  count = 0;
    for (int  i = 1;  i <= √N ;  i++) {
        if (N % i == 0) {
            if ( i == N/i) {
                count++;
            }
            else {
                // both  i  and  N/i  are  factors
                count += 2;
            }
        }
    }
    return  count;
}
```

iterations → $\sqrt{N}$

$N = 15$

$\sqrt{15} \approx 3$

i	count	
1	2	(1, 15)
2		
3	4	(3, 5)

$N = 4$

$\sqrt{4} = 2$

i	count	
1	2	(1, 4)
2	3	(2, 2)

N	itr ($\sqrt{N}$)	time
10^{18}	10^{9}	10sec

10^{8} itr → 1 sec

10^{9} itr → $\frac{1}{10^{8}} \times 10^{9}$

$= 10$ sec

Q.2 Check if no. is prime or not.

prime no. → two factors (1 and no. itself)

{10, 11, 23, 2, 25, 27, 31}

↳ primes: 11, 23, 2, 31

```
boolean is_prime(int N) {

        if(count factors(N) == 2) {
                return true;
        }
        else {
                return false;
        }
}
```

$\sqrt{N}$ iterations

Q.3 Given an Array, reverse it.

A = { 10, 20, 30, 40 }
0 1 2 3

$\{ \overset{40}{\cancel{10}}, \overset{30}{\cancel{20}}, \overset{20}{\cancel{30}}, \overset{10}{\cancel{40}} \}$
0 1 2 3

s	e
0	3
1	2
2	1

A = { 10, 20, 30, 40, 50, 60, 70 }
0 1 2 3 4 5 6

$\{ \overset{70}{\cancel{10}}, \overset{60}{\cancel{20}}, \overset{50}{\cancel{30}}, 40, \overset{30}{\cancel{50}}, \overset{20}{\cancel{60}}, \overset{10}{\cancel{70}} \}$
0 1 2 3 4 5 6

s
e

s	e
0	6
1	5
2	4
3	3

work till s < e

```
void   reverse ( int [ ] A ) {
    int s = 0;
    int e = A.length - 1;

    while ( s < e ) {
        // swap  A[s]  and  A[e]
        int temp = A[s];
        A[s] = A[e];
        A[e] = temp;
        s++; e--;
    }
}
```

0 1 2 3 4
$\cancel{10}$ $\cancel{20}$ 30 $\cancel{40}$ $\cancel{50}$
50 40 20 10

0 1 2 3
$\cancel{10}$ $\cancel{20}$ $\cancel{30}$ $\cancel{40}$
40 30 20 10

Q.4 Reverse part of an array.

A =	{ 10,	19,	40,	50,	25,	64,	38,	37 }
	0	1	2	3	4	5	6	7

s = 2

e = 5

{ 10,	19,	64,	25,	50,	40,	38,	37 }
0	1	2	3	4	5	6	7

```
void  reversePart ( int [ ] A, int s, int e ) {
        while ( s < e ) {
            // swap  A [s]  and  A [e]
            int  temp = A [s] ;
            A [s] = A [e] ;
            A [e] = temp ;
            s++ ; e-- ;
        }
}
```

		25		19		
A =	10	~~19~~	24	~~25~~	30	18
	0	1	2	3	4	5
			s			
			e			

s = 1

e = 3

Q-5 Given an array, rotate it from last to first k times.

Google, amazon

constraints : (i) don't use extra array
(ii) do it in efficient way

A = 10 20 30 40 50 60 K=3

↓

60 10 20 30 40 50

↓

50 60 10 20 30 40

↓

40 50 60 10 20 30

A = 19 30 14 16 18 17 38 K=4

16 18 17 38 19 30 14

A = 19 30 14 16 18 17 38

k = 4

16 18 17 38 19 30 14

rev

38 17 18 16 | 14 30 19

rev rev

16 18 17 38 19 30 14

k = 4

10	20	30	40	50	60
0	1	2	3	4	5

60	50	40	30	20	10
0	1	2	3	4	5

30	40	50	60	10	20
0	1	2	3	4	5

i) reverse entire array

(ii) reverse the first k elements

(iii) rev. the remaining elements

```
void   rotate (int [ ]A, int k ) {
         int n= A.length;

         reversePart (A, 0, n-1);
         reversePart (A, 0, k-1);
         reversePart (A, k, n-1);
}
```

k=3

A= 10 20 30 40 19 50 60

0 1 2 3 4 5 6

↓

19 50 60 10 20 30 40

~~60~~ ~~50~~ ~~19~~ ~~40~~ ~~30~~ ~~20~~ ~~10~~

0 1 2 3 4 5 6

19 50 60 10 20 30 40

```
void   rotate (int [ ]A, int k ) {
         int n= A.length;

         reversePart (A, 0, n-1); ✓
         reversePart (A, 0, k-1);  }  Array
         reversePart (A, k, n-1);  }  index
                                      out of bound
}
```

What if k > n

A= 10 20 30 40 50

k = 12

10 20 30 40

k = 4

n = 4

↓

40 10 20 30

↓

30 40 10 20

↓

20 30 40 10

↓

10 20 30 4

n = 5 nr : 2

k = 12

n = 6 nr = 3

k = 27

```
void rotate (int [] A, int k) {
    int n= A. length;
    K = K % n;
    reversePart (A, 0, n-1);
    reversePart (A, 0, K-1);
    reversePart (A, K, n-1);
}
```

log basics

$\log_b a$ b power what is equals to a

$\log_b a = c$

$\boxed{b^c = a}$

$\log_2 64 = 6$

$\log_2 8 = 3$

$\log_{10} 10000 = 4$

$\log_3 81 = 4$

***

i) $N = 2^k$

$k = \log_2 N$

(ii) $\log_b b^N = N$

Problem solving and DSA

- Time complexity
- Arrays : Prefix sum, subarrays, sliding window, 2D matrices.
- Bit manipulation
- Hashing
- Recursion
- Sorting
- Searching
- 2 pointer technique
- Strings
- Linked list
- Trees, BST, heaps
- Dynamic programming
- Graphs

52 classes

Doubts

$$\log_b a = ?$$

$$b^{?} = a$$

$$\log_2 31 = \underline{\underline{4.954}}$$

↳ int : 4

$$\log_2 32 = 5$$

$$2^{\cancel{?}5} = 32$$

$$\log_2 16 = 4$$

$$2^{\cancel{?}4} = 16$$

$$\log_3 81 = 4$$

$$3^{\cancel{?}4} = 81$$