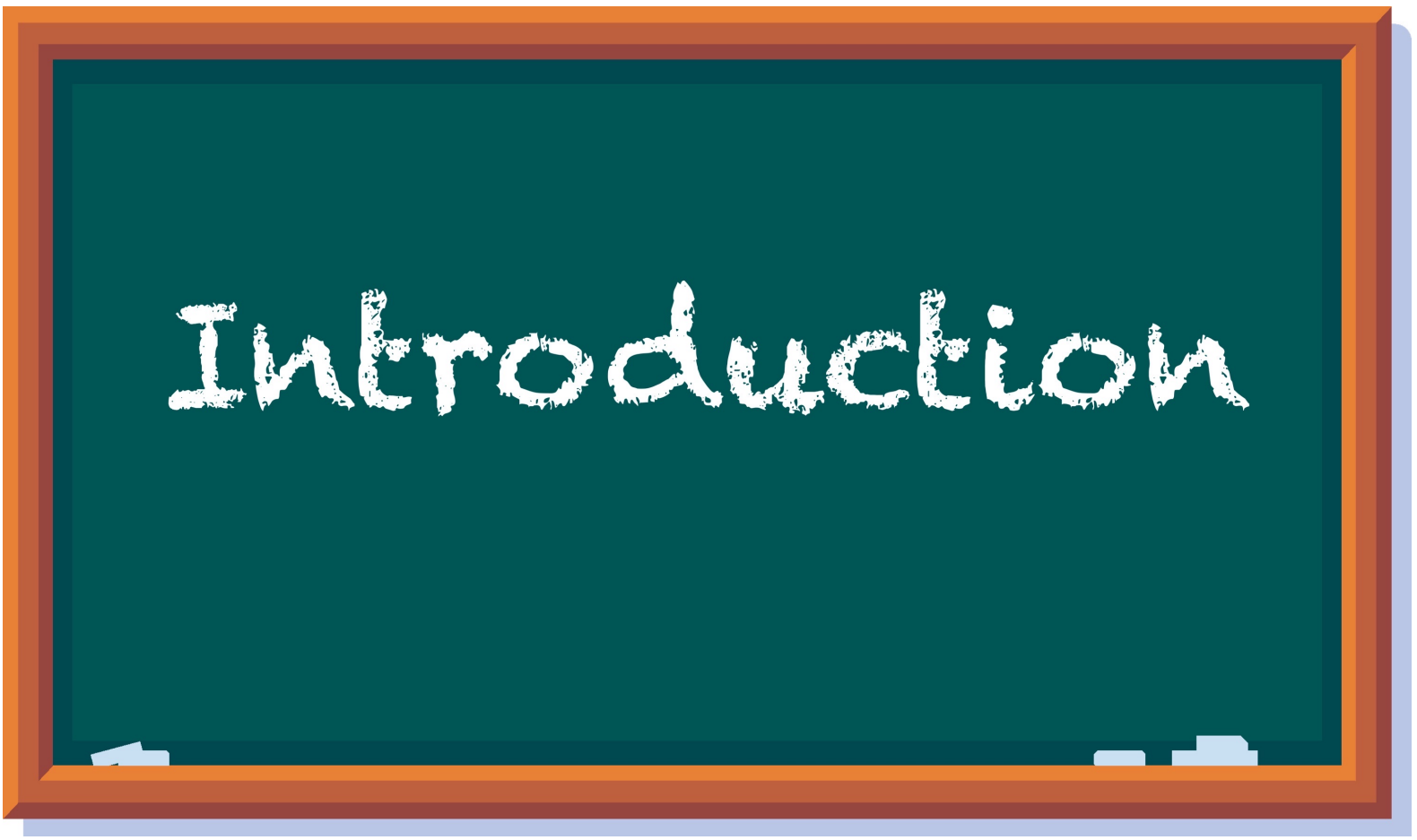
SCALER

Introduction to Problem Solving

TABLE OF CONTENTS


Introduction

Few terms that you shall see/hear throughout the course!

1. PSP (Problem Solving Percentage) - Solved Assignment Problems / Total Open Assignment Problems

- There are two types of section - Assignment and Additional. Assignment section consists of implementation of the problems done in class. PSP is calculated based on only Assignment Problems

- Additional Problems are slight modifications of assignment problems, they are not part of PSP but once you're done with assignment, we highly recommend to complete additional problems as well.

- Try to keep PSP least 85% no matter what. It shall really help you stay focused and we have seen in the past that people with ***>= 85%***, do well in interviews.

2. Attendance

- Try to maintain at least ***75% attendance*** either through classes or by watching recordings.

- Though I will recommend you to come to classes regularly because otherwise it may create backlogs.

- So, I expect all of you to attend live classes and if for any reasons you are unable to, then please send me a message stating the reason.

Next Month

- Introduction to Problem Solving
- Time Complexity
- Introduction to Arrays
- Prefix Sum
- Carry Forward
- Subarrays
- 2D Matrices
- Sorting Basics
- Strings Basics
- Bit Manipulation Basics
- Interview Problems
- Contest [covers full Intermediate DSA]

Contest will be organised after Intermediate Module

- It will be for 1.5 hours and will be conducted within class duration followed by Contest Discussion (Instructor shall be discussing contest problems).

- It will consist of 3 questions and we expect you to solve >=2 problems. If for any reason you are unable to solve, then we shall also be having re-attempts as well. (We will provide more info on re-attempts moving forward)

- Contests are critical to retaining what you have learnt and measuring where you need improvement. Please take contests seriously.

FAQs

- Notes will be uploaded after the class.
- Assignment will be unlocked after the class ends.
- There is no deadline for assignments.
- If asking a questions, ask in public chat.
- If answering a questions, answer in private chat.

Factor

number that divides N

N % i == 0

6 ⇒ 1, 2, 3, 6

< *Question* > : Given N. Find the count of factors of N. (N > 0)

1. **N = 24**

1, 2, 3, 4, 6, 8, 12, 24

ans = 8

2. **N = 10**

1, 2, 5, 10

ans = 4

```
count = 0
for( i=1 ; i<=N ; i++ ){
|       if ( N%i == 0)
|           count++
}
```

loop runs N times

N = 100 100 iterations

Info: 10^8 iterations take 1 sec

N	iter	time
10^8	10^8	1 sec
10^9	10^9 $= 10 \times 10^8$	10 sec
10^{18}	$= 10^{10} \times 10^8$	10^{10} sec $\approx$ 317 years

Optimisation

$i \times ? = N$ $\quad ? = N/i$

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

$N = 24$

$i \leq N/i$

$i \times i \leq N$

$i \leq \sqrt{N}$

```
count = 0
for( i=1 ; i*i<=N ; i++){
    if (N%i == 0)
        count += 2
}
```

N = 25

i	N/i
1	25
5	5

< / > Code

```
count = 0
for( i=1 ; i*i <= N ; i++ ){
    if (N%i == 0) {
        if( i != N/i )
            count += 2
        else
            count += 1
    }
}
```

N	iters	time
10^{18}	$\sqrt{N} = 10^9$	10 sec

Prime Numbers

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

< *Question* > : Given a number N. Check if it is prime or not.

Prime ⇒ exactly 2 factors

< / > *Pseudo Code*

```
count = 0
for( i=1 ; i*i <= N ; i++ ){
    if (N%i == 0) {
        if( i != N/i )
            count += 2
        else
            count += 1
    }
}
if( count == 2)
    return true
else return false
```

$1+2+3+\ldots\ldots+N = \frac{N(N+1)}{2}$

$\frac{100\times101}{2} = 5050$

Range

[→ include

(→ exclude

[1 , 11] → 1, 2, 3, 4, ..., 10, 11

(11)

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

[a , b] → $a, a+1, a+2, \ldots\ldots b-1, b$

$\Rightarrow b-a+1$

[a , b) → $b-a$

(a , b] → $b-a$

What is a iteration?

Number of times a loop runs

Quiz- 1

```
for(i=1; i≤N; i++){
    if(i==N) {break}
}
```

i = 1
i = 2
i = 3
⋮
N-1
N

ans = N

Quiz- 2

```
s=0;
for(int i=0; i≤100; i++){
    s=s+i+i²
}
```

i = 0
i = 1
⋮
i = 100

i ⇒ [0, 100]

100 - 0 + 1

= 101

Quiz- 3

```
for(i=1; i≤N; i++){
    if(i%2==0){
        print(i);
    }
}
for(j=1; j≤n; j++){
    if(j%2==0){
        print(j);
    }
}
```

N

M

ans = N + M

Geometric Progression

- Series where the ratio of two consecutive terms remains same.

$a = 5$, $r = 2$, $n = 8$

5 , 10, 20 , 40 , 80 , 160 , 320 , 640

$a \Rightarrow$ first term $\quad r \Rightarrow$ ratio

$n \Rightarrow$ number of terms

$$a, ar, ar^2, ar^3, \text{----}, ar^{n-1}$$

$$a + ar + ar^2 + \text{----} + ar^{n-1}$$

$$\Rightarrow a(1 + r + r^2 + \text{---} + r^{n-1})$$

$$\boxed{\frac{a(r^n - 1)}{r - 1}}$$

Put $a = 5$, $r = 2$, $n = 8$

$$\frac{5(2^8 - 1)}{2 - 1} = 1275$$

How to compare two algorithms?

N = 10^8. Given N elements, sort the elements in increasing order.

Algo 1

15 sec

Windows XP

↓ mac

8 sec

(java)

Algo 2

10 sec (Py)

Mac Pro

↓ java

6 sec

What do we use for comparison ?

⇒ # of iterations

Big O

{done}