Analysis of Time Complexity

- Running time depends on many factors (processor, intigrated or dedi'cated GPU, refreshrate, etc), but we compute it based on <u>input size</u>.

∴ We calculate <u>Rate of Growth of time wrt input.</u>

<u>Assumption:</u>

1. All arithmetic & logical operations take =1 unit time.

   a+b → 1unit

   2×3 → 1unit

   2 &3 → 1unit

2 All return statement take 1 unit time

   return 5 → 1 unit

   return val → 1 unit

eg: <u>Constant Time :</u>

```
int  sum( int  a, int  b)
{  int  c;
   c= a+b;          1unit + 1unit = 2
   return  c;       1unit
}
```

$\therefore$ Total cost $= 2 + 1$

$= 3$ unit

i.e. Constant time.

$\therefore$ Time Complexity $= O(1)$

Big Oh

i.e. Worst Case Time Complexity

eg: Linear Time

	1	2	3	4	5	6	7	8	9	10
a	3	2	7	6	1	13	17	19	15	29

```
                                  10
int     sum( int  a[10] , int  n )
 { int  tot;
     tot =0;                               — 1 unit
     for( int  i=1; i<=n; i++) }             n                         = 2×n = (2n)
           tot = tot + a[i];          } — 1unit + 1unit = 2.
     return  tot;                          — 1unit
 }
```

Total Cost = 1 unit + 2n unit + 1 unit

$= (2 + 2n)$ unit

$= (2n + 2)$

Lets assume:	$(\underline{2n}+2)$
If $n=5$,	$10+2$
If $n=50$,	$100+2$
If $n=100$,	$200+2$
If $n=1000$,	$2000+2$
If $n=$ 1 Lakh ,	2 Lakh+2
If $n=$ 1cr ,	2cr+2
If $n=$ 10000cr ,	20000cr+2
⋮	⋮ ⋮

$$\therefore \quad (2n+2) \approx 2n$$

If	$2n$
$n=5$,	2×5
$n=$ 5Lk ,	$2\times$5Lk
h= 10000 cr ,	$2\times$10000 cr
⋮	⋮ ⋮

$$2n \approx \underline{\underline{n}}$$

$$\boxed{T(n) = O(n)}$$

General Rules :

① **Loop:**

$$\left.\begin{array}{ll} \texttt{for(i=1; i<=n; i++)} & \\ \{ & \\ \quad a=a+b; & — \; C_1 \\ \quad c=c+d; & — \; C_2 \\ \} & \end{array}\right\} (C_1+C_2)n$$

$$\therefore T(n) = (C_1+C_2)n$$

$$= C \cdot n$$

$$\therefore T(n) = O(n).$$

③ **Nested for Loops:**

$$\left.\begin{array}{ll} \texttt{for(i=1; i<=n; i++)} & — \; n \\ \quad \texttt{for(j=1; j<=n; j++)} & — \; n \\ \quad \{ & \\ \qquad a = i+j \; ; & — \; c \\ \quad \} & \end{array}\right\} c \cdot n \cdot n$$

$$\therefore T(n) = c \cdot n^2 \qquad \therefore \boxed{T(n) = O(n^2)} .$$

4 Consecutive Stmts:

```
a= a+b;                  —— c1
for( i=1; i<=n; i++)     } c2·n
     x=x+y;
for ( j=1; j<=n; j++)    } c3·n
     c= c+d
return c;                } c4
```

$$T(n)= C_1 + \underline{C_2 n + C_3 n} + C_4$$

$$= (C_2+C_3)n + \underline{C_1+C_4}$$

$$= C_1 \cdot n + \underline{C_2}$$

$$= C_1 \cdot n$$

$$= n$$

$$\therefore \boxed{T(n)= O(n)}.$$

5 Consecutive Stmnts & Nested mixed:

```
a=a+b;                                        — c1
for (i=1; i<=n; i++)                          } c2 n
    c=c+d;
for( j=1; j<=n; j++)                          }
    for (k=1; k<=n; k++)                      }  c3 · n^3
        for( w=1; w<=n, w++)                  }
            z= j+k+w                          }

for (q=1; q<=n; q++)                          }
    for( p=1; p<=n; p++)                      }  c4 n^2
        z=q*p;                                }
```

$$T(n) = c_1 + c_2 n + c_3 n^3 + c_4 n^2$$

$$= c_2 \cdot n + c_3 \cdot n^3 + c_4 \cdot n^2$$

$$= n + n^3 + n^2$$

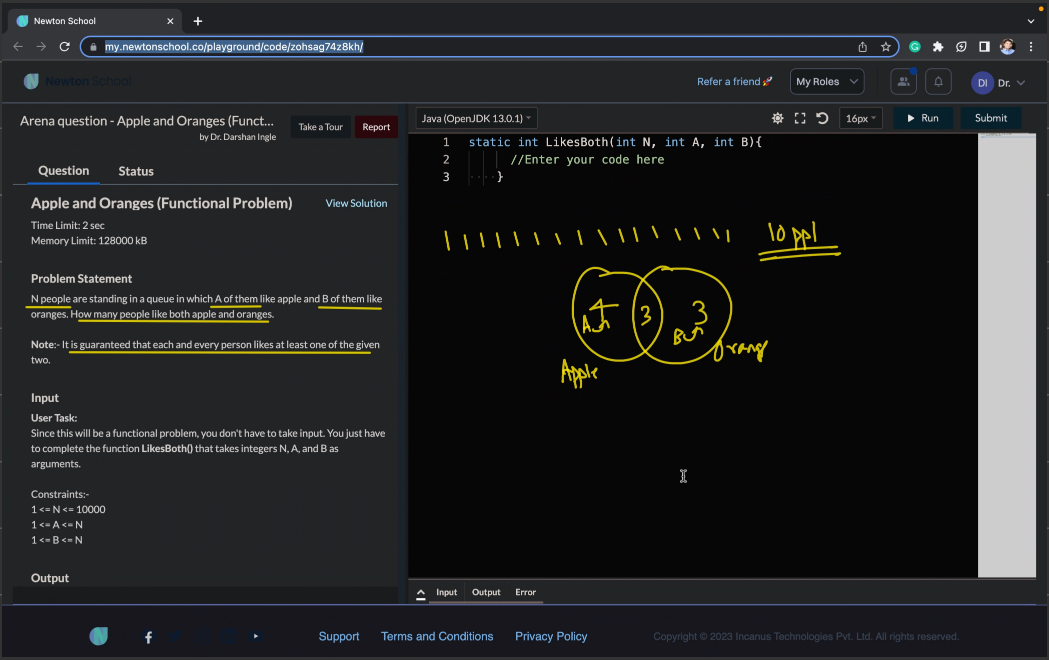
$$\therefore T(n) = O(n^3)$$

⑥ if_else stmnt:

```
if  condition
  {
      // 1 for loop        ... $O(n)$
  }
else
  {
      // 2 nested for loops        .... $O(n^2)$
  }
```

$T(n) = O(n^2)$.

https://my.newtonschool.co/playground/code/zohsag74z8kh/



Arena question - Apple and Oranges (Funct...
by Dr. Darshan Ingle

Apple and Oranges (Functional Problem)

Time Limit: 2 sec
Memory Limit: 128000 kB

Problem Statement

N people are standing in a queue in which A of them like apple and B of them like oranges. How many people like both apple and oranges.

Note:- It is guaranteed that each and every person likes at least one of the given two.

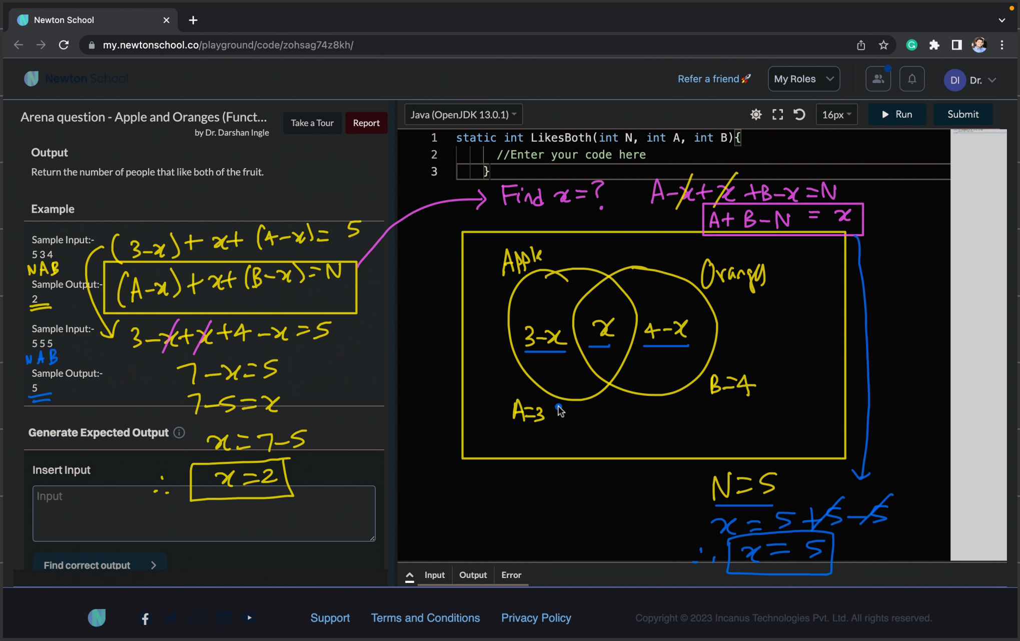
Input

User Task:
Since this will be a functional problem, you don't have to take input. You just have to complete the function **LikesBoth()** that takes integers N, A, and B as arguments.

Constraints:-
1 <= N <= 10000
1 <= A <= N
1 <= B <= N

Output

```
static int LikesBoth(int N, int A, int B){
    //Enter your code here
    }
```



Arena question - Apple and Oranges (Funct...
by Dr. Darshan Ingle

Output

Return the number of people that like both of the fruit.

Example

Sample Input:-
5 3 4
NAB
Sample Output:-
2

Sample Input:-
5 5 5
NAB
Sample Output:-
5

$(3-x)+x+(4-x)=5$

$\boxed{(A-x)+x+(B-x)=N}$

$3-\cancel{x}+\cancel{x}+4-x=5$

$7-x=5$

$7-5=x$

$x=7-5$

$\therefore \boxed{x=2}$

Generate Expected Output

Insert Input

Input

```
static int LikesBoth(int N, int A, int B){
    //Enter your code here
    }
```

Find $x=?$

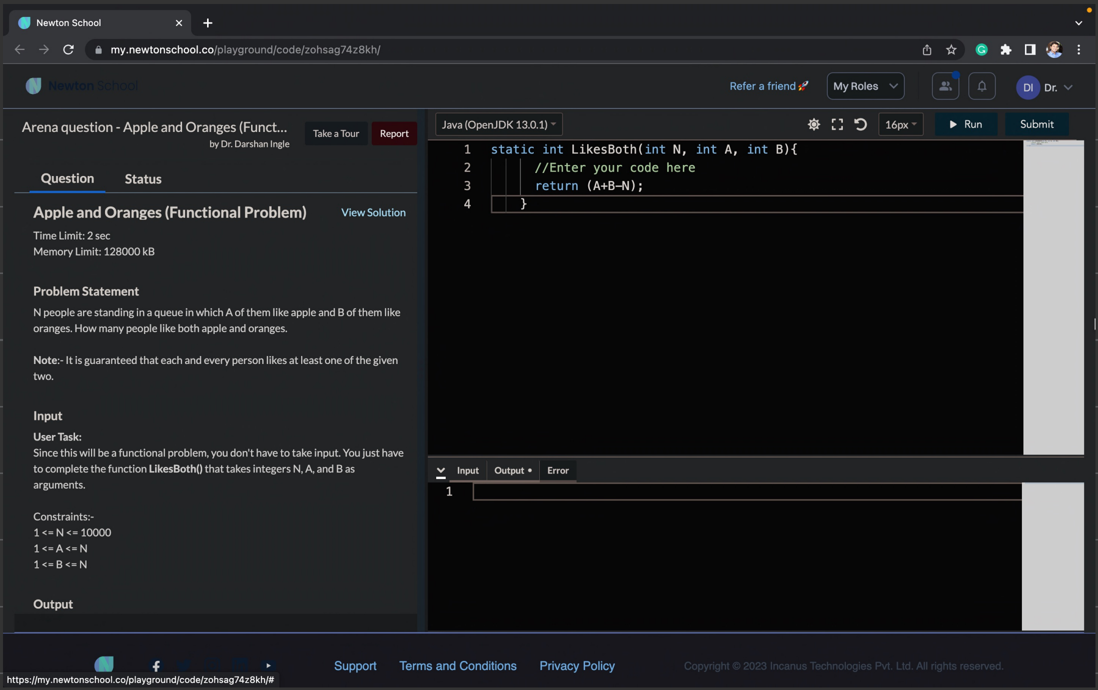
$A-\cancel{x}+\cancel{x}+B-x=N$

$\boxed{A+B-N=x}$

$N=5$

$x=5+\cancel{5}-\cancel{5}$

$\therefore \boxed{x=5}$



Arena question - Apple and Oranges (Funct...

by Dr. Darshan Ingle

Apple and Oranges (Functional Problem)

Time Limit: 2 sec
Memory Limit: 128000 kB

Problem Statement

N people are standing in a queue in which A of them like apple and B of them like oranges. How many people like both apple and oranges.

Note:- It is guaranteed that each and every person likes at least one of the given two.

Input

User Task:
Since this will be a functional problem, you don't have to take input. You just have to complete the function **LikesBoth()** that takes integers N, A, and B as arguments.

Constraints:-
1 <= N <= 10000
1 <= A <= N
1 <= B <= N

Output

```java
static int LikesBoth(int N, int A, int B){
    //Enter your code here
    return (A+B-N);
}
```