

REPORT
on
Master Coding and Competitive Programming - Part-1
(*Course Code: V23CSES01*)
B.TECH V Semester (2023 - 2027 Batch)
Academic Year: 2025-26



SRI VASAVI ENGINEERING COLLEGE (Autonomous)

Pedatadepalli , Tadepalligudem - 534101

Department of Computer Science & Engineering (Accredited by NBA)

SRI VASAVI ENGINEERING COLLEGE (Autonomous)
PEDATADEPALLI, TADEPALLIGUDEM.



Certificate

This is to certify Mr. **Jaideep Vantipalli** bearing Roll No. **23A81A0562** of CSE Branch of V Semester submitted the Report on _____ as a part of **Master Coding and Competitive Programming - Part-1** (Course Code:V23CSES01) during the academic year 2025-2026.

Faculty In charge

Head of the Department

Examiner1

Examiner2

Geeks For Geeks Profile :

<https://www.geeksforgeeks.org/user/jaideepvantipalli/>

LeetCode Profile :

https://leetcode.com/u/Jaideep_Vantipalli/

INDEX

SI . NO	Topics	Page No.
1	Programs on Mathematical reasoning	5-14
2	Programs on Prime Numbers	15-20
3	Programs on Fibonacci Series	21-24
4	Programs on Arrays	25-34
5	Programs on Strings	35-44
6	Programs on Matrices	45-49
7	Programs on Series, patterns	50-54

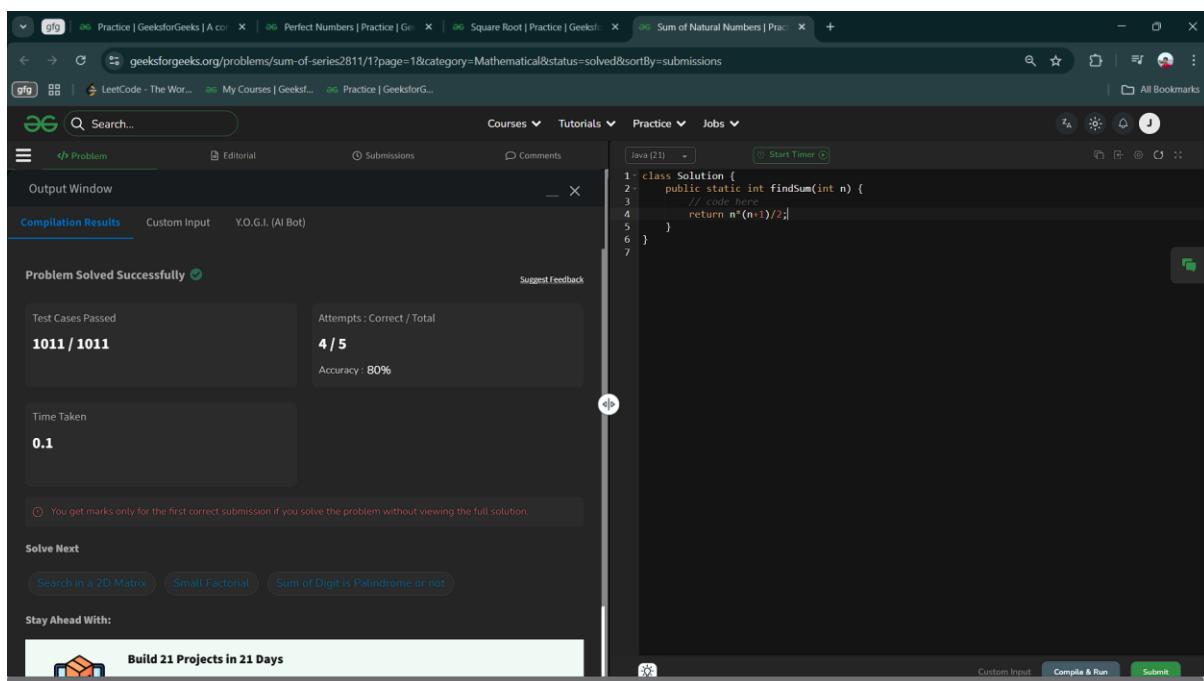
1. Programs on Mathematical reasoning .

1.1 Sum of Natural Numbers

Aim : Given an integer n. Your task is to calculate the sum of all natural numbers from 1 up to n (inclusive). If n is 0, the sum should be 0

```
class Solution {
    public static int findSum(int n) {
        return n*(n+1)/2;
    }
}
```

Output :



The screenshot shows a browser window for the GeeksforGeeks Practice platform. The URL in the address bar is geeksforgeeks.org/problems/sum-of-series2811/1?page=1&category=Mathematical&status=solved&sortBy=submissions. The page displays a Java code editor with the provided solution. Below the code, it says "Problem Solved Successfully". It shows 1011 / 1011 test cases passed, 4 / 5 attempts correct, and 80% accuracy. A note indicates that marks are only given for the first correct submission. There are also "Solve Next" and "Stay Ahead With:" sections.

1.2 Square Root

Aim : Given a positive integer n, find the square root of n. If n is not a perfect square, then return the floor value.

Floor value of any number is the greatest Integer which is less than or equal to that number.

Code :

```
class Solution {
    int floorSqrt(int n) {
        double x=Math.sqrt(n);
        if(Math.floor(x)==Math.ceil(x))
            return (int)x;
        else
            return (int)Math.floor(x);
    }
}
```

Output :

The screenshot shows a Java code editor on the GeeksforGeeks Practice platform. The code implements the logic described in the text above to calculate the floor square root of a given integer n. The code editor interface includes tabs for Java (21), Start Timer, and a code area with syntax highlighting. Below the code editor is a problem details section for 'Square Root' with difficulty 'Easy', accuracy '54.03%', and other statistics. A 'Test Cases Passed' section shows 1111 / 1111 test cases passed. The bottom right corner of the screenshot has a page number '6'.

1.3 Perfect Numbers

Aim : Given a number n, check if the number is perfect or not. A number is said to be perfect if sum of all its factors excluding the number itself is equal to the number.

Code :

```
class Solution {
    static boolean isPerfect(int n) {
        // code here
        int i=2,sum=1;
        while(i*i<=n){
            if(n%i==0){
                if (i*i!=n)
                    sum=sum+i+n/i;
                else
                    sum=sum+i;
            }
            i++;
        }
        if(sum==n)
            return true;
        else
            return false;
    }
}
```

Output :

The screenshot shows a browser window with the URL geeksforgeeks.org/problems/perfect-numbers3207/1?page=1&category=Mathematical&status=solved&sortBy=submissions. The page title is "Perfect Numbers | Practice". The code editor contains the provided Java code for the "Solution" class. The output window shows the result of running the code with input n=6, which outputs true, indicating it's a perfect number. The status bar at the bottom indicates "Attempts : Correct / Total 2 / 2 Accuracy : 100%".

1.4 LCM And GCD

Aim : Given two integers a and b, You have to compute their LCM and GCD and return an array containing their LCM and GCD.

Code :

```
class Solution {
    public static int[] lcmAndGcd(int a, int b) {
        // code here
        return new int[]{a*b/gcd(a,b),gcd(a,b)};
    }
    static int gcd(int m,int n){
        while(m!=n){
            if(m>n)
                m=m-n;
            else
                n=n-m;
        }
        return m;
    }
}
```

Output :

The screenshot shows the GeeksforGeeks online judge interface for the 'LCM And GCD' problem. The problem details are as follows:

- Difficulty:** Basic
- Accuracy:** 37.02%
- Submissions:** 225K+
- Points:** 1

The problem statement asks for two integers a and b, to compute their LCM and GCD and return an array containing their LCM and GCD.

Examples:

Input: a = 5 , b = 10	Output: [10, 5]
Explanation: LCM of 5 and 10 is 10, while their GCD is 5.	

Output Window:

```
Input: a = 14 , b = 8
Output: [56, 4]
```

Compilation Results:

Problem Solved Successfully

Test Cases Passed: 1111 / 1111

Attempts : Correct / Total: 3 / 3

Accuracy: 100%

Time Taken: 00:00:00

Custom Input: Compile & Run: Submit:

The code editor shows the Java code provided above. The code uses Euclid's algorithm for finding the GCD and then calculates the LCM using the formula $\text{LCM} = \frac{a \times b}{\text{GCD}}$.

1.5 Sum 1 to n Divisors

Aim : Given a positive integer n , The task is to find the value of $\sum_i F(i)$ where i is from 1 to n and function $F(i)$ is defined as the sum of all divisors of i .

Code :

```
class Solution {
    public static long sumOfDivisors(long n) {
        // code here
        int sum=0;
        for(int i=1;i<=n;i++){
            sum+=i*(n/i);
        }
        return sum;
    }
}
```

Output :

The screenshot shows the GeeksforGeeks Online Judge interface for the problem "Sum 1 to n Divisors".

- Header:** Shows tabs for Practice, Reverse digits, Factorial, Armstrong Number, Sum 1 to n Divisors, Math - LeetCode, Factorial Trailing Zeros, Palindrome Number, etc.
- Toolbar:** Includes Courses, Tutorials, Practice, and Jobs dropdowns, and a Start Timer button.
- Problem Details:**
 - Title:** Sum 1 to n Divisors
 - Difficulty:** Easy
 - Accuracy:** 43.37%
 - Submissions:** 221K+
 - Points:** 2
- Description:** Given a positive integer n , The task is to find the value of $\sum_i F(i)$ where i is from 1 to n and function $F(i)$ is defined as the sum of all divisors of i .
- Code Area:** Shows the Java code provided in the question.
- Output Window:** Displays the input and output for the sample case where $n = 4$. The input is "n = 4" and the output is "15". Below this, it shows the explanation: $F(1) = 1$, $F(2) = 1 + 2 = 3$, $F(3) = 1 + 3 = 4$.
- Compilation Results:**
 - Test Cases Passed:** 1120 / 1120
 - Attempts:** Correct / Total 2 / 2
 - Accuracy:** 100%
- Time Taken:** 0.26
- Buttons:** Custom Input, Compile & Run, Submit.

1.6 Armstrong Numbers

Aim : You are given a 3-digit number n, Find whether it is an Armstrong number or not.

An **Armstrong number** of three digits is a number such that the sum of the cubes of its digits is equal to the **number itself**. 371 is an Armstrong number since $3^3 + 7^3 + 1^3 = 371$

Code :

```
class Solution {
    static boolean armstrongNumber(int n) {
        // code here
        int t=n;
        int r,sum=0;
        while(n!=0){
            r=n%10;
            sum=sum+(r*r*r);
            n=n/10;
        }
        if(t==sum){
            return true;
        }
        else{
            return false;
        }
    }
}
```

Output :

The screenshot shows the GeeksforGeeks platform interface for solving the Armstrong Numbers problem. The code editor contains the Java solution provided above. The output window shows that all test cases have passed, with a total of 1111/1111 correct attempts and 100% accuracy. The time taken for execution was 0.13 seconds.

```
1 // User function Template for Java
2 class Solution {
3     static boolean armstrongNumber(int n) {
4         // code here
5         int t=n;
6         int r,sum=0;
7         while(n!=0){
8             r=n%10;
9             sum=sum+(r*r*r);
10            n=n/10;
11        }
12        if(t==sum){
13            return true;
14        }
15        else{
16            return false;
17        }
18    }
19 }
```

1.7 Factorial

Aim : Given a positive integer, n. Find the factorial of n

Code :

```
class Solution {

    static int factorial(int n) {

        // code here

        int fact=1;

        for(int i=1;i<=n;i++){

            fact=fact*i;

        }

        return fact;

    }
}
```

Output :

The screenshot shows a browser window with multiple tabs open, including GeeksforGeeks practice problems and LeetCode. The main focus is on the GeeksforGeeks practice interface for the Factorial problem.

Problem Details:

- Difficulty: Basic
- Accuracy: 40.58%
- Submissions: 206K+
- Points: 1

Description:

Given a positive integer, n. Find the factorial of n.

Examples:

Input: n = 5 Output: 120 Explanation: $1 \times 2 \times 3 \times 4 \times 5 = 120$
Input: n = 4 Output: 24 Explanation: $1 \times 2 \times 3 \times 4 = 24$

Output Window:

Compilation Results: 3 / 3
Attempts : Correct / Total: 3 / 3
Accuracy: 100%

Time Taken: 0.29

Custom Input, Compile & Run, Submit buttons are visible at the bottom.

1.8 Reverse digits

Aim : You are given an integer n. Your task is to reverse the digits, ensuring that the reversed number has no leading zeroes.

Code :

```
class Solution {
public:
    int reverseDigits(int n) {
        // Code here
        int r,sum=0;
        while(n!=0){
            r=n%10;
            sum=sum*10+r;
            n=n/10;
        }
        return sum;
    }
};
```

Output :

The screenshot shows the GeeksforGeeks online judge interface for the 'Reverse digits' problem. The code editor contains the provided C++ code. The output window shows the input 'n = 200' and output '200'. The compilation results show 'Problem Solved Successfully' with 100/100 test cases passed and 2/2 attempts correct, resulting in 100% accuracy. The time taken is listed as 0ms.

```
1. class Solution {
2. public:
3.     int reverseDigits(int n) {
4.         // Code here
5.         int r,sum=0;
6.         while(n!=0){
7.             r=n%10;
8.             sum=sum*10+r;
9.             n=n/10;
10.        }
11.        return sum;
12.    }
13.};
```

1.9 Palindromic Number

Aim : Given an integer x , return true if x is a palindrome, and false otherwise.

Code :

```
class Solution {
    public boolean isPalindrome(int x) {
        int temp=x,sum=0;
        while(x>0){
            int r=x%10;
            sum=sum*10+r;
            x=x/10;
        }
        return sum==temp?true:false;
    }
}
```

Output :

The screenshot shows a LeetCode problem page for "9. Palindrome Number". The code editor displays the Java solution provided above. The test results section shows the code was accepted with a runtime of 0 ms, passing all three test cases. The input field contains the value 121.

```
1 class Solution {
2     public boolean isPalindrome(int x) {
3         int temp=x,sum=0;
4         while(x>0){
5             int r=x%10;
6             sum=sum*10+r;
7             x=x/10;
8         }
9         return sum==temp?true:false;
10    }
11 }
```

1.10 Factorial Trailing Zeros

Aim : Given an integer n, return *the number of trailing zeroes in n!*

Code :

```
class Solution {
    public int trailingZeroes(int n) {
        int c=0;
        if(n<=4) return c;
        for(int i=5;n/i>=1;i=i*5){
            c=c+n/i;
        }
        return c;
    }
}
```

Output :

The screenshot shows a LeetCode problem page for 'Factorial Trailing Zeroes' (Problem 172). The code editor displays the following Java code:

```
class Solution {
    public int trailingZeroes(int n) {
        int c=0;
        if(n<=4) return c;
        for(int i=5;n/i>=1;i=i*5){
            c=c+n/i;
        }
        return c;
    }
}
```

The 'Test Result' section indicates the code has been accepted with a runtime of 0 ms, passing all three test cases (Case 1, Case 2, Case 3).

2. Programs on Prime Numbers

2.1 Prime Number

Aim : Given a number n, determine whether it is a prime number or not.

Note: A prime number is a number greater than 1 that has no positive divisors other than 1 and itself.

Code :

```
class Solution {
    static boolean isPrime(int n) {
        if(n<=1){
            return false;
        }else{
            for(int i=2;i*i<=n;i++){
                if(n%i==0){
                    return false;
                }
            }
            return true;
        }
    }
}
```

Output :

The screenshot shows the GeeksforGeeks online judge interface for the 'Prime Number' problem. The code editor contains the provided Java code for determining if a number is prime. The output window shows the code running successfully. The compilation results show 1120/1120 test cases passed, with an accuracy of 50% and 2/4 attempts correct. The page URL is geeksforgeeks.org/problems/prime-number/1?page=1&category=Prime%20Number&status=solved&sortBy=submissions.

2.2 Largest prime factor

Aim: Given a number n, your task is to find the largest prime factor of n.

Code :

```
class Solution {
    static int largestPrimeFactor(int n) {
        ArrayList<Integer> al=new ArrayList<>();
        while(n%2==0){
            if(!al.contains(2))
                al.add(2);
            n/=2;
        }
        for(int i=3;i*i<=n;i+=2){
            while(n%i==0){
                if(!al.contains(i))
                    al.add(i);
                n/=i;
            }
        }
        if(n>1){
            if(!al.contains(n))
                al.add(n);
        }
        return al.get(al.size()-1);
    }
}
```

Output :

The screenshot shows a browser window with multiple tabs open, likely on GeeksforGeeks. The active tab is for the problem "Largest prime factor". The page displays the problem statement, examples, and a code editor. The code editor contains the Java code provided above, with line numbers 1 through 25. The code uses an ArrayList to store prime factors and iterates from 3 up to the square root of n to find the largest one. The browser interface includes a search bar, navigation buttons, and a toolbar with various icons.

2.3 Prime Pair with Target Sum

Aim : Given a number n, find out if n can be expressed as a+b, where both a and b are prime numbers. If such a pair exists, return the values of a and b, otherwise return [-1,-1] as an array of size 2.

Code :

```
class Solution {
    public static ArrayList<Integer> getPrimes(int n) {
        ArrayList<Integer> al=new ArrayList<>();
        for(int i=2;i<=n/2;i++){
            if(isprime(i)){
                if(isprime(n-i)){
                    al.add(i);
                    al.add(n-i);
                    return al;
                }
            }
        }
        al.add(-1);
        al.add(-1);
        return al;
    }
}
```

```
static boolean isprime(int n){
    if(n<=1) return false;
    if(n==2||n==3) return true;
    if(n%2==0||n%3==0) return
    false;
    for(int i=5;i*i<=n;i+=6){
        if(n%i==0||n%(i+2)==0)
            return false;
    }
    return true;
}
```

Output :

The screenshot shows a Java code editor with the following code:

```
1- class Solution {
2-     public static ArrayList<Integer> getPrimes(int n) {
3-         ArrayList<Integer> al=new ArrayList<>();
4-         for(int i=2;i<=n/2;i++){
5-             if(isprime(i)){
6-                 if(isprime(n-i)){
7-                     al.add(i);
8-                     al.add(n-i);
9-                     return al;
10-                }
11-            }
12-        }
13-        al.add(-1);
14-        al.add(-1);
15-        return al;
16-    }
17-    static boolean isprime(int n){
18-        if(n<=1) return false;
19-        if(n==2||n==3) return true;
20-        if(n%2==0||n%3==0) return false;
21-        for(int i=5;i*i<=n;i+=6){
22-            if(n%i==0||n%(i+2)==0)
23-                return false;
24-        }
25-        return true;
26-    }
27- }
```

Below the code editor, the 'Output Window' displays:

- Test Cases Passed: 1115 / 1115
- Attempts: Correct / Total: 2 / 2
- Accuracy: 100%

2.4 Next Prime Number

Aim : Given an integer n. Write a program to find the first prime number greater than n.

Code :

```
class Solution {
    public static int nextPrime(int n) {
        int prime=0;
        for(int i=n+1;i>=0;i++){
            if(isprime(i)){
                prime=i;
                break;
            }
        }
        return prime;
    }

    static boolean isprime(int n){
        if(n<=1) return false;
        if(n==2||n==3) return true;
        if(n%2==0||n%3==0) return false;
        for(int i=5;i*i<=n;i+=6){
            if(n%i==0||n%(i+2)==0)
                return false;
        }
        return true;
    }
}
```

Output :

The screenshot shows a browser window with the GeeksforGeeks Practice interface. The URL in the address bar is geeksforgeeks.org/problems/next-prime-number/1?page=1&category=Prime%20Number&status=solved&sortBy=submissions. The page title is "Next Prime Number | Practice".

The main content area displays the "Next Prime Number" problem statement and examples. The problem statement asks for the first prime number greater than a given integer n. Examples show that for input 15, the output is 17 (Explanation: 17 is the next prime number), and for input 7, the output is 11 (Explanation: 11 is the prime number next to 7).

On the right side of the screen, the Java code for the solution is shown in a code editor:

```
1 // User function Template for Java
2 class Solution {
3     public static int nextPrime(int n) {
4         int prime=0;
5         for(int i=n+1;i>=0;i++){
6             if(isprime(i)){
7                 prime=i;
8                 break;
9             }
10        }
11        return prime;
12    }

13    static boolean isprime(int n){
14        if(n<=1) return false;
15        if(n==2||n==3) return true;
16        if(n%2==0||n%3==0) return false;
17        for(int i=5;i*i<=n;i+=6){
18            if(n%i==0||n%(i+2)==0)
19                return false;
20        }
21        return true;
22    }
23 }
```

The code editor includes a "Start Timer" button. Below the code editor, the status bar shows "Compilation Results" and "Custom Input Y.O.G.I. (AI Bot)".

At the bottom of the page, there is a summary of the results: "Problem Solved Successfully" with a green checkmark, "Test Cases Passed 202 / 202", "Attempts: Correct / Total 2 / 3", "Accuracy: 66%", and a "Time Taken" section.

2.5 Composite and Prime

Aim : Given two integers l and r find the absolute difference between the number of composites and the number of primes between the range l and r (both inclusive).

```
class Solution {
    public int Count(int l, int r) {
        ArrayList<Integer> prl = new ArrayList<>();
        ArrayList<Integer> cmpl = new ArrayList<>();
        for(int i=l;i<=r;i++){
            if(isprime(i))
                prl.add(i);
            else
                cmpl.add(i);
        }
        return Math.abs(prl.size()-cmpl.size());
    }
    static boolean isprime(int n){
        if(n<=1) return false;
        if(n==2||n==3) return true;
        if(n%2==0||n%3==0) return false;
        for(int i=5;i*i<=n;i+=6){
            if(n%i==0||n%(i+2)==0)
                return false;
        }
        return true;
    }
}
```

Output :

```
// User function Template for Java
class Solution {
    public int Count(int l, int r) {
        // code here
        ArrayList<Integer> prl = new ArrayList<>();
        ArrayList<Integer> cmpl = new ArrayList<>();
        for(int i=l;i<=r;i++){
            if(isprime(i))
                prl.add(i);
            else
                cmpl.add(i);
        }
        return Math.abs(prl.size()-cmpl.size());
    }
    static boolean isprime(int n){
        if(n<1) return false;
        if(n==2||n==3) return true;
        if(n%2==0||n%3==0) return false;
        for(int i=5;i*i<=n;i+=6){
            if(n%i==0||n%(i+2)==0)
                return false;
        }
        return true;
    }
}
```

2.6 Count Prime(using Sieve)

Aim :Given an integer n, return *the number of prime numbers that are strictly less than n.*

Code :

```
class Solution {
    public int countPrimes(int n) {
        if(n<=2) return 0;
        if(n==3) return 1;
        int count=0;
        boolean[] parr=new boolean[n];
        Arrays.fill(parr,true);
        parr[0]=false;
        parr[1]=false;
        for(int i=2;i*i<n;i++){
            if(parr[i]){
                for(int j=i*i;j<n;j+=i)
                    parr[j]=false;
            }
        }
        for(int i=0;i<parr.length;i++){
            if(parr[i])
                count++;
        }
        return count;
    }
}
```

Output :

The screenshot shows a browser window with the URL leetcode.com/problems/count-primes/. The page title is "204. Count Primes". The code area contains the Java solution provided above. The status bar at the bottom right indicates "Accepted" and "Runtime: 0 ms". Below the code, there are sections for "Testcase" and "Test Result" showing successful execution for three cases. The browser's address bar and various tabs are visible at the top.

3. Problems on Fibonacci Series :

3.1 Fibonacci Number

Aim : The Fibonacci numbers, commonly denoted $F(n)$ form a sequence, called the Fibonacci sequence, such that each number is the sum of the two preceding ones, starting from 0 and 1. That is,

$$F(0) = 0, F(1) = 1$$

$$F(n) = F(n - 1) + F(n - 2), \text{ for } n > 1.$$

```
class Solution {
```

```
    public int fib(int n) {
        if(n==0) return 0;
        if(n==1) return 1;
        int a=0,b=1,c=0;
        for(int i=2;i<=n;i++){
            c=a+b;
            a=b;
            b=c;
        }
        return c;
    }
}
```

Output :

The screenshot shows the LeetCode platform with the following details:

- Problem:** 509. Fibonacci Number
- Description:** The Fibonacci numbers, commonly denoted $F(n)$, form a sequence, called the Fibonacci sequence, such that each number is the sum of the two preceding ones, starting from 0 and 1. That is,
- Code:**

```
1 class Solution {
2     public int fib(int n) {
3         if(n==0) return 0;
4         if(n==1) return 1;
5         int a=0,b=1,c=0;
6         for(int i=2;i<=n;i++){
7             c=a+b;
8             a=b;
9             b=c;
10        }
11        return c;
12    }
13 }
```
- Testcase:** Case 1: Input: n = 2, Output: 1. Case 2: Input: n = 3, Output: 2. Case 3: Input: n = 4, Output: 3.
- Constraints:** 0 ≤ n ≤ 30

3.2 Fibonacci series up to Nth term

Aim : You are given an integer n, return the fibonacci series till the nth(0-based indexing) term. Since the terms can become very large return the terms modulo 10^9+7 .

```
class Solution {
```

```
    int[] Series(int n) {
        // code here
        int arr[] = new int[n+1];
        arr[0] = 0;
        arr[1] = 1;
        for (int i=2; i<=n; i++) {
            arr[i] = (arr[i-1] + arr[i-2]) % 1000000007;
        }
        return arr;
    }
}
```

Output :

The screenshot shows the GeeksforGeeks Practice interface for solving a problem titled "Fibonacci series up to Nth term".

- Problem Details:** Difficulty: Easy, Accuracy: 51.0%, Submissions: 60K+, Points: 2, Average Time: 20m.
- Description:** You are given an integer n, return the fibonacci series till the nth(0-based indexing) term. Since the terms can become very large return the terms modulo 10^9+7 .
- Example:** Input: n = 5, Output: 0 1 1 2 3 5, Explanation: 0 1 1 2 3 5 is the Fibonacci series up to the 5th term.
- Code Editor:** Java (21) code area containing the provided solution.
- Output Window:** Shows Compilation Results: Problem Solved Successfully, Test Cases Passed: 1120 / 1120, Attempts: Correct / Total: 2 / 8, Accuracy: 25%.
- Time Taken:** 1:29.
- Buttons:** Custom Input, Compile & Run, Submit.

3.3 Check if the number is Fibonacci

Aim : Check if a given number N is the Fibonacci number. A Fibonacci number is a number that occurs in the Fibonacci series.

```
class Solution {
    static String checkFibonacci(int n) {
        // code here
        if(n==0||n==1) return "Yes";
        int a=0,b=1,c=0;
        while (c<=n) {
            c=a+b;
            if(c==n) return "Yes";
            a=b;
            b=c;
        }
        return "No";
    }
}
```

Output :

The screenshot shows a browser window with the GeeksforGeeks website open. The URL is [geeksforgeeks.org/problems/check-if-the-number-is-fibonacci4654/1?page=1&category=Fibonacci&status=solved&sortBy=submissions](https://www.geeksforgeeks.org/problems/check-if-the-number-is-fibonacci4654/1?page=1&category=Fibonacci&status=solved&sortBy=submissions). The page displays a Java problem titled "Check if the number is Fibonacci". The code area contains the provided Java code. The "Output Window" section shows the result of running the code with input N = 34, which outputs "Yes". The "Compilation Results" section shows "Problem Solved Successfully" with 105/105 test cases passed. The accuracy is 66%.

```
// User function Template for Java
class Solution {
    static String checkFibonacci(int n) {
        // code here
        if(n==0||n==1) return "Yes";
        int a=0,b=1,c=0;
        while (c<=n) {
            c=a+b;
            if(c==n) return "Yes";
            a=b;
            b=c;
        }
        return "No";
    }
}
```

3.4 Nth Even Fibonacci Number

Aim : Given a positive integer n, find the nth Even Fibonacci number.

```
class Solution {
    static int nthEvenFibonacci(int n) {
        // code here
        int a=0,b=1,c=0,count=1;
        while(count<=n){
            c=a+b;
            if(c%2==0) count++;
            a=b;
            b=c;
        }
        return c;
    }
}
```

Output :

The screenshot shows the GeeksforGeeks online judge interface for the "Nth Even Fibonacci Number" problem. The code editor contains the provided Java solution. The output window shows two test cases: Input: n = 1, Output: 2 and Input: n = 2, Output: 8. Both cases pass. The compilation results show 15/15 test cases passed, with an accuracy of 100%. The page also displays difficulty, accuracy, and submission statistics.

```
// User function Template for Java
class Solution {
    static int nthEvenFibonacci(int n) {
        // code here
        int a=0,b=1,c=0,count=1;
        while(count<=n){
            c=a+b;
            if(c%2==0) count++;
            a=b;
            b=c;
        }
        return c;
    }
}
```

4. Programs on Arrays:

4.1 Missing in Array

Aim : You are given an array arr[] of size n - 1 that contains distinct integers in the range from 1 to n (inclusive). This array represents a permutation of the integers from 1 to n with one element missing. Your task is to identify and return the missing element.

Code :

```
class Solution {
    int missingNum(int arr[]) {
        // code here
        Arrays.sort(arr);
        for(int i=0;i<arr.length;i++){
            if(arr[i]!=i+1) return i+1;
        }
        return arr.length+1;
    }
}
```

Output :

The screenshot shows the GeeksforGeeks online judge interface for the 'Missing in Array' problem. The problem details state: "You are given an array arr[] of size n - 1 that contains distinct integers in the range from 1 to n (inclusive). This array represents a permutation of the integers from 1 to n with one element missing. Your task is to identify and return the missing element." The Java code provided is identical to the one shown above. In the 'Compilation Results' section, it shows "Problem Solved Successfully" with 1115 / 1115 test cases passed and 5 / 10 attempts correct, with 50% accuracy. The 'Output Window' shows the input [1, 2, 3, 5] and output 4, explaining that all numbers from 1 to 5 are present except 4.

4.2 Min and Max in Array

Aim : Given an array arr[]. Your task is to find the minimum and maximum elements in the array

Code :

```
class Solution {
    public ArrayList<Integer> getMinMax(int[] arr) {
        int min = Integer.MAX_VALUE;
        int max = Integer.MIN_VALUE;
        ArrayList<Integer> list = new ArrayList<>();
        for(int num: arr){
            if(num > max){
                max = num;
            }
            if(num < min){
                min = num;
            }
        }
        list.add(min);
        list.add(max);
        return list;
    }
}
```

Output :

The screenshot shows a browser window with the GeeksforGeeks URL: [geeksforgeeks.org/problems/find-minimum-and-maximum-element-in-an-array4428/1?page=1&category=Arrays&status=solved&sortBy=difficulty](https://www.geeksforgeeks.org/problems/find-minimum-and-maximum-element-in-an-array4428/1?page=1&category=Arrays&status=solved&sortBy=difficulty). The page displays the 'Min and Max in Array' problem details and a code editor with the provided Java solution. The code editor shows the Java code with line numbers. Below the code editor is the 'Output Window' which shows the problem solved successfully with 1111/1111 test cases passed and 2/4 attempts correct. The accuracy is 50%.

4.3 Palindromic Array

Aim : Given an array arr[] of positive integers. Return true if all the array elements are palindrome otherwise, return false.

```
class Solution {
    public static boolean isPalinArray(int[] arr) {
        // add code here.
        int r;
        for(int i=0;i<arr.length;i++){
            int n=arr[i],sum=0;
            int temp=n;
            while(n!=0){
                r=n%10;
                sum=sum*10+r;
                n=n/10;
            }
            if(temp==sum)
                continue;
            else return false;
        }
        return true;
    }
}
```

Output :

The screenshot shows the GeeksforGeeks online judge interface for the "Palindromic Array" problem. The code editor contains the provided Java solution. The output window shows the code runs successfully. The compilation results show 1115/1115 test cases passed with an accuracy of 100% and 2/2 attempts correct. The page also displays difficulty, accuracy, and submission statistics.

```
/*
 *Complete the Function below*/
class Solution {
    public static boolean isPalinArray(int[] arr) {
        // add code here.
        int r;
        for(int i=0;i<arr.length;i++){
            int n=arr[i],sum=0;
            int temp=n;
            while(n!=0){
                r=n%10;
                sum=sum*10+r;
                n=n/10;
            }
            if(temp==sum)
                continue;
            else return false;
        }
        return true;
    }
}
```

4.4 Union of 2 Sorted Arrays

Aim : Given two sorted arrays $a[]$ and $b[]$, where each array may contain duplicate elements , the task is to return the elements in the union of the two arrays in sorted order.

Union of two arrays can be defined as the set containing distinct common elements that are present in either of the arrays.

```
class Solution {
    public static ArrayList<Integer> findUnion(int a[], int b[]) {
        // code here
        TreeSet<Integer> ts=new TreeSet<>();
        for(int i:a) ts.add(i);
        for(int i:b) ts.add(i);
        return new ArrayList<>(ts);
    }
}
```

Output :

The screenshot shows the GeeksforGeeks platform interface. The user is solving a problem titled "Union of 2 Sorted Arrays". The problem details state: "Given two sorted arrays $a[]$ and $b[]$, where each array may contain duplicate elements , the task is to return the elements in the union of the two arrays in sorted order. Union of two arrays can be defined as the set containing distinct common elements that are present in either of the arrays." Below this, there are examples and a code editor.

Code Editor:

```
1- class Solution {
2-     public static ArrayList<Integer> findUnion(int a[], int b[]) {
3-         // code here
4-         TreeSet<Integer> ts=new TreeSet<>();
5-         for(int i:a) ts.add(i);
6-         for(int i:b) ts.add(i);
7-         return new ArrayList<>(ts);
8-     }
9- }
```

Output Window:

Compilation Results: Custom Input Y.O.G.I. (AI Bot)

Problem Solved Successfully

Test Cases Passed: 1115 / 1115

Attempts : Correct / Total: 2 / 2

Accuracy: 100%

Time Taken: [unspecified]

Custom Input Compile & Run Submit

4.5 Majority Element

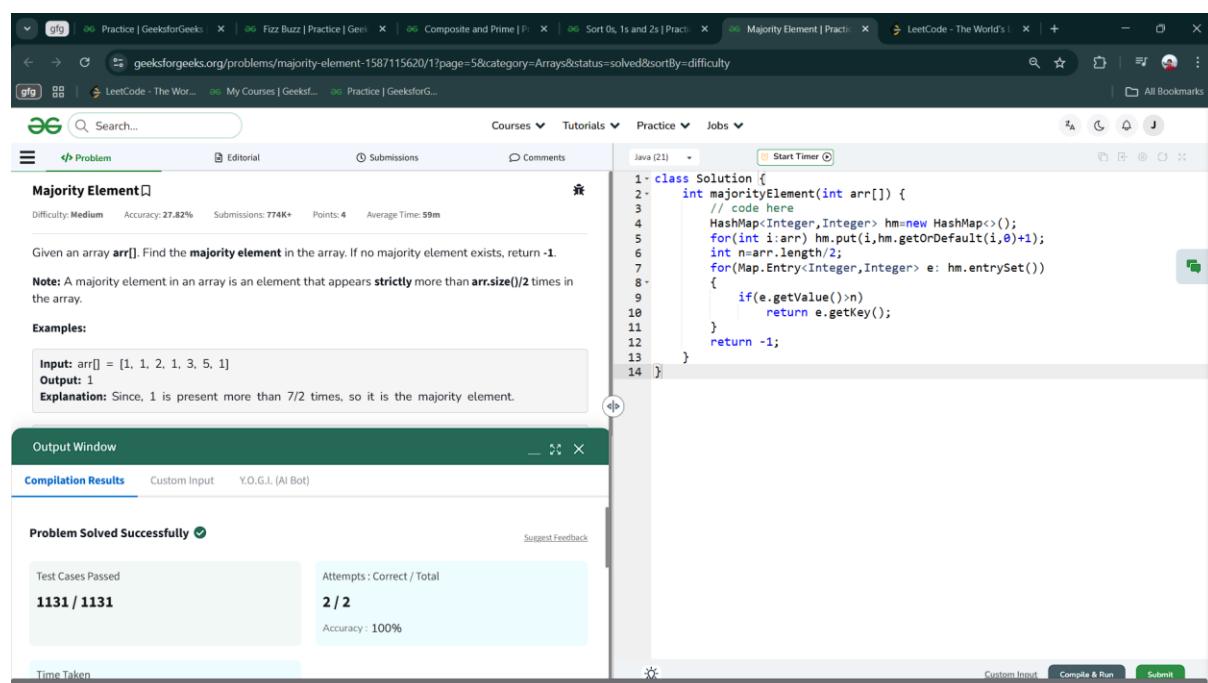
Aim : Given an array arr[]. Find the majority element in the array. If no majority element exists, return -1.

Note: A majority element in an array is an element that appears strictly more than arr.size()/2 times in the array.

```
class Solution {
```

```
    int majorityElement(int arr[]) {
        // code here
        HashMap<Integer, Integer> hm = new HashMap<>();
        for (int i : arr) hm.put(i, hm.getOrDefault(i, 0) + 1);
        int n = arr.length / 2;
        for (Map.Entry<Integer, Integer> e : hm.entrySet())
        {
            if (e.getValue() > n)
                return e.getKey();
        }
        return -1;
    }
}
```

Output :



The screenshot shows the GeeksforGeeks Online Judge interface for the 'Majority Element' problem. The code area contains the provided Java solution. The output window shows 'Compilation Results' with 'Test Cases Passed' as 1131/1131 and 'Attempts : Correct / Total' as 2/2, indicating 100% accuracy. A green checkmark icon is present in the results section.

```
1- class Solution {
2-     int majorityElement(int arr[]) {
3-         // code here
4-         HashMap<Integer, Integer> hm = new HashMap<>();
5-         for (int i : arr) hm.put(i, hm.getOrDefault(i, 0) + 1);
6-         int n = arr.length / 2;
7-         for (Map.Entry<Integer, Integer> e : hm.entrySet())
8-         {
9-             if (e.getValue() > n)
10-                 return e.getKey();
11-         }
12-         return -1;
13-     }
14- }
```

4.6 Numbers containing 1, 2 and 3

Aim : You are given an array arr[] of integers. Find all the numbers in the array whose digits consist only of [1, 2, 3]. Return an array containing only those numbers from arr[]. The order of the numbers in the output array should be the same as their order in the input array. If there is no such element in arr[], Return [-1].

```
class Solution {
    public ArrayList<Integer> filterByDigits(int[] arr) {
        ArrayList<Integer> al=new ArrayList<>();
        for(int i=0;i<arr.length;i++){
            int n=arr[i];
            if(n==1||n==2||n==3){al.add(n);
            continue;
            }
            if(n>9){
                if(findddigit(n)) al.add(n);
            }
        }
        return al;
    }
    static boolean findddigit(int n){
        while(n>0){
            if(n%10!=1&&n%10!=2&&n%10!=3) return false;
            else n/=10;
        }
        return true;
    }
}
```

}Output :

The screenshot shows a browser window with the GeeksforGeeks website open. The URL is geeksforgeeks.org/problems/numbers-containing-1-2-and-32555/1?page=3&category=Arrays&status=solved&sortBy=difficulty. The page displays the problem statement, examples, and a code editor with the provided Java solution. The code editor shows the Java code with line numbers. Below the code editor is an 'Output Window' showing 'Compilation Results' and 'Problem Solved Successfully'. The status bar at the bottom indicates 'Time Taken'.

```
1- class Solution {
2-     public ArrayList<Integer> filterByDigits(int[] arr) {
3-         // code here
4-         ArrayList<Integer> al=new ArrayList<>();
5-         for(int i=0;i<arr.length;i++){
6-             int n=arr[i];
7-             if(n==1||n==2||n==3){al.add(n);
8-             continue;
9-             }
10-            if(n>9){
11-                if(findddigit(n)) al.add(n);
12-            }
13-        }
14-        return al;
15-    }
16-    static boolean findddigit(int n){
17-        while(n>0){
18-            if(n%10!=1&&n%10!=2&&n%10!=3) return false;
19-            else n/=10;
20-        }
21-        return true;
22-    }
23- }
```

4.7 Composite and Prime

Aim : Given two integers l and r find the absolute difference between the number of composites and the number of primes between the range l and r (both inclusive)

class Solution {

```
public int Count(int l, int r) {
    ArrayList<Integer> prl = new ArrayList<>();
    ArrayList<Integer> cmpl = new ArrayList<>();
    for(int i=l;i<=r;i++){
        if(isprime(i))
            prl.add(i);
        else
            cmpl.add(i);
    }
    return Math.abs(prl.size()-cmpl.size());
}

static boolean isprime(int n){
    if(n<=1) return false;
    if(n==2||n==3) return true;
    if(n%2==0||n%3==0) return false;
    for(int i=5;i*i<=n;i+=6){
        if(n%i==0||n%(i+2)==0)
            return false;
    }
    return true;
}
```

Output :

4.8 Fizz Buzz

The screenshot shows a browser window with the GeeksforGeeks website open. The URL is [geeksforgeeks.org/problems/composite-and-prime-03591?page=6&category=Arrays&status=solved&sortBy=difficulty](https://www.geeksforgeeks.org/problems/composite-and-prime-03591?page=6&category=Arrays&status=solved&sortBy=difficulty). The page displays a Java problem titled "Composite and Prime". The code editor contains the Java code provided above. The output window shows the program solved successfully with 1115/1115 test cases passed and 3/4 attempts correct. The accuracy is 75%.

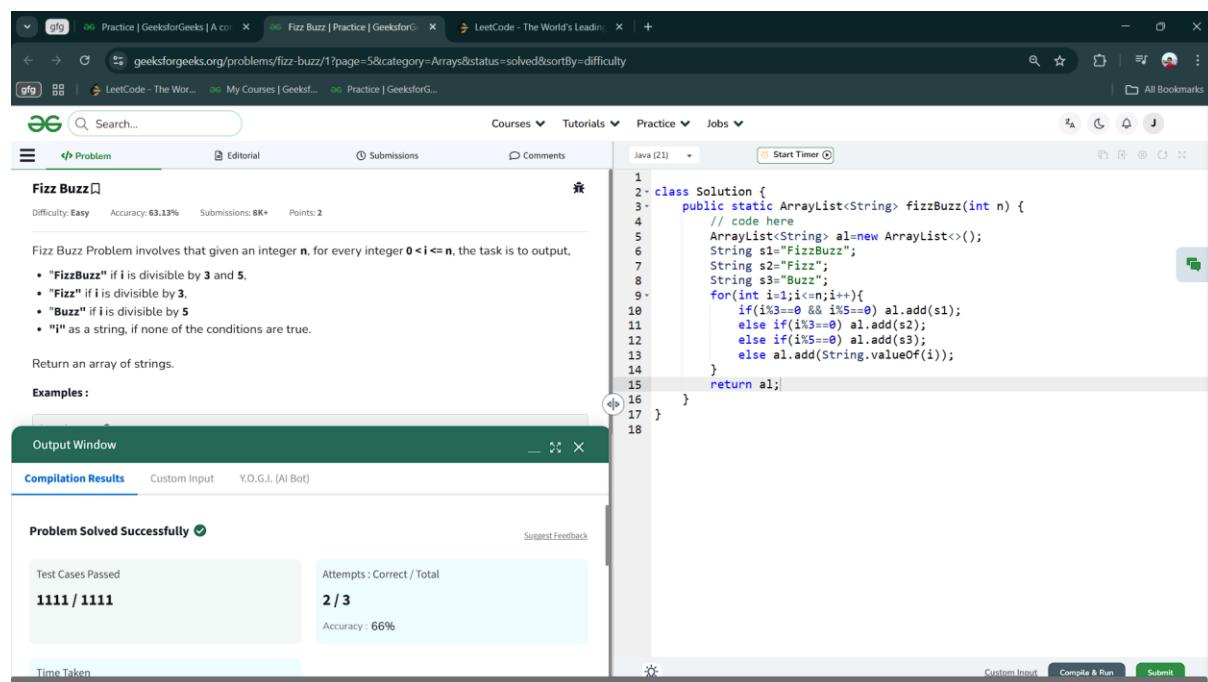
```
1 // User function Template for Java
2
3 - class Solution {
4 -     public int Count(int l, int r) {
5 -         // code here
6 -         ArrayList<Integer> prl = new ArrayList<>();
7 -         ArrayList<Integer> cmpl = new ArrayList<>();
8 -         for(int i=l;i<=r;i++){
9 -             if(isprime(i))
10 -                 prl.add(i);
11 -             else
12 -                 cmpl.add(i);
13 -         }
14 -         return Math.abs(prl.size()-cmpl.size());
15 -     }
16 -     static boolean isprime(int n){
17 -         if(n<1) return false;
18 -         if(n==2||n==3) return true;
19 -         if(n%2==0||n%3==0) return false;
20 -         for(int i=5;i*i<=n;i+=6){
21 -             if(n%i==0||n%(i+2)==0)
22 -                 return false;
23 -         }
24 -         return true;
25 -     }
}
```

Aim : Fizz Buzz Problem involves that given an integer n , for every integer $0 < i \leq n$, the task is to output,

- "FizzBuzz" if i is divisible by 3 and 5,
- "Fizz" if i is divisible by 3,
- "Buzz" if i is divisible by 5
- " i " as a string, if none of the conditions are true.

```
class Solution {
    public static ArrayList<String> fizzBuzz(int n) {
        // code here
        ArrayList<String> al=new ArrayList<>();
        String s1="FizzBuzz";
        String s2="Fizz";
        String s3="Buzz";
        for(int i=1;i<=n;i++){
            if(i%3==0 && i%5==0) al.add(s1);
            else if(i%3==0) al.add(s2);
            else if(i%5==0) al.add(s3);
            else al.add(String.valueOf(i));
        }
        return al;
    }
}
```

Output :



The screenshot shows a browser window with the GeeksforGeeks website open. The URL is geeksforgeeks.org/problems/fizz-buzz/1?page=5&category=Arrays&status=solved&sortBy=difficulty. The page displays the Fizz Buzz problem statement and a Java code editor. The code editor contains the provided Java solution for the Fizz Buzz problem. Below the code editor is an 'Output Window' showing the results of the submission. The results indicate that all test cases passed, with 1111/1111 correct and 2/3 attempts made, resulting in 66% accuracy. A green success message 'Problem Solved Successfully' is visible.

```
1 class Solution {
2     public static ArrayList<String> fizzBuzz(int n) {
3         // code here
4         ArrayList<String> al=new ArrayList<>();
5         String s1="FizzBuzz";
6         String s2="Fizz";
7         String s3="Buzz";
8         for(int i=1;i<=n;i++){
9             if(i%3==0 && i%5==0) al.add(s1);
10            else if(i%3==0) al.add(s2);
11            else if(i%5==0) al.add(s3);
12            else al.add(String.valueOf(i));
13        }
14        return al;
15    }
16 }
```

4.9 Max Consecutive ones

Aim : Given a binary array `nums`, return *the maximum number of consecutive 1's in the array.*

```
class Solution {
    public int findMaxConsecutiveOnes(int[] nums) {
        int res = 0;
        int count = 0;
        for (int n : nums) {
            if (n == 0) {
                count = 0;
            } else {
                count++;
            }
            if (res < count) {
                res = count;
            }
        }
        return res;
    }
}
```

Output :

The screenshot shows a LeetCode problem page for "485. Max Consecutive Ones". The code editor contains the provided Java solution. The "Test Result" section indicates the code was accepted with a runtime of 0 ms. It shows two test cases: Case 1 with input [1,1,0,1,1,1] and output 3, and Case 2 with input [1,0,1,1,0,1] and output 2.

4.10 Binary Search

Aim : Given an array of integers nums which is sorted in ascending order, and an integer target, write a function to search target in nums. If target exists, then return its index. Otherwise, return -1.

```
class Solution {
    public int search(int[] nums, int target) {
        int low=0,high=nums.length-1;
        while(low<=high){
            int mid=(low+high)/2;
            if(nums[mid]==target) return mid;
            else if(nums[mid]>target) high=mid-1;
            else low=mid+1;
        }
        return -1;
    }
}
```

Output :

The screenshot shows a browser window with the URL leetcode.com/problems/binary-search/. The page displays the 'Binary Search' problem from the 'Array' category. The code editor contains the provided Java solution. The test result indicates the code was accepted with a runtime of 0 ms. The input for the test case is `nums = [-1,0,3,5,9,12]` and `target = 9`.

```
1 class Solution {
2     public int search(int[] nums, int target) {
3         int low=0,high=nums.length-1;
4         while(low<=high){
5             int mid=(low+high)/2;
6             if(nums[mid]==target) return mid;
7             else if(nums[mid]>target) high=mid-1;
8             else low=mid+1;
9         }
10        return -1;
11    }
12 }
```

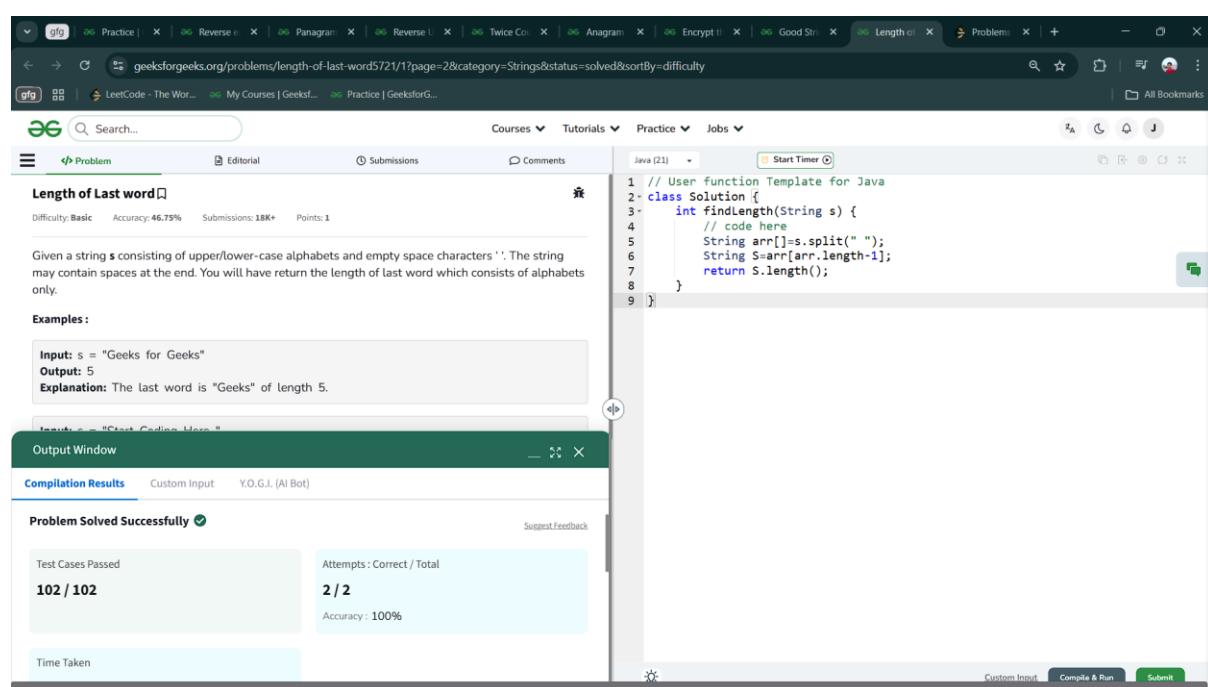
5 . Programs on Strings :

5.1 Length of Last word

Aim : Given a string s consisting of upper/lower-case alphabets and empty space characters ‘ ‘. The string may contain spaces at the end. You will have return the length of last word which consists of alphabets only.

```
class Solution {
    int findLength(String s) {
        // code here
        String arr[] = s.split(" ");
        String S = arr[arr.length - 1];
        return S.length();
    }
}
```

Output :



The screenshot shows the GeeksforGeeks online judge interface for the 'Length of Last word' problem. The problem details are as follows:

- Difficulty:** Basic, **Accuracy:** 46.75%, **Submissions:** 18K+, **Points:** 1
- Description:** Given a string s consisting of upper/lower-case alphabets and empty space characters ' '. The string may contain spaces at the end. You will have return the length of last word which consists of alphabets only.
- Examples:**
 - Input:** s = "Geeks for Geeks"
Output: 5
Explanation: The last word is "Geeks" of length 5.
 - Input:** s = "Code Geeks Here"
Output: 5

The code editor window shows the Java code provided above. The output window shows the test case "Code Geeks Here" and the output "5". The compilation results show "Problem Solved Successfully" with 102/102 test cases passed and 2/2 attempts correct, resulting in 100% accuracy. The time taken is not explicitly shown.

5.2 Good String

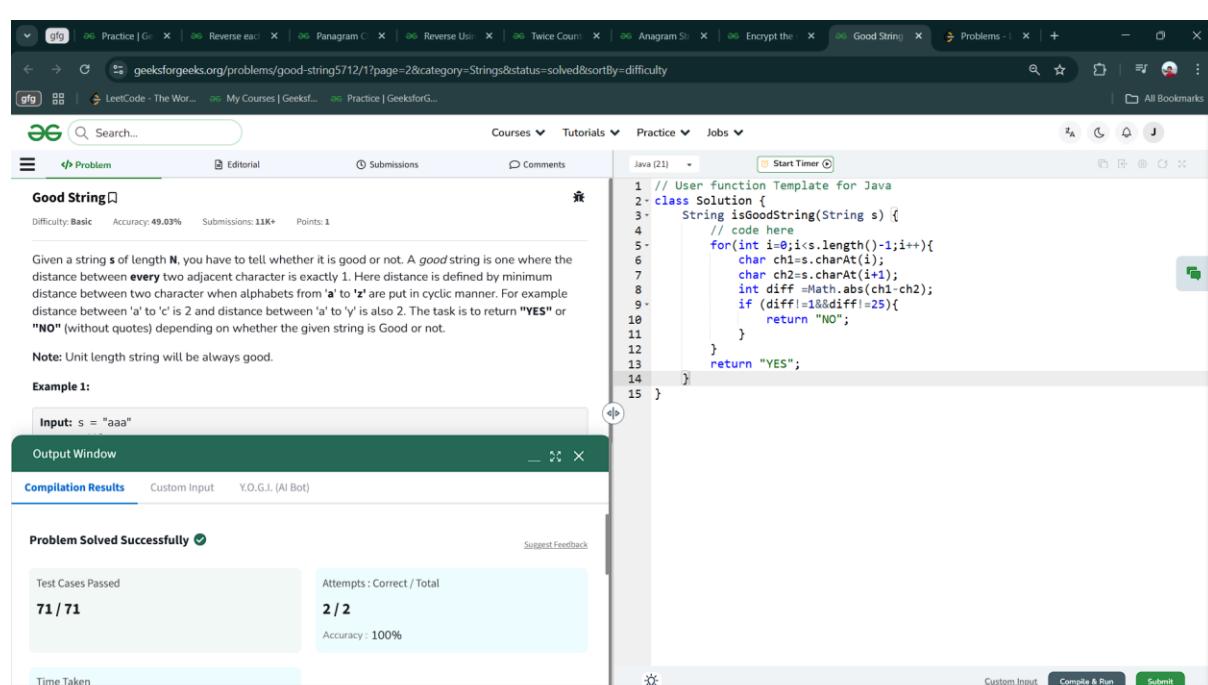
Aim : Given a string s of length N, you have to tell whether it is good or not.
A good string is one where the distance between every two adjacent character is exactly 1. Here distance is defined by minimum distance between two character when alphabets from 'a' to 'z' are put in cyclic manner. For example distance between 'a' to 'c' is 2 and distance between 'a' to 'y' is also 2. The task is to return "YES" or "NO" (without quotes) depending on whether the given string is Good or not.

Note: Unit length string will be always good.

```
class Solution {
```

```
    String isGoodString(String s) {
        // code here
        for(int i=0;i<s.length()-1;i++){
            char ch1=s.charAt(i);
            char ch2=s.charAt(i+1);
            int diff =Math.abs(ch1-ch2);
            if (diff!=1&&diff!=25){
                return "NO";
            }
        }
        return "YES";
    }
}
```

Output :



The screenshot shows a browser window for GeeksforGeeks with the URL [geeksforgeeks.org/problems/good-string5712/1?page=2&category=Strings&status=solved&sortBy=difficulty](https://www.geeksforgeeks.org/problems/good-string5712/1?page=2&category=Strings&status=solved&sortBy=difficulty). The page displays the 'Good String' problem details and a Java code editor. The code is identical to the one provided above. In the code editor, line 1 has a comment // User function Template for Java. The code is submitted and passed all test cases successfully, with an accuracy of 100% and 2/2 attempts correct. The output window shows the input "aaa" and the output "YES".

5.3 Encrypt the string - 1

Aim : Bingu was testing all the strings he had at his place and found that most of them were prone to a vicious attack by Banju, his arch-enemy. Bingu decided to encrypt all the strings he had, by the following method. Every substring of identical letters is replaced by a single instance of that letter followed by the number of occurrences of that letter. Then, the string thus obtained is further encrypted by reversing it.

class Solution {

```
String encryptString(String s) {
    // code here
    StringBuilder sb = new StringBuilder();
    int c=1;
    for(int i=1;i<s.length();i++){
        if(s.charAt(i-1)!=s.charAt(i)){
            sb.append(s.charAt(i-1));
            sb.append(c);
            c=1;
        }else c++;
    }
    sb.append(s.charAt(s.length()-1));
    sb.append(c);
    return sb.reverse().toString();
}
```

Output :

The screenshot shows the GeeksforGeeks online judge interface. The user has selected the Java tab for the problem. The code area contains the provided Java code for the Solution class. The output window shows the result of running the code with the input "aab", which outputs "1a2b1c". The compilation results section indicates that the problem was solved successfully, with 38/38 test cases passed and 2/2 attempts correct, achieving 100% accuracy.

5.4 Anagram Strings

Aim : Given two strings S1 and S2 . Return "1" if both strings are anagrams otherwise return "0" .

Note: An anagram of a string is another string with exactly the same quantity of each character in it, in any order.

class Solution {

```
static int areAnagram(String S1, String S2) {
    // code here
    if (S1.length() != S2.length()) return 0;
    HashMap<Character, Integer> hm = new HashMap<>();

    for (int i=0;i<S1.length();i++)
        hm.put(S1.charAt(i), hm.getOrDefault(S1.charAt(i), 0) + 1);

    for (int i=0;i<S1.length();i++) {
        if (!hm.containsKey(S2.charAt(i))) return 0;
        hm.put(S2.charAt(i), hm.get(S2.charAt(i)) - 1);
        if (hm.get(S2.charAt(i))==0)
            hm.remove(S2.charAt(i));
    }
    return hm.isEmpty()?1:0;
}
```

Output :

The screenshot shows a browser window with multiple tabs open, including 'geeksforgeeks.org/problems/java-anagram-strings3549/1?page=4&category=Strings&status=solved&sortBy=difficulty'. The main content is a Java problem titled 'Anagram Strings'. It includes a problem statement, difficulty level (Basic), accuracy (49.66%), submissions (18K+), and points (1). Below the statement, there are examples and explanations for inputs like 'cdbkdub' and 'dsbkcsdn' (Output: 0) and 'geeks' and 'skgee' (Output: 1). A note states that S1 has the same quantity of each character in it as S2. The right side of the screen shows the Java code with line numbers 1 through 19. The code uses a HashMap to count characters in S1 and then checks if S2 has the same counts. It returns 1 if S2 is a permutation of S1 and 0 otherwise. At the bottom, there are sections for 'Your Task', 'Expected Time Complexity: O(n)', 'Expected Auxiliary Space: O(K), Where K=Constant', and buttons for 'Custom Input', 'Compile & Run', and 'Submit'.

```

1 // User function template for Java
2 class Solution {
3     static int areAnagram(String S1, String S2) {
4         // code here
5         if (S1.length() != S2.length()) return 0;
6         HashMap<Character, Integer> hm = new HashMap<>();
7
8         for (int i=0;i<S1.length();i++)
9             hm.put(S1.charAt(i), hm.getOrDefault(S1.charAt(i), 0) + 1);
10
11        for (int i=0;i<S1.length();i++) {
12            if (!hm.containsKey(S2.charAt(i))) return 0;
13            hm.put(S2.charAt(i), hm.get(S2.charAt(i)) - 1);
14            if (hm.get(S2.charAt(i))==0)
15                hm.remove(S2.charAt(i));
16        }
17        return hm.isEmpty()?1:0;
18    }
19 }
```

5.5 Twice Counter

Aim : Given a list of N words. Count the number of words that appear exactly twice in the list

```
class Solution {
```

```
    public int countWords(String list[], int n) {
        // code here
        int count=0;
        HashMap<String, Integer> hm=new HashMap<>();
        for(String s:list){
            hm.put(s,hm.getOrDefault(s,0)+1);
        }
        for(String s:hm.keySet()){
            if(hm.get(s)==2) count++;
        }
        return count;
    }
}
```

Output :

The screenshot shows a browser window with multiple tabs open, including 'geeksforgeeks.org/problems/twice-counter/4236/?page=5&category=Strings&status=solved&sortBy=difficulty'. The main content is the 'Twice Counter' problem page on GeeksforGeeks.

Problem Details:

- Difficulty: Easy
- Accuracy: 62.61%
- Submissions: 44K+
- Points: 2

Description:

Given a list of N words. Count the number of words that appear **exactly twice** in the list.

Example 1:

Input:
N = 3
list = {Geeks, For, Geeks}
Output: 1
Explanation: 'Geeks' is the only word that appears twice.

Code Snippet:

```
// User function Template for Java
class Solution {
    public int countWords(String list[], int n) {
        // code here
        int count=0;
        HashMap<String, Integer> hm=new HashMap<>();
        for(String s:list){
            hm.put(s,hm.getOrDefault(s,0)+1);
        }
        for(String s:hm.keySet()){
            if(hm.get(s)==2) count++;
        }
        return count;
    }
}
```

Output Window:

Compilation Results: Custom Input Y.O.G.I. (AI Bot)

Problem Solved Successfully ✓

Test Cases Passed: 215 / 215

Attempts : Correct / Total: 2 / 2

Accuracy: 100%

Time Taken: [unintelligible]

Custom Input Compile & Run Submit

5.6 Reverse Using Stack

Aim : You are given a string s , the task is to reverse the string using stack.

```
class Solution {
    public String reverse(String S) {
        // code here
        Stack<Character> st=new Stack<>();
        for(char c:S.toCharArray()){
            st.push(c);
        }
        StringBuilder s=new StringBuilder();
        while(st.size()!=0){
            s.append(st.peek());
            st.pop();
        }
        return s.toString();
    }
}
```

Output :

The screenshot shows the GeeksforGeeks Practice interface for the 'Reverse Using Stack' problem. The code editor displays the Java solution provided above. The output window shows 'Problem Solved Successfully' with 300/300 test cases passed and 2/2 attempts correct, achieving 100% accuracy. The bottom right corner of the screenshot has a small watermark or logo.

```
1- class Solution {
2-     public String reverse(String S) {
3-         // code here
4-         Stack<Character> st=new Stack<>();
5-         for(char c:S.toCharArray()){
6-             st.push(c);
7-         }
8-         StringBuilder s=new StringBuilder();
9-         while(st.size()!=0){
10-             s.append(st.peek());
11-             st.pop();
12-         }
13-         return s.toString();
14-     }
15- }
```

5.7 Panagram Checking

Aim : Given a string s, check if it is a "Panagram" or not. Return true if the string is a Panagram, else return false.

A "Panagram" is a sentence containing every letter in the English Alphabet either in lowercase or Uppercase.

class Solution {

```
public static boolean checkPanagram(String s) {
    // code here
    if(s.length()<26) return false;
    s = s.toLowerCase();
    int a[] = new int[26];
    for(int i=0;i<s.length();i++){
        char ch = s.charAt(i);
        if(ch<='z' && ch>='a') a[ch-'a']++;
    }
    for(int i=0;i<26;i++){
        if(a[i] == 0) return false;
    }
    return true;
}
```

Output :

The screenshot shows a browser window with the URL geeksforgeeks.org/problems/panagram-checking-1587115620/1?page=5&category=Strings&status=solved&sortBy=difficulty. The page displays a LeetCode problem titled "Panagram Checking". The code editor contains the Java solution provided above. The output window shows the results of the submission, indicating that all test cases passed (1115 / 1115), the attempts were correct (2 / 2), and the accuracy was 100%. A green success message "Problem Solved Successfully" is visible.

5.8 Reverse each word in a given string

Aim : You are given a string s. You need to reverse each word in it where the words are separated by spaces and return the modified string.

Note: The string may contain leading or trailing spaces, or multiple spaces between two words. The returned string should only have a single space separating the words, and no extra spaces should be included.

```
class Solution {
    public String reverseWords(String s) {
        // Code here
        s=s.trim();
        s=s.replaceAll("\\s+","");
        String a[]={s.split(" ")};
        StringBuilder sb=new StringBuilder();
        for(String str:a){
            sb.append(reverse(str.toCharArray())).append(" ");
        }
        return sb.toString().trim();
    }
    public String reverse(char ch[]){
        int i=0,j=ch.length-1;
        while(i<j){
            char c=ch[j];
            ch[j]=ch[i];
            ch[i]=c;
            i++;
            j--;
        }
        return String.valueOf(ch);
    }
}
```

Output :

The screenshot shows a Java code editor interface. The code is pasted into the editor, and the cursor is visible at the end of the second line of the reverseWords() method. The editor has tabs for 'Editorial' and 'Comments'. Below the code, there's a 'Start Timer' button. The code editor is part of a larger application window with a navigation bar at the top.

5.9 Valid Parentheses

Aim : Given a string s containing just the characters '(', ')', '{', '}', '[' and ']', determine if the input string is valid.

An input string is valid if:

1. Open brackets must be closed by the same type of brackets.
2. Open brackets must be closed in the correct order.
3. Every close bracket has a corresponding open bracket of the same type

class Solution {

```
public boolean isValid(String s) {
    Stack<Character> stack = new Stack<>();
    for (char ch : s.toCharArray()) {
        if (ch == '(' || ch == '[' || ch == '{') {
            stack.push(ch);
        } else {
            if (stack.isEmpty()) {
                return false;
            }
            char top = stack.pop();
            if (ch == ')' && top != '(') {
                return false;
            }
            if (ch == ']' && top != '[') {
                return false;
            }
            if (ch == '}' && top != '{') {
                return false;
            }
        }
    }
    return stack.isEmpty();
}
```

}Output :

The screenshot shows a browser window with the LeetCode URL: leetcode.com/problems/valid-parentheses/?envType=problem-list-v2&envId=string. The page displays the problem statement for 'Valid Parentheses'. The code editor contains the Java solution provided above. The 'Test Result' section indicates that all test cases (Case 1, Case 2, Case 3, Case 4, Case 5) have been accepted with a runtime of 0 ms. The bottom of the screen shows the LeetCode navigation bar with options like 'Home', 'Search', 'Premium', and 'Logout'.

5.10 Jewels and stones

Aim : You're given strings jewels representing the types of stones that are jewels, and stones representing the stones you have. Each character in stones is a type of stone you have. You want to know how many of the stones you have are also jewels.

Letters are case sensitive, so "a" is considered a different type of stone from "A".

```
class Solution {
    public int numJewelsInStones(String jewels, String stones) {
        int res = 0;
        HashSet<Character> hs= new HashSet<>();
        for (char i:jewels.toCharArray())
            hs.add(i);
        for (char s: stones.toCharArray())
            if (hs.contains(s)) res++;
        return res;
    }
}
```

Output :

The screenshot shows a LeetCode problem page for '771. Jewels and Stones'. The code editor contains the provided Java solution. Below the code, the 'Test Result' section shows the code was accepted with a runtime of 0 ms. It includes two test cases, both of which passed. The input for the first test case is 'jewels = "aA"' and 'stones = "aAaaa"'. The interface also shows some navigation links like 'Topics' and 'Companies'.

```
class Solution {
    public int numJewelsInStones(String jewels, String stones) {
        int res = 0;
        HashSet<Character> hs= new HashSet<>();
        for (char i:jewels.toCharArray())
            hs.add(i);
        for (char s: stones.toCharArray())
            if (hs.contains(s)) res++;
        return res;
    }
}
```

6. Programs on Matrices :

6.1 Find difference between sum of diagonals

Aim : Given a matrix Grid[][] of size NxN. Calculate the absolute difference between the sums of its diagonals.

```
class Solution {
    int diagonalSumDifference(int N, int[][] Grid) {
        // code here
        int diff=0;
        for(int i=0;i<Grid.length;i++){
            diff+=Grid[i][i];
            diff-=Grid[i][Grid.length-i-1];
        }
        return Math.abs(diff);
    }
}
```

Output :

The screenshot shows a browser window with the GeeksforGeeks website open. The URL in the address bar is geeksforgeeks.org/problems/find-difference-between-sum-of-diagonals1554/1?page=1&category=Matrix&status=solved&sortBy=difficulty. The page displays a Java problem titled "Find difference between sum of diagonals". The code area contains the provided Java code. Below the code, the "Output Window" shows "Compilation Results" with "Test Cases Passed: 12 / 12" and "Attempts: Correct / Total: 2 / 2", indicating 100% accuracy. A green checkmark icon next to "Problem Solved Successfully" is visible.

```
1 // User function Template for Java
2
3 class Solution {
4     int diagonalSumDifference(int N, int[][] Grid) {
5         // code here
6         int diff=0;
7         for(int i=0;i<Grid.length;i++){
8             diff+=Grid[i][i];
9             diff-=Grid[i][Grid.length-i-1];
10        }
11        return Math.abs(diff);
12    }
13 }
```

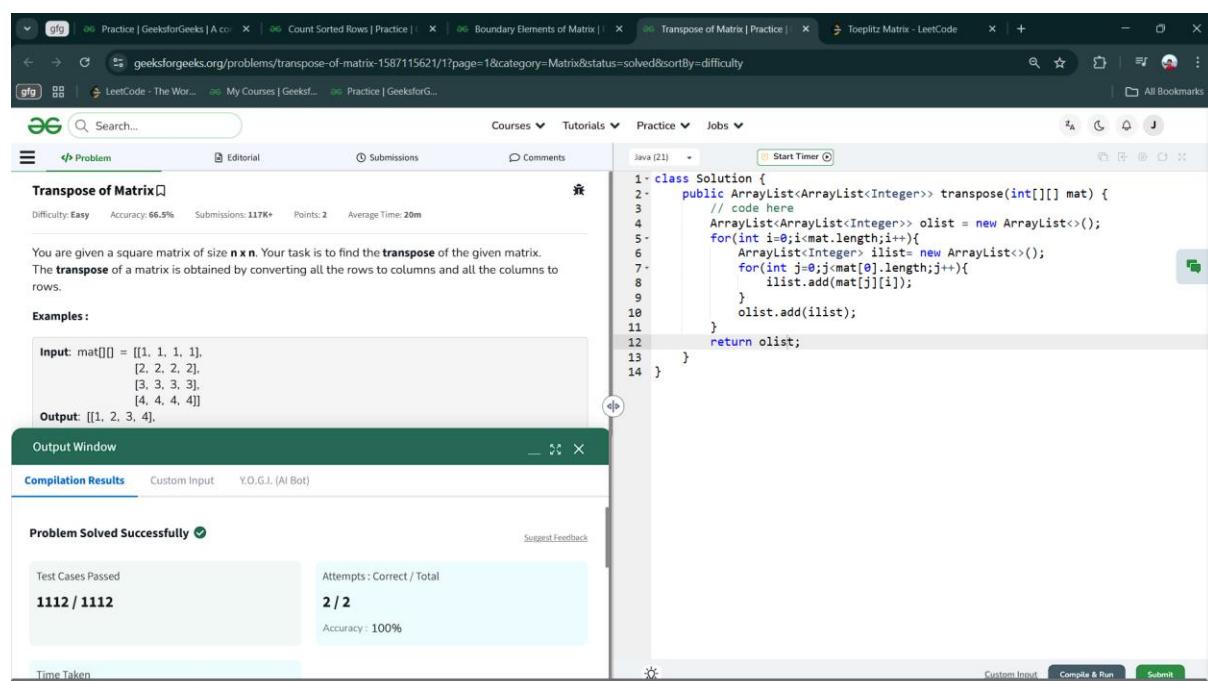
6.2 Transpose of Matrix

Aim : You are given a square matrix of size $n \times n$. Your task is to find the transpose of the given matrix.

The transpose of a matrix is obtained by converting all the rows to columns and all the columns to rows.

```
class Solution {
    public ArrayList<ArrayList<Integer>> transpose(int[][] mat) {
        ArrayList<ArrayList<Integer>> olist = new ArrayList<>();
        for(int i=0;i<mat.length;i++){
            ArrayList<Integer> ilist= new ArrayList<>();
            for(int j=0;j<mat[0].length;j++){
                ilist.add(mat[j][i]);
            }
            olist.add(ilist);
        }
        return olist;
    }
}
```

Output :



The screenshot shows a browser window with the GeeksforGeeks website open. The URL is [geeksforgeeks.org/problems/transpose-of-matrix-1587115621/1?page=1&category=Matrix&status=solved&sortBy=difficulty](https://www.geeksforgeeks.org/problems/transpose-of-matrix-1587115621/1?page=1&category=Matrix&status=solved&sortBy=difficulty). The page displays the 'Transpose of Matrix' problem. The code submitted is identical to the one shown above. In the 'Compilation Results' section, it shows 'Problem Solved Successfully' with 1112 / 1112 test cases passed, 2 / 2 attempts correct, and 100% accuracy. The time taken is listed as 0 ms.

```
1- class Solution {
2-     public ArrayList<ArrayList<Integer>> transpose(int[][] mat) {
3-         // code here
4-         ArrayList<ArrayList<Integer>> olist = new ArrayList<>();
5-         for(int i=0;i<mat.length;i++){
6-             ArrayList<Integer> ilist= new ArrayList<>();
7-             for(int j=0;j<mat[0].length;j++){
8-                 ilist.add(mat[j][i]);
9-             }
10-            olist.add(ilist);
11-        }
12-        return olist;
13-    }
14- }
```

6.3 Boundary Elements of Matrix

Aim : Given an $n \times n$ matrix .In the given matrix, you have to find the boundary elements of the matrix.

```
class Solution {
    public int[] BoundaryElements(int[][] matrix) {
        ArrayList<Integer> al=new ArrayList<>();
        int n=matrix.length,m=matrix[0].length;
        for(int i=0;i<n;i++){
            for(int j=0;j<m;j++){
                if(i==0||j==0||i==n-1||j==m-1){
                    al.add(matrix[i][j]);
                }
            }
        }
        int mat[]={};
        for(int i=0;i<al.size();i++){
            mat[i]=al.get(i);
        }
        return mat;
    }
}
```

Output :

The screenshot shows the GeeksforGeeks platform interface for solving a programming problem. The problem title is "Boundary Elements of Matrix". The code area contains the Java solution provided above. The output window shows the input matrix and the generated boundary array. The results section indicates the problem was solved successfully with 2/2 test cases passed and 100% accuracy. The bottom right corner has buttons for "Custom Input", "Compile & Run", and "Submit".

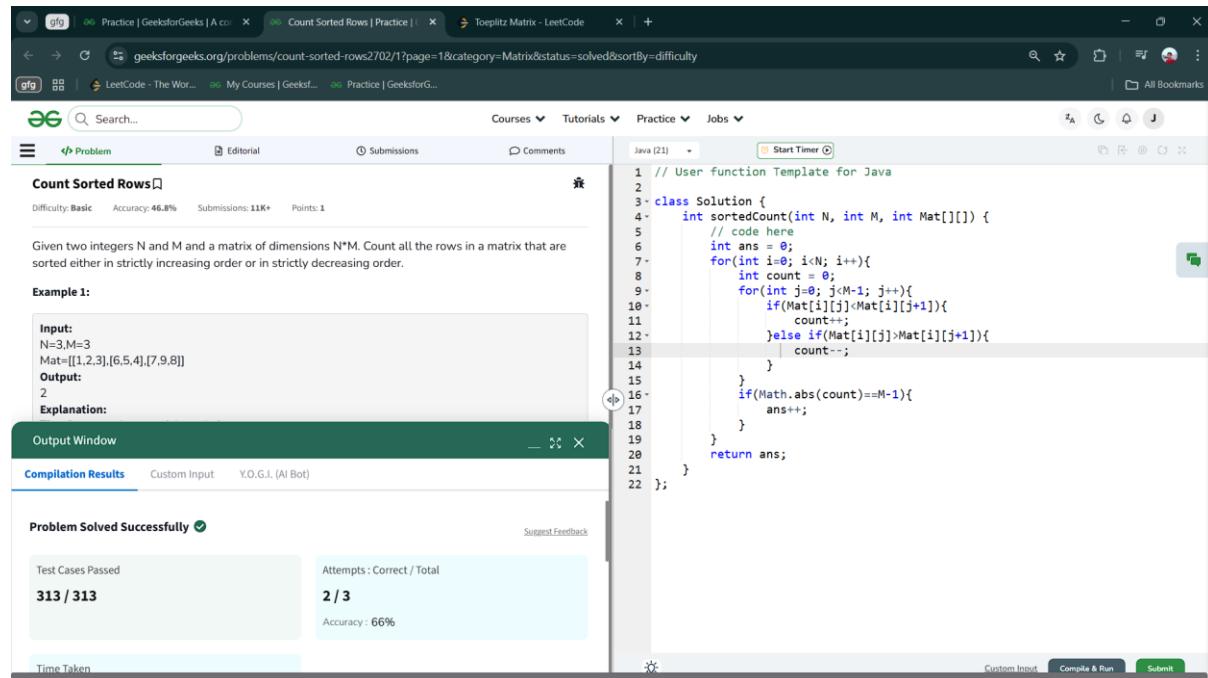
```
1 // User function Template for Java
2
3 class Solution {
4     public int[] BoundaryElements(int[][] matrix) {
5         // code here
6         ArrayList<Integer> al=new ArrayList<>();
7         int n=matrix.length,m=matrix[0].length;
8         for(int i=0;i<n;i++){
9             for(int j=0;j<m;j++){
10                 if(i==0||j==0||i==n-1||j==m-1){
11                     al.add(matrix[i][j]);
12                 }
13             }
14         }
15         int mat[]={};
16         for(int i=0;i<al.size();i++){
17             mat[i]=al.get(i);
18         }
19         return mat;
20     }
21 }
```

6.4 Count Sorted Rows

Aim : Given two integers N and M and a matrix of dimensions N*M. Count all the rows in a matrix that are sorted either in strictly increasing order or in strictly decreasing order.

```
class Solution {
    int sortedCount(int N, int M, int Mat[][]){
        // code here
        int ans = 0;
        for(int i=0; i<N; i++){
            int count = 0;
            for(int j=0; j<M-1; j++){
                if(Mat[i][j]<Mat[i][j+1]){
                    count++;
                }else if(Mat[i][j]>Mat[i][j+1]){
                    count--;
                }
            }
            if(Math.abs(count)==M-1){
                ans++;
            }
        }
        return ans;
    }
};
```

Output :



The screenshot shows the GeeksforGeeks Practice interface for the 'Count Sorted Rows' problem. The code editor contains the provided Java solution. The output window shows the code was compiled successfully. The results section indicates 313 test cases passed out of 313, with an accuracy of 66%.

```
1 // User function Template for Java
2
3 class Solution {
4     int sortedCount(int N, int M, int Mat[][]){
5         // code here
6         int ans = 0;
7         for(int i=0; i<N; i++){
8             int count = 0;
9             for(int j=0; j<M-1; j++){
10                 if(Mat[i][j]<Mat[i][j+1]){
11                     count++;
12                 }else if(Mat[i][j]>Mat[i][j+1]){
13                     count--;
14                 }
15             }
16             if(Math.abs(count)==M-1){
17                 ans++;
18             }
19         }
20         return ans;
21     }
22 }
```

6.5 Toeplitz Matrix

Aim : Given an $m \times n$ matrix, return *true* if the matrix is Toeplitz. Otherwise, return *false*.

A matrix is Toeplitz if every diagonal from top-left to bottom-right has the same elements.

```
class Solution {
    public boolean isToeplitzMatrix(int[][] mat) {
        int n=mat.length;
        int m=mat[0].length;
        for(int i=0;i<n-1;i++){
            for(int j=0;j<m-1;j++){
                if(mat[i][j]!=mat[i+1][j+1]){
                    return false;
                }
            }
        }
        return true;
    }
}
```

Output :

The screenshot shows a browser window with the LeetCode URL: leetcode.com/problems/toeplitz-matrix/. The page displays the problem statement and a Java code editor. The code editor contains the provided solution for checking if a matrix is Toeplitz. Below the code editor, there is a 'Testcase' section showing two accepted test cases: 'Case 1' and 'Case 2'. The input for Case 1 is a 3x4 matrix with values [1, 2, 3, 4], [5, 1, 2, 3], and [9, 5, 1, 2]. The output for Case 1 is 'true'. The input for Case 2 is a 3x3 matrix with values [1, 2, 3], [4, 1, 2], and [5, 4, 1]. The output for Case 2 is also 'true'.

```
1 class Solution {
2     public boolean isToeplitzMatrix(int[][] mat) {
3         int n=mat.length;
4         int m=mat[0].length;
5         for(int i=0;i<n-1;i++){
6             for(int j=0;j<m-1;j++){
7                 if(mat[i][j]!=mat[i+1][j+1]){
8                     return false;
9                 }
10            }
11        }
12        return true;
13    }
}
```

7. Programs on series and patterns :

7.1 Geek's Weight

Aim : Geek is getting really fat. He wants to lose his weight but can't get the motivation to workout. Seeing this, his friend Heisenberg offers him a deal.

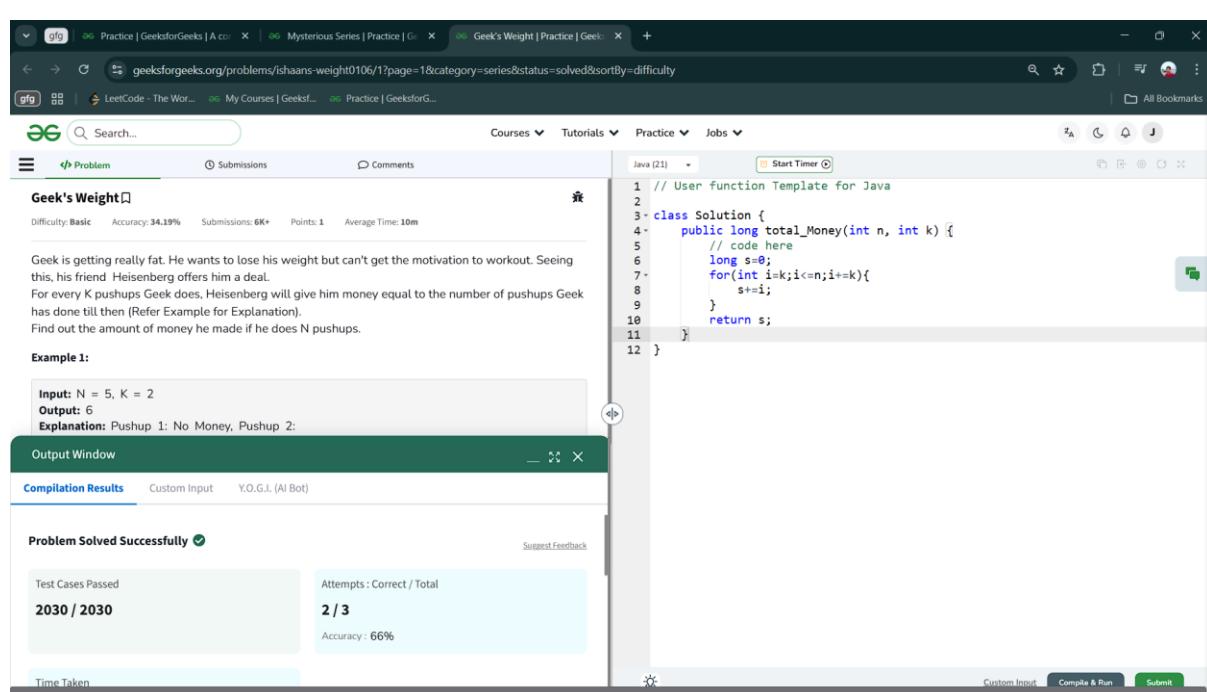
For every K pushups Geek does, Heisenberg will give him money equal to the number of pushups Geek has done till then (Refer Example for Explanation).

Find out the amount of money he made if he does N pushups.

```
class Solution {

    public long total_Money(int n, int k) {
        // code here
        long s=0;
        for(int i=k;i<=n;i+=k){
            s+=i;
        }
        return s;
    }
}
```

Output :



The screenshot shows the GeeksforGeeks online judge interface for the 'Geek's Weight' problem. The code editor contains the provided Java solution. The output window shows 'Problem Solved Successfully' with 2030/2030 test cases passed and an accuracy of 66%. The compilation results show 'Compilation Results' and 'Custom Input' tabs.

```
1 // User function Template for Java
2
3- class Solution {
4-     public long total_Money(int n, int k) {
5-         // code here
6-         long s=0;
7-         for(int i=k;i<=n;i+=k){
8-             s+=i;
9-         }
10-        return s;
11-    }
12 }
```

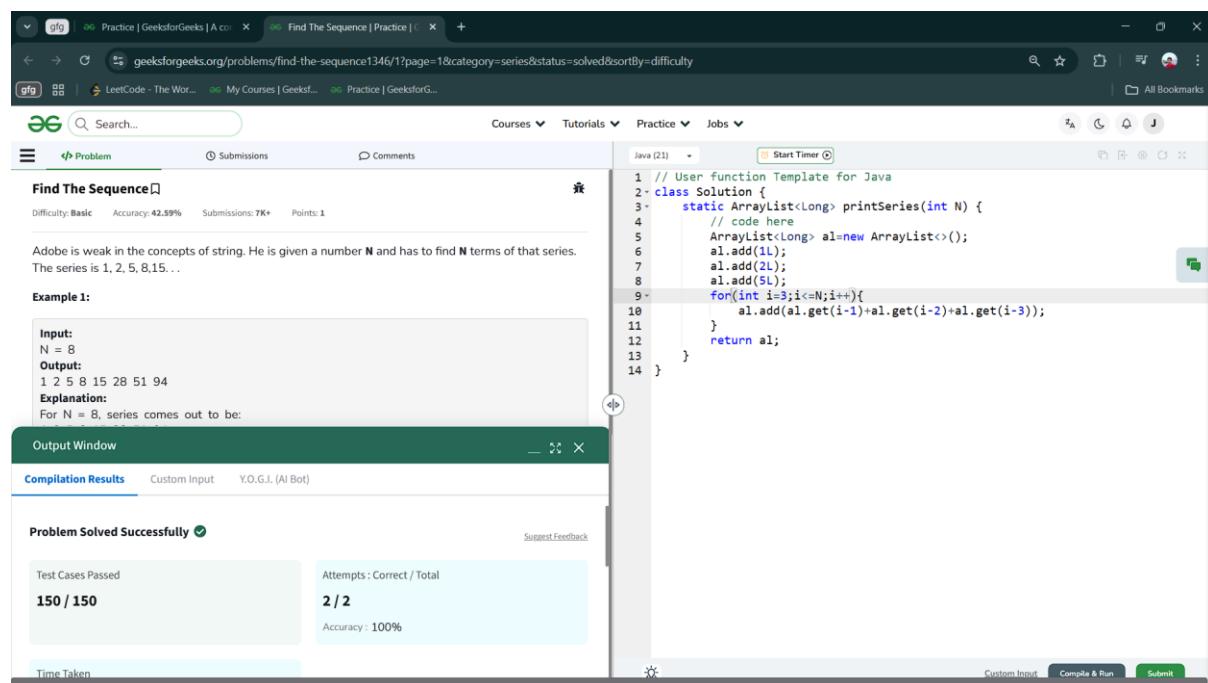
7.2 Find The Sequence

Aim : Adobe is weak in the concepts of string. He is given a number N and has to find N terms of that series. The series is 1, 2, 5, 8,15. .

```
class Solution {
```

```
    static ArrayList<Long> printSeries(int N) {
        ArrayList<Long> al=new ArrayList<>();
        al.add(1L);
        al.add(2L);
        al.add(5L);
        for(int i=3;i<=N;i++){
            al.add(al.get(i-1)+al.get(i-2)+al.get(i-3));
        }
        return al;
    }
}
```

Output :



The screenshot shows the GeeksforGeeks Practice interface for the 'Find The Sequence' problem. The code editor contains the provided Java solution. The output window shows 'Compilation Results' with '150 / 150' test cases passed and 'Accuracy: 100%'. The status bar at the bottom indicates 'Time Taken'.

```
1 // User function Template for Java
2 class Solution {
3     static ArrayList<Long> printSeries(int N) {
4         // code here
5         ArrayList<Long> al=new ArrayList<>();
6         al.add(1L);
7         al.add(2L);
8         al.add(5L);
9         for(int i=3;i<=N;i++){
10             al.add(al.get(i-1)+al.get(i-2)+al.get(i-3));
11         }
12     }
13     return al;
14 }
```

7.3 Pattern-1

Aim : Given a number N, identify the pattern from the given Examples and print the Output.

```
class Solution {
```

```
    static String[] findThePattern(int N) {
        char ch='A';
        String[] a=new String[N];
        for(int i=0;i<N;i++){
            StringBuilder sb=new StringBuilder();
            for(int j=0;j<N;j++){
                if(i==0||i==N-1||j==0||j==N-1) sb.append(ch++);
                else sb.append('$');
            }
            a[i]=sb.toString();
        }
        return a;
    };
}
```

Output :

The screenshot shows a browser window for GeeksforGeeks with the URL geeksforgeeks.org/problems/pattern-1-practice-geeksforgeeks-13116/1?page=1&category=pattern-printing&sortBy=difficulty. The page title is "Pattern-1 | Practice | GeeksforGeeks". The main content area displays a Java code editor with the provided solution for "Pattern-1". Below the code editor is an "Output Window" showing the result for N=4: ABCD E\$F C\$G. At the bottom, the "Compilation Results" section shows "Problem Solved Successfully" with 7/7 test cases passed and 2/2 attempts correct, resulting in 100% accuracy.

```
1- class Solution {
2-     static String[] findThePattern(int N) {
3-         // code here
4-         char ch='A';
5-         String[] a=new String[N];
6-         for(int i=0;i<N;i++){
7-             StringBuilder sb=new StringBuilder();
8-             for(int j=0;j<N;j++){
9-                 if(i==0||i==N-1||j==0||j==N-1) sb.append(ch++);
10-                else sb.append('$');
11-            }
12-            a[i]=sb.toString();
13-        }
14-        return a;
15-    }
16-};
```

7.4 Pattern 2

Aim : Geek is very fond of patterns. Once, his teacher gave him a pattern to solve. He gave Ram an integer n and asked him to build a pattern.
Help Ram build a pattern.

```
class Solution {

    void printTriangle(int n) {
        for(int i=1;i<=n;i++){
            for(int j=1;j<=n-i;j++){
                System.out.print(" ");
            }
            for(int j=1;j<=(2*i)-1;j++){
                System.out.print("*");
            }
            System.out.println();
        }
    }
}
```

Output :

7.5 Triangle growing downwards

Aim : Given string str of a constant length, print a triangle out of it. The triangle should start with the first character of the string and keep growing downwards by adding one character from the string.

The spaces on the left side of the triangle should be replaced with a dot character.

```
class Solution {

    static void printTriangleDownwards(String str) {

        int n=str.length();

        for(int i=1;i<=n;i++){

            for(int j=1;j<=n-i;j++){

                System.out.print(".");

            }

            System.out.println(str.substring(0,i));

        }

    }

}
```

Output :

The screenshot shows a browser window for GeeksforGeeks with the URL geeksforgeeks.org/problems/triangle-growing-downwards2344/1?page=1&category=pattern-printing&status=solved&sortBy=difficulty. The page displays a Java code editor with the provided solution and an output window showing the correct output for the sample input "geeks". Below the editor, the "Compilation Results" section indicates "Problem Solved Successfully" with 80/80 test cases passed and 100% accuracy. The "Output Window" shows the expected output: ".g .ge .gee .geek .geeks".

```
1 // User function Template for Java
2 class Solution {
3     static void printTriangleDownwards(String str) {
4         // code here
5         int n=str.length();
6         for(int i=1;i<=n;i++){
7             for(int j=1;j<=n-i;j++){
8                 System.out.print(".");
9             }
10            System.out.println(str.substring(0,i));
11        }
12    }
13 }
```