

## ANIMAL CODE

```
interface Animal{  
    void eat();  
    void makeSound();  
}
```

```
interface Bird{  
    static int legs = 2;  
    void fly();  
}
```

```
class Parrot implements Bird, Animal{  
    public void eat(){  
        System.out.println("Parrots can eat up to 100 gms in a day")  
    }  
    public void makeSound(){  
        System.out.println("Parrots make sound of screech");  
    }  
    public void fly(){  
        System.out.println("Parrots can fly up to 50 miles in a day");  
    }  
}
```

## EMPLOYEE CODE

```
1 > import java.io.*;
6
7 abstract class Employee{
8     abstract void setSalary(int salary);
9     abstract int getSalary();
10    abstract void setGrade(String grade);
11    abstract String getGrade();
12    void label(){
13        System.out.print("Employee's data:\n");
14    }
15 }
16
17 class Engineer extends Employee{
18     private int salary;
19     private String grade;
20     public void setSalary(int salary){
21         this.salary = salary;
22     }
23     public int getSalary(){
24         return salary;
25     }
26     public void setGrade(String grade){
27         this.grade = grade;
28     }
29     public String getGrade(){
30         return grade;
31     }
32 }
33
34 class Manager extends Employee{
35     private int salary;
36     private String grade;
37     public void setSalary(int salary){
38         this.salary = salary;
39     }
40     public int getSalary(){
41         return salary;
42     }
43     public void setGrade(String grade){
44         this.grade = grade;
45     }
46     public String getGrade(){
47         return grade;
48     }
49 }
50 > public class Solution { ...
```

<https://youtu.be/c1ANbuywmBo> - Almost equivalent string CODE

Maximum cumulative hackos – SQL CODE

```
SELECT MAX(MONTHS * HACKOS) AS MAXIMUM_HACKOS, COUNT(*) AS NUMBER_OF_HACKERS  
FROM HACKER  
WHERE MONTHS * HACKOS = (SELECT MAX(MONTHS * HACKOS) FROM HACKER)
```

CAR CODE

```
import java.io.*;  
import java.util.*;  
import java.text.*;  
import java.math.*;  
import java.util.regex.*;  
  
class Car  
{  
    public void printTopSpeed()  
    {  
        System.out.println("Top speed of the vehicle is 100 kmph");  
    }  
    public void printTopSpeed(int topSpeed)  
    {  
        System.out.println("Top speed of the vehicle is " +topSpeed+ " kmph ");  
    }  
    public void printTopSpeed(String vehicleName,int topSpeed)  
    {
```

```

        System.out.println("Top speed of the vehicle " +vehicleName+ " is " +topSpeed+ " kmph ");
    }

    public void fuelType()

    {

        System.out.println("Car fuel type is Petrol");

    }

    class SUV extends Car{

        public void fuelType()

        {

            System.out.println("SUV fuel type is Diesel");

        }

    }

    public class Solution{

        public static void main(String[] args) {

            Scanner sc = new Scanner(System.in);

            for(int i=0;i<2;i++) {

                String input = sc.nextLine();

                Car suv = new SUV();

                if(input.equals("topSpeed")){

                    suv.topSpeed();

                }

                if(input.equals("fuelType")){

                    suv.fuelType();

                }

            }

        }

    }

```

```

Car car = new Car();

if(input.equals("topSpeed")){

    car.topSpeed();

}

if(input.equals("fuelType")){

    car.fuelType();

}

}

}

}

```

## CAR ENGINE CODE

BETA

Can't read the text? [Switch theme](#)

### 52. Car Engine

Complete the challenge by implementing the following class:

- class Car which has the following methods
  - \* public void printTopSpeed() method which prints "Top speed of the vehicle is 100 kmph" with a new line.
  - \* public void printTopSpeed(int topSpeed) method which prints ("Top speed of the vehicle is " + topSpeed + " kmph") with a new line.
  - \* public void printTopSpeed(String vehicleName, int topSpeed) method which prints ("Top speed of the vehicle " + vehicleName + " is " + topSpeed + " kmph") with a new line.

Once submitted, a hidden *Solution* class will check the code by calling appropriate methods. The *Solution* class can be expanded at the bottom of the code editing window.

▼ Sample Case 0

**Sample Input**

```

4
topSpeed
topSpeed 200
topSpeed BMW 220

```

Info

Language: Java 8

Autocomplete Ready

```

1 > import java.io.*; ...
6
7 class Car{
8     public void printTopSpeed(){
9         System.out.println("Top speed of the vehicle is 100 kmph");
10    }
11    public void printTopSpeed(int topSpeed){
12        System.out.println("Top speed of the vehicle is "+topSpeed+" kmph");
13    }
14    public void printTopSpeed(String vehicleName, int topSpeed){
15        System.out.println("Top speed of the vehicle "+vehicleName+" is "+topSpeed+" kmph");
16    }
17 }
18 > public class Solution { ...

```

Test Results

Custom Input

Run Code

Run Tests

Submit

## Nutrition code

```
import java.util.*;

abstract class Food{

    double proteins;

    double fats;

    double carbs;

    double tastyScore;

    void getMacroNutrients(){}

}

class Bread extends Food{

    String type;

    public Bread(double proteins,double fats,double carbs) {

        this.proteins=proteins;

        this.fats=fats;

        this.carbs=carbs;

        this.tastyScore=8;

        this.type = "vegeterian";

    }

    void getMacroNutrients()

    {

        System.out.println("A slice of bread has "+String.valueOf(this.proteins)+" gms of protein,
"+String.valueOf(this.fats)+

        " gms of fats and "+String.valueOf(this.carbs)+" gms of carbohydrates.");

    }

}
```

```

class Egg extends Food{

    String type;

    public Egg(double proteins,double fats,double carbs) {

        this.proteins=proteins;

        this.fats=fats;

        this.carbs=carbs;

        this.tastyScore=8;

        this.type = "non-vegeterian";

    }

    void getMacroNutrients()

    {

        System.out.println("An egg has "+String.valueOf(this.proteins)+" gms of protein,
"+String.valueOf(this.fats)+

            " gms of fats and "+String.valueOf(this.carbs)+" gms of carbohydrates.");

    }

}

public class Solution {

    public static void main(String args[]) {

        Scanner sc = new Scanner(System.in);

        int cnt = Integer.parseInt(sc.nextLine());

        for(int i=0;i< cnt;i++)

        {

            String name = sc.nextLine();

            if(name.equals("Bread")) {

                Bread breadobj = new Bread(4, 1.1, 13.8);

```

```

for (int j = 0; j < 3; j++) {

    String command = sc.nextLine();

    if (command.equals("getMacros"))

        breadobj.getMacroNutrients();

    else if (command.equals("getTaste"))

        System.out.println("Taste: " + breadobj.tastyScore);

    else if (command.equals("getType"))

        System.out.println("Bread is " + breadobj.type);

}

}

if(name.equals("Egg")) {

    Egg eggobj = new Egg(6.3, 5.3, 0.6);

    for (int j = 0; j < 3; j++) {

        String command = sc.nextLine();

        if (command.equals("getMacros"))

            eggobj.getMacroNutrients();

        else if (command.equals("getTaste"))

            System.out.println("Taste: " + eggobj.tastyScore);

        else if (command.equals("getType"))

            System.out.println("Bread is " + eggobj.type);

    }

}

}

}

}

```



## CAR FUELING

```
3 class Car {
4     public void topSpeed() {
5         System.out.println("Top Speed of the vehicle is 100 kmph");
6     }
7
8     public void fuelType() {
9         System.out.println("Car fuel type is Petrol");
10    }
11 }
12
13 class SUV extends Car {
14     public void fuelType() {
15         System.out.println("SUV fuel type is Diesel");
16     }
17 }
18
```

## BOTANY QUERY

Autocomplete Ready

1 /\*  
2 Enter your query here.  
3 Please append a semicolon ";" at the end of the query  
4 \*/  
5 select plant.name,a.type from plant  
6 join(select plant\_species,type,count(plant\_species)  
over(partition by plant\_species)as count from weather)  
a  
7 on a.plant\_species=plant.species and a.count=1;

Line: 7 Col: 48

Test Results

Run Query Submit

Compiled successfully. Correct answer.

Test case 0 Your Output (stdout)

Curtisia Hail

### Library structure code

```
class Library{

    private int number_of_books;

    private String name;

    private Map<String,Integer> bookGenres = new HashMap<>();


    public int getNumber_of_books() {

        return number_of_books;

    }


    public void setNumber_of_books(int number_of_books) {

        this.number_of_books = number_of_books;

    }


    public String getName() {

        return name;

    }


    public void setName(String name) {

        this.name = name;

    }


    public Map<String, Integer> getBookGenres() {

        return bookGenres;

    }

}
```

```

public void setBookGeneres(Map<String, Integer> bookGeneres) {

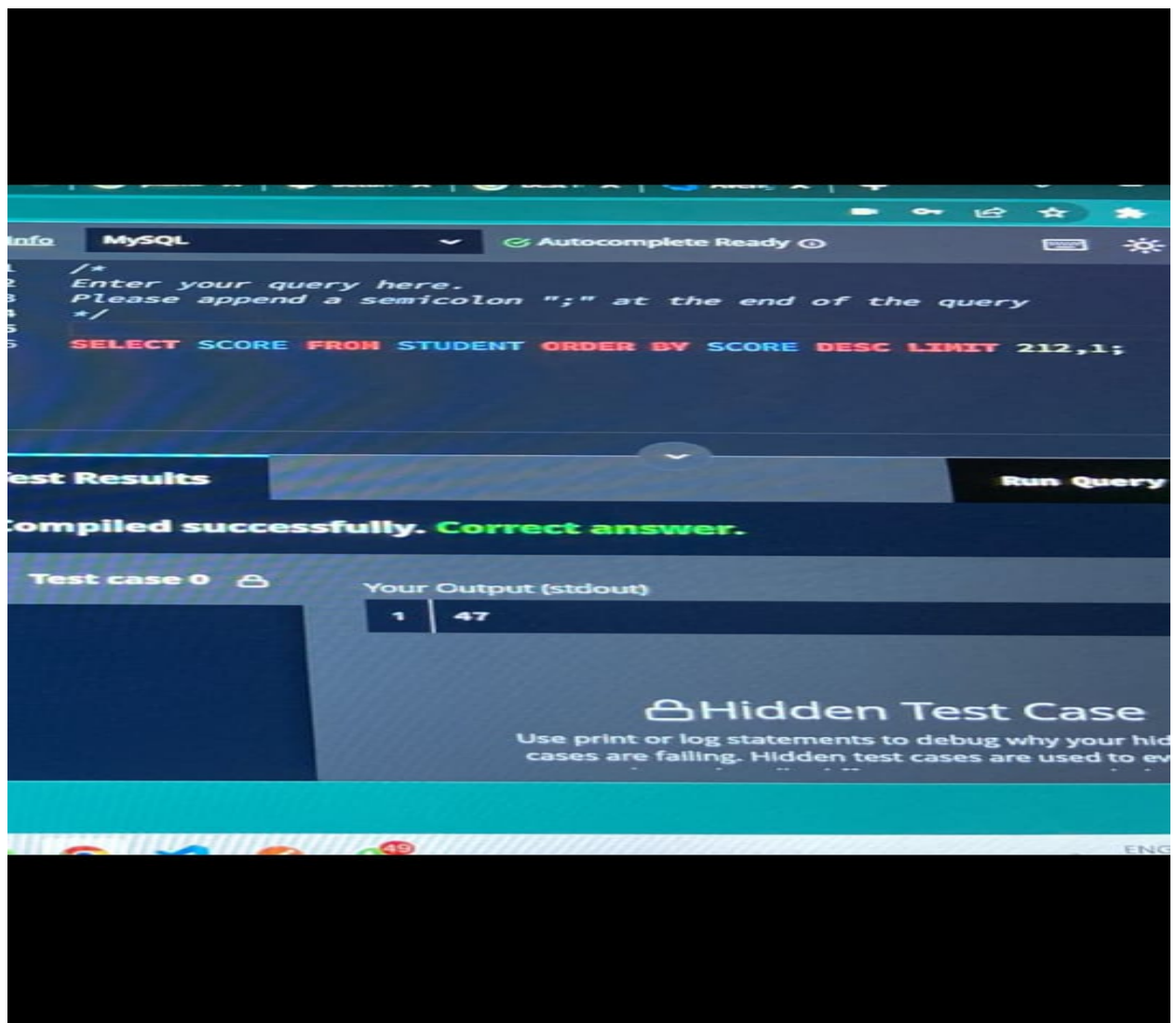
    this.bookGeneres = bookGeneres;

}

}

```

## STUDENT RANK CODE



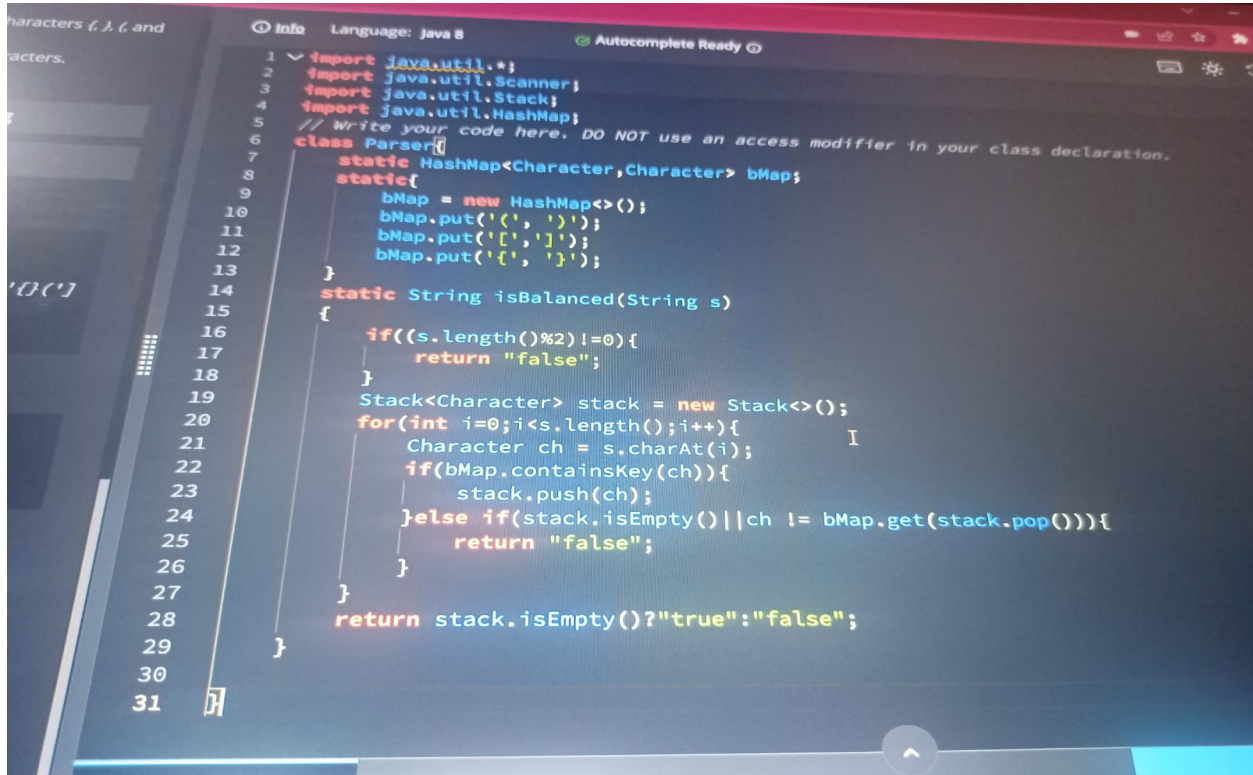
## Examination data management CODE- SQL

SELECT student\_id, subject, COUNT(\*) AS no\_of\_times

FROM Examination

GROUP BY student\_id, subject;

## Java Braces CODE



## Student class

class Student

{

private String myName;

private static int myRegNum = 0;

Student(String name)

{

myName = name;

myRegNum += 1;

```
}
```

```
@Override
```

```
public String toString() {
```

```
    return this.myRegNum + ": " + this.myName;
```

```
}
```

```
}
```

### **Substring CODE**

```
class Solution {
```

```
    public int countBinarySubstrings(String s) {
```

```
        int[] groups = new int[s.length()];
```

```
        int t = 0;
```

```
        groups[0] = 1;
```

```
        for (int i = 1; i < s.length(); i++) {
```

```
            if (s.charAt(i-1) != s.charAt(i)) {
```

```
                groups[++t] = 1;
```

```
            } else {
```

```
                groups[t]++;
```

```
            }
```

```
        }
```

```
        int ans = 0;
```

```
        for (int i = 1; i <= t; i++) {
```

```
            ans += Math.min(groups[i-1], groups[i]);
```

```
        }
```

```
        return ans;
    }
}
```

## Selling products CODE

```
1 import java.util.Map.Entry;
2 import java.util.Map;
3 import java.util.HashMap;
4 public class MinimumDistinctId
5 {
6     public static int getMinimumDistinctIds(int arr[], int n, int m)
7     {
8         Map<Integer, Integer> MAP = new HashMap<Integer, Integer>();
9         int temp = 0;
10        int cap = 0;
11        for (int i = 0; i < n; i++)
12        {
13            if (MAP.containsKey(arr[i]) == false)
14            {
15                MAP.put(arr[i], 1);
16                cap++;
17            }
18            else
19                MAP.put(arr[i], MAP.get(arr[i]) + 1);
20        }
21        for (Entry<Integer, Integer> entry:MAP.entrySet())
22        {
23            if (entry.getKey() <= m)
24            {
25                m = m - entry.getKey();
26                temp++;
27            }
28            else
29                return cap - temp;
30        }
31        return cap - temp;
32    }
33    public static void main(String[] args)
34    {
35        int arr[] = {1, 1,1,2,3,2};
36        int m = 2;
37        System.out.println(getMinimumDistinctIds(arr, arr.length, m));
38    }
39 }
```

Execute Mode, Version, Inputs & Arguments

JDK 17.0.1  ☐ Interactive Stdin Inputs

CommandLine Arguments

Result

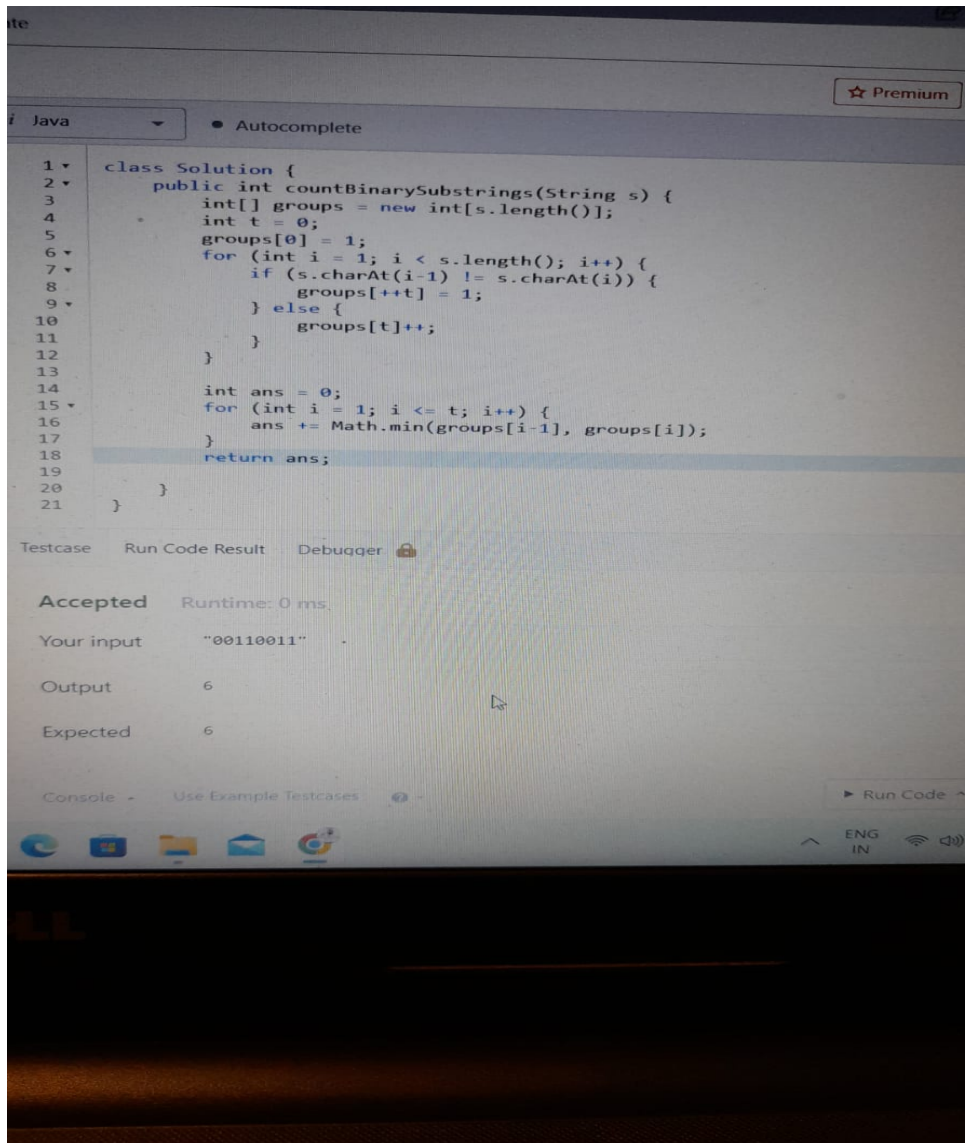
CPU Time: 0.07 sec(s), Memory: 31952 kilobyte(s)

2

Note: Please check [our documentation](#), or [Youtube channel](#). for more details



## Binary string CODE

A screenshot of a Java IDE interface. The top bar shows 'Java' and 'Autocomplete'. A 'Premium' badge is in the top right. The code editor displays a Java class 'Solution' with a method 'countBinarySubstrings'. The code counts the number of binary substrings in a given string 's'. It uses an array 'groups' to store the count of consecutive 0s and 1s. The logic is as follows: initialize 'groups[0] = 1'; iterate from index 1 to the end of the string; if the current character is different from the previous one, increment the current group count and reset the previous group count to 1; otherwise, increment the previous group count. Finally, iterate through the 'groups' array and sum the minimum of adjacent group counts to get the total number of binary substrings. The test case section shows 'Accepted' status, 'Runtime: 0 ms', 'Your input: "00110011"', 'Output: 6', and 'Expected: 6'. The bottom status bar shows 'Console', 'Use Example Testcases', and a 'Run Code' button. The system tray at the bottom shows icons for a globe, a folder, a mail icon, and a Google icon, along with system status icons for volume, network, and language (ENG IN).

```
1 class Solution {
2     public int countBinarySubstrings(String s) {
3         int[] groups = new int[s.length()];
4         int t = 0;
5         groups[0] = 1;
6         for (int i = 1; i < s.length(); i++) {
7             if (s.charAt(i-1) != s.charAt(i)) {
8                 groups[++t] = 1;
9             } else {
10                 groups[t]++;
11             }
12         }
13         int ans = 0;
14         for (int i = 1; i <= t; i++) {
15             ans += Math.min(groups[i-1], groups[i]);
16         }
17         return ans;
18     }
19 }
20
21 }
```

Testcase Run Code Result Debugger

Accepted Runtime: 0 ms

Your input "00110011"

Output 6

Expected 6

Console Use Example Testcases Run Code

## Car fueling

class Car

```
{
    public void topSpeed()
    {
        System.out.println("Top speed of the vehicle is 100 kmph");
    }
}
```

```

public void fuelType()
{
    System.out.println("Car fuel type is Petrol");
}
}

```

```

class SUV extends Car{
    public void fuelType()
    {
        System.out.println("SUV fuel type is Diesel");
    }
}

```

#### EMPLOYEE PROFILE CODE

The screenshot shows a Java IDE with the following code:

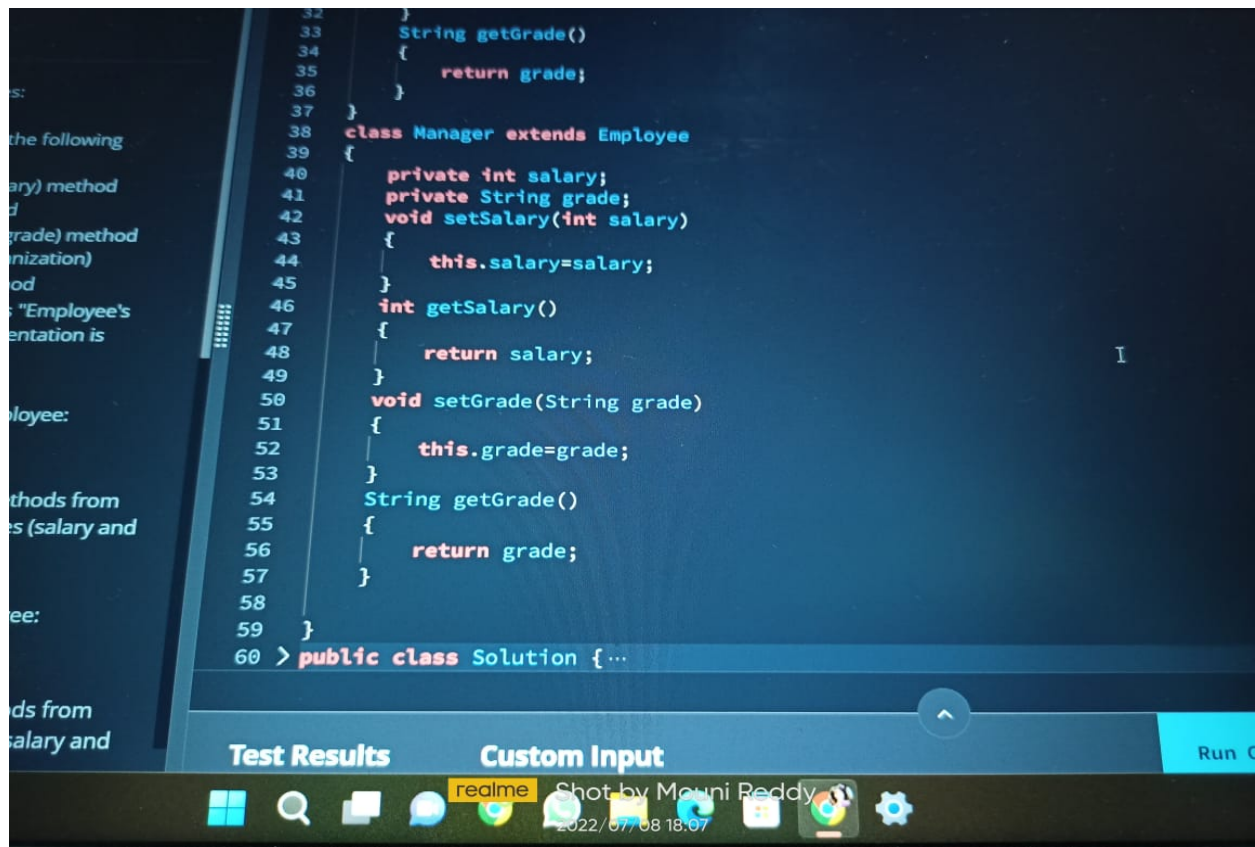
```

1 > import java.io.*;
2
3 abstract class Employee {
4     abstract void setSalary(int salary);
5     abstract int getSalary();
6     abstract void setGrade(String grade);
7     abstract String getGrade();
8     protected void label()
9     {
10         System.out.println("Employee's data:");
11     }
12 }
13
14 class Engineer extends Employee
15 {
16     private int salary;
17     private String grade;
18     void setSalary(int salary)
19     {
20         this.salary=salary;
21     }
22     int getSalary()
23     {
24         return salary;
25     }
26     void setGrade(String grade)
27     {
28         this.grade=grade;
29     }
30     String getGrade()
31     {
32         return grade;
33     }
34 }

```

The IDE interface includes a sidebar on the left with a search bar and a list of files. The bottom of the screen shows a taskbar with various application icons, including a terminal, a file explorer, and a settings icon. A watermark "realme" is visible in the bottom center, and a timestamp "2022/07/08 18:07" is in the bottom right corner.





### Count Binary string code

```
class Solution {
    public int countBinarySubstrings(String s) {
        int curr = 1, prev = 0, ans = 0;
        for (int i = 1; i < s.length(); i++)
            if (s.charAt(i) == s.charAt(i-1)) curr++;
            else {
                ans += Math.min(curr, prev);
                prev = curr;
                curr = 1;
            }
        return ans + Math.min(curr, prev);
    }
}
```

```
}
```

### Serial multiplier CODE

```
import java.util.*;

public class SerialMultiplier
{
    static int first=1, second=1, third=1, fourth=1, fifth=1;

    static int result=0;

    public SerialMultiplier(int first)
    {
        this.first=first;
        result=first;
    }

    public SerialMultiplier(int first, int second)
    {
        this.first=first;
        this.second=second;
        result=first*second;
    }

    public SerialMultiplier(int first, int second, int third)
    {
        this.first=first;
        this.second=second;
        this.third=third;
        result=first*second*third;
    }
}
```

```

}

public SerialMultiplier(int first, int second, int third, int fourth)
{
    this.first=first;

    this.second=second;

    this.third=third;

    this.fourth=fourth;

    result=first*second*third*fourth;
}

public SerialMultiplier(int first, int second, int third, int fourth, int fifth)
{
    this.first=first;

    this.second=second;

    this.third=third;

    this.fourth=fourth;

    this.fifth=fifth;

    result=first*second*third*fourth*fifth;

}

public static void main( String args[])
{
    Scanner sc= new Scanner(System.in);

    int n=sc.nextInt();

    int a[]=new int[n];

```

```
for(int i=0;i<n;i++)
```

```
{
```

```
    a[i]=sc.nextInt();
```

```
}
```

```
if(n==1)
```

```
{
```

```
    SerialMultiplier obj=new SerialMultiplier(a[0]);
```

```
}
```

```
if(n==2)
```

```
{
```

```
    SerialMultiplier obj=new SerialMultiplier(a[0],a[1]);
```

```
}
```

```
if(n==3)
```

```
{
```

```
    SerialMultiplier obj=new SerialMultiplier(a[0],a[1],a[2]);
```

```
}
```

```
if(n==4)
```

```
{
```

```
    SerialMultiplier obj=new SerialMultiplier(a[0],a[1],a[2],a[3]);
```

```
}
```

```
if(n==5)
```

```
{
```

```
    SerialMultiplier obj=new SerialMultiplier(a[0],a[1],a[2],a[3],a[4]);
```

```
}
```

```
        System.out.println(result);
    }

}
```

### Count Binary Substrings CODE

```
class Solution {

    public int countBinarySubstrings(String s) {

        int count=0;

        for(int i=0;i<s.length()-1;i++){

            count+=doCount(s,i,i+1,0);

        }

        return count;

    }

    private int doCount(String s,int start,int end,int count){

        if(s.charAt(start)=='0'&& s.charAt(end)=='1'){

            while(start>=0&&end<s.length()&&s.charAt(start)=='0'&&s.charAt(end)=='1'){

                count++;

                start--;

                end++;

            }

        }else if(s.charAt(start)=='1'&&s.charAt(end)=='0'){
```

```
while(start>=0&&end<s.length()&&s.charAt(start)=='1'&&s.charAt(end)=='0'){  
    count++;  
    start--;  
    end++;  
}  
}  
return count;  
}  
}
```