IT 23033

Answer to the question no-1

```
import Java.io.File
import Java.util.Scanner
import Java.io.PrintWriter

Public class SeriesSum {
    Public static void main(String[] args){
    Try{
        File file = new File("input.txt");
        Scanner sc = new Scanner(file);
        PrintWriter PW = new Printwr("Output.txt");
        if(Sc.hasNextLine()){
        ~~Sc~~ String Str = Sc.NextLine();
        String[] s = Str.Split(",");
        for(int i=0; i<s.length(); i++){
            int n= Integer.parsInt(s[i]);
            int SSum = (n*(n+1))/2;
            PW.Print(SSum);
```

IT 23033

```
if (i!= ~~str~~.length() -1)
  ~~Pr~~ pw.Print(",");
}
~~fi~~se.close();
 pw.close();
} catch (Exception e){
   ~~S~~ System.Out.Println(" File not found");
 }
}
}
```

IT23033

Answer to the question no - 3

```
Inport ~~Jaga~~ java.util.Scanner;
public class factorion{
    public static void main(String[] args){
    ~~int~~ Scanner sc = new Scanner("System.in");
      int start = sc.NextInt();
      int end = sc.~~NextInt~~();
      for(int i=start; i<=end; i++){
          int Sum=0; temp = i;
          while(temp!=0){
          ~~Sum += temp%10~~
          Sum += fact(temp%10);
           temp /= 10;
          }
          if(i==Sum)
            System.out.print(i+"  ");
      }
    Sc.close
}
```

IT23033

```
public static int fact (int n){
  int facto=1;
  for (int i=1; i<=n; i++)
      facto*=i;
  return facto;
}

}
```

IT23033

Answer to the question no - 4

Differenence among class, local and listance valiable;

Class Variable	Instance Variable	Local variable
1) Declared with 'Static' keyward insidea class and Outside of method	1) Declared without Static keyward. inside a class outside of method.	1) Declared inside method, constructor on block.
2) variable is shared accross all instances	2) belongs to a instance of the class	2) Limited access to the method on block.
3) Intialized with default value of 0 on null.	3) Initialized with default value of 0 on null	3) Not initialized with value. must be asigned a value before use
4) accessed using classname.variable Name.	4) accessed using objectName.VariableName.	5) accessed directly within the black.

IT23033

Answer to the question no- 5

```
Public class ArraySum {
    public static int ArraySum (int[] array ){
      int Sum= 0;
      for (int i : array)
      {
        Sum += i;
      }
      return Sum;
    }
    public static void main(String[] args){
      int [] array = { 1, 2, 3, 5 ,10 ,20};
      ~~Publi~~
      System.Out.println("Sum of array is :" +
                        ArraySum(array));

    }
}
```

IT23033

Answer to the question no - 8

```
import Java.util.Scanner;
public class CharCheck {
    public static void main(String[] args){
     Scanner sc = new Scanner(System.in);
     char ch = sc.next().charAt(0);
     if (Character.IsLetter(ch))
        System.out.println(ch +" is a Letter");
     else if (Character.isDigit (ch))
        System.out.println( ch +" is a Digit");
     else if (Character.isWhitespace(ch))
        System.Out.Println (ch+" is a whitespace.");

     else {
      System.Out.Println (ch+ "is a special characte
     }
     sc.close();
    }
}
```

IT 23033

Answer to the question no-7

```
import ~~Java~~ ~~$~~ Java.util.Scanner;
pablie class root{
    public static void main(String[] args){
        int a,b,c, det;
      Scanner sc= new Scanner(System.in);
     a =sc.nextInt; b =sc.nextInt; c =sc.next Int;
     det= b*b-4*a*c;
    if(det<0)
        ~~pr~~system.Out.println(" root is not real");
    else{
    double root1=(double)(-b+Math.sqrt(det))/(2*a);
   double root 2 =(double)(-b-Math.sqrt(det))/(2*a);

  System.Out.println("Smallest root is: "+ Math.min(
                                      root1, root2));

  sc.close();
  }
}
```

it23033

<u>Answer to the question no-9</u>

Method overriding is a feature in Java, that allows subclasses to provide new implication for a method that is defined by it's superclass.

Working process of overriding:

i) When a subclass overrides a method, Only the subclass version of the method is exiented.

ii) this process is known as runtime polymorphism. as it resolves at runtime.

iii) Overriding enables customization for subclass objects while ~~con~~ maintaining a constant interface.

When subclass overrides a method:

Subclass method executes replacing the superclass method. Runtime polymorphism executes in runtime.

Super can call the overridden superclass method.

☐ Super keyward is used to call on overridden method from superclass. This allows the subclass to extend or modify the behaviour without replacing it.

☐ Issues of Overriding:

I) can not reduce access like public → Private.

II) Can not throw ~~boarden~~ brodden exceptions.

III) Final methon can not be overridden.

☐ Issues of constructor:

i) Constructors can not be overridden as they are not inherited.

ii) Super() must be used for superclass initialization.

it23633

Answer to the question no - 10

Diffenenee between statie and non-statie members:

Statie	non - statie
1) Belong to a class and shared among all objeets.	1) Belong to the individual objects. each instance has it's own copy-
11) Accessed using class name on instanee	11) Accessed only through an Object of the class
111) Can be called without cineating class objeet.	111) can not be called without creating class objeet.
1v) public class abe{ ~~Public~~ static void demo() { System. Out. println(" this is statie."); } }	1v) public class abc{ void demo(){ System. Out. println(" This is non-statie."); } }

it23033

```
import Java.util.Scanner
Public class palindrome{
    public static void main(){
        ~~String~~ Scanner sc =new Scanner(System.in);
        String s = sc.nextLine();
        int l = 0, r = s.length()-1;
        boolean f = true;
        While (l<r){
            if (s.charAt(l) != s.charAt(r))
                f= false;
            l++; r--;
        }
        if(f)
            System.out.println("Palindrome");
        else
            System.out.println("Not palindrome");
        sc.close();
    }
}
```

1+23033

Answer to the question no- 11

Difference between Abstract class and Interface.

Abstract class	interface
1) Abstract class can have both conecrete and abstract method.	1) interface contains only abstract method.
11) Can have instance Variable with any access modifier	11) Can have only public Static final constants.
111) Can have constructor	111) can not have constructor
IV) Can be inharited by Subclass	IV) Only can be implimented by Subclass
V) abstract class Animal{ abstract ~~public~~ String Eat(); public String Sound(){ Return "make sound"; } }	V) Interface Animal{ String Eat(); String Sound(); }

IT23033

Answer to the question no. 12

~~BasieClass~~

```
Class BasieClass{
 Void PrintResult { String a , String res){
      System.Out.println(a + " = " + res);
  }
 Class SumClass extends BasieClass{
        void SeriesSum(){
          double sum = 0.0;
          for(double i=0.1; i<=1.0; i+= 0.1)
              Sum += i;
          PrintResult ("Series Sum", Sum);
 }
Void Divison Multiple Class extends BasieClass{
     void GCDandLCM (int a, int b){
       PrintResult ("GCD of "+a+"and"+b, GCD (a,b));
       PrintResult ("LCM of"+a+" and"+b, LCM(a,b));
     }
}
```

```
int GCD (int a, int b){
        if( b==0)
          return a;
      GCD(a, b%a);
      }
int Lcm ( int a, int b){
      int lcm = a*b/GCD(a,b);
      return lcm;
   }
}
Void NumberConversionClass extends BasicClass{
    Void convert(int n){
    PrintResult("binary of"+n, Integer.toBinaryString(n));
    PrintResult("Hexadecimal of +n, Integer.toHexaString(n));
    PrintResult ("Octal of"+n, Integer.toOctalString(n));
}
Void CustomPrintClass extends BasicClass{
      Void Pr(String S){
       System.Out.println (S);
      }
}
```

IT23033

```
public class MainClass{
 public static void main (String[] args){
   SumClass se = new SumClass();
   DivisonMultipleClass GcdLcm = new DivisonMultipleClass();

   NumberConversionClass cnv = new NumberConversionClass();
   Customprintclas cp = new Customprintclass();

   se.SeriesSum();
   GcdLcm.GCDandLCM(12,15);
   cnv.convert(8);
   CP.PR("this is custom print");

 }
}
```