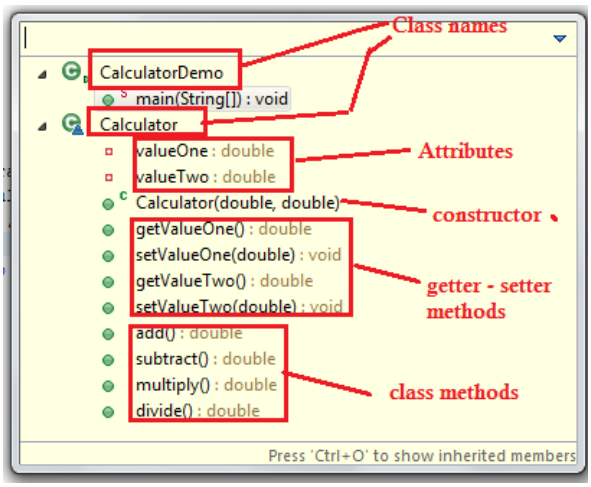
Refer below sample problem with solution to solve remaining problems. Ensure same approach is followed. Use exactly same class names, attribute names and types, constructor, method names and signatures as specified in the problem statement. Any mistake on these instructions may result into invalid submission of the assignments even if logic is 100% correct.

1. Declare class Calculator with attributes:
   double valueOne, double valueTwo
   Write below methods in this class.
   Add() //addes two values
   Subtract() //subtracts valueTwo from valueOne
   Multiply() // multiplies valueTwo and valueOne
   Divide() //Divides valueTwo with valueOne. Returns 0 of valueOne is 0.

   Create calculator object and test above methods from main method in driver class (CalculatorDemo). Follow class outline diagram as given below. Ensure class attributes are private and other methods are public.




```
import java.util.Scanner;
public class CalculatorDemo {


	public static void main(String[] args) {

		Scanner sc = new Scanner(System.in);
		System.out.println("Enter first value");
		double one = sc.nextDouble();
		System.out.println("Enter second value");
		double two = sc.nextDouble();
		//create object
		Calculator calc = new Calculator(one, two);
		System.out.println("Addition:"+ calc.add());
		System.out.println("Subtraction:"+ calc.subtract());
		System.out.println("Multiplication:"+ calc.multiply());
		System.out.println("Division:"+calc.divide());
		calc.setValueTwo(0);
```

```
            System.out.println("Result for divide by
zero:"+calc.divide());
      }

}

class Calculator
{
      private double valueOne;
      private double valueTwo;

      public Calculator(double valueOne, double valueTwo) {
            this.valueOne = valueOne;
            this.valueTwo = valueTwo;
      }
      public double getValueOne() {
            return valueOne;
      }
      public void setValueOne(double valueOne) {
            this.valueOne = valueOne;
      }
      public double getValueTwo() {
            return valueTwo;
      }
      public void setValueTwo(double valueTwo) {
            this.valueTwo = valueTwo;
      }

      public double add(){
            return valueOne + valueTwo;
      }

      public double subtract(){
            return valueOne - valueTwo;
      }

      public double multiply(){
            return  valueOne* valueTwo;
      }

      public double divide()
      {
            if(valueTwo == 0)
                  return 0;
            else
                  return valueOne/valueTwo;
      }
}
```

Sample Output:

```
Enter first value
22
Enter second value
9
Addition:31.0
Subtraction:13.0
Multiplication:198.0
Division:2.4444444444444446
Result for divide by zero:0.0
```

Each trainee is supposed to write above program and try out even if they are aware about Java language. Along with learning Java numeric computations, below points are very important to practice and applicable throughout this ILP training as well as most important for any software code.

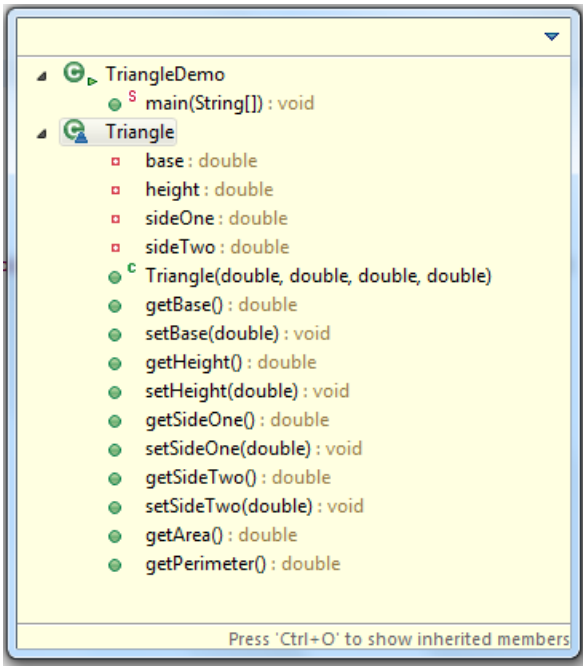
- Use exactly same class names as mentioned
- Use exactly same method signature (method name, return type, method parameter type, position of each method parameter)
- Define attributes with same name and data type as given in class outline.
- Define constructors and getter setters as given in the class outline.
- Ensure attributes are private and other methods which will be called from main method, getter-setter methods and constructor is public.
- Use main method only for input and output and testing object creation and object methods.

As mentioned above, any logic which may be 100% correct is not valid if above points are not taken care. Hence, simply building logic does not certify us as project ready. Building exact and complete solution does.

After this problem is tried out, think over the solution if we want to create object of Calculator class without specifying values for valueOne and valueTwo attributes. Disuss and implement the same in above solution.

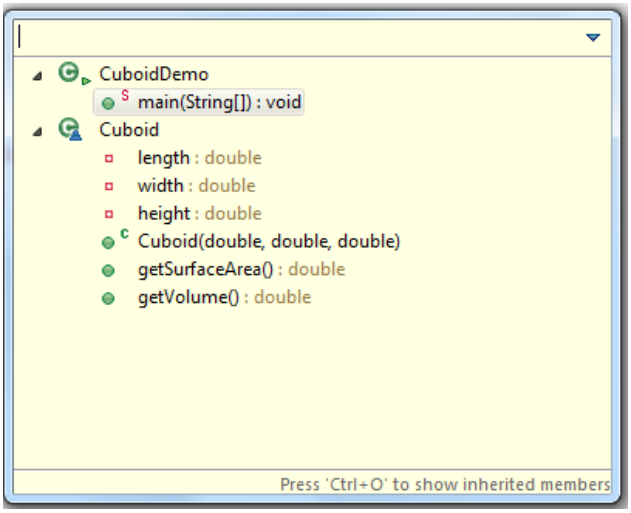
2. Declare class Triangle with below attributes:
   double base, double height, double sideOne, doubleSideTwo
   Write below methods in this class.
   GetArea() // return area of the triangle
   GetPeremeter() //return peremeter of the triangle

   Create triangle object and test above methods from main method in driver class (TriangleDemo). Follow class outline diagram as given below. Ensure class attributes are private and other methods are public.



3. Declare class Cuboid with below attributes:
   length, width, height
   Write below methods in this class.
   GetSurfaceArea() // return surface area of the cuboid
   GetVolume() // return volume of the cuboid

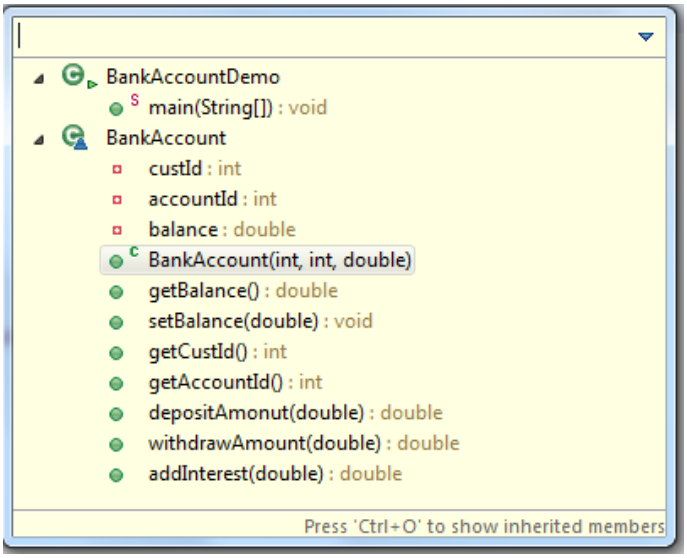
   Create cuboid object and test above methods using main method in driver class (CuboidDemo). Follow class outline diagram as given below. Ensure class attributes are private and other methods are public.



In Cuboid class, there are no getters and setters. Think about any possible limitation when we do not have getters and setters. Discuss with your colleagues.

4. Create class BankAccount with below attributes:
   customerId, accountId,balance
   Write below methods in this class.
   DepositAmonut(double amount) // adds the amount to existing balance and returns new balance
   WithdrawAmount(double amount) // deducts the amount from existing balance and returns new balance.
   // however, if amount is greater than balance, returns -1
   AddInterest(double percent) // adds amount based on interest percentage and returns new balance

   Create bank account object and test above methods using main method in driver class (BankAccountDemo). Follow class diagram as given below. Ensure class attributes are private and other methods are public.



In BankAccount class, we do not have setter methods for custId and accountId. What is the advantage? Discuss with your colleagues.