Introduction to OOPs

- Languages like Pascal, C, FORTRAN, and COBOL are called procedure-oriented programming languages. Since in these languages, a programmer uses procedures or functions to perform a task. When the programmer wants to write a program, he will first divide the task into separate sub tasks, each of which is expressed as functions/ procedures. This approach is called ***procedure-oriented approach.***
- The languages like C++ and Java use classes and object in their programs and are called Object Oriented Programming languages. The main task is divided into several modules and these are represented as classes. Each class can perform some tasks for which several methods are written in a class. This approach is called "***Object Oriented approach".***

OOP Vs POP

	OOP	POP
Definition	OOP stands for Object-oriented programming it focuses on data rather than the algorithm.	POP stands for Procedure-oriented programming, focuses on procedural abstractions.
Programs	Divided into small chunks called objects which are instances of classes.	Divided into small parts based on the functions.
Accessing Mode	Four accessing modes are used in OOP to access attributes or methods – 'Private', 'Public', 'default' , and 'Protected'	No such accessing mode is required
Execution	Various methods can work simultaneously	Follows a systematic step-by-step approach to execute functions.
Security	It is of high secure because of it data hiding feature	There is no such way of data hiding in POP, thus making it less secure.

Features of OOP

- There are mainly '4' features of OOP's are there, which are listed below.

1. Encapsulation
2. Inheritance
3. Polymorphism
4. Abstraction

Understanding Encapsulation:

- Generally there are two forms of encapsulation in OOP.

First Form:

- Encapsulation is a technique that packages related data and behaviors into a single unit.
- Here, the common characteristics and behaviors of a student are packaged into a single unit: the Studentclass.
- This is the process of encapsulation.
- Encapsulation hides implementation details of the Student class from other objects.

Second Form:

- Encapsulation is the technique of making the fields in a class private and providing access to the fields via methods.
- If a field is declared private, it cannot be accessed by anyone outside the class.
- Such that we provide security to the data from outside world without misusing it, which is commonly known as 'Information Hiding' or 'Data Hiding'.
- In process of Information Hiding, the other objects cannot access the data directly. Instead, they have to invoke the getters which are designed to protect the data from misuse or unwanted changes.

What is the need for Encapsulation?

- **Flexibility**: It's more flexible and easier to change the encapsulated code with new requirements.
- **Reusability**: Encapsulated code can be reused throughout the application or across multiple applications.
- **Maintainability**: If an application is encapsulated in separate units (classes, interfaces, methods, setters, getters, etc) then it's easy to change or update a part of the application without affecting other parts, which reduces the time of maintenance.

OOP's Features:

1) Encapsulation

2) Inheritance

3) Polymorphism

4) Abstraction

Encapsulation

It is a process of wrapping up of data or binding up of data into a Single unit.

It is a process of making the fields as private & providing access to those fileds with the help of public methods, ie. through setters & getters methods.

This is not encapsulation

ClassA.java

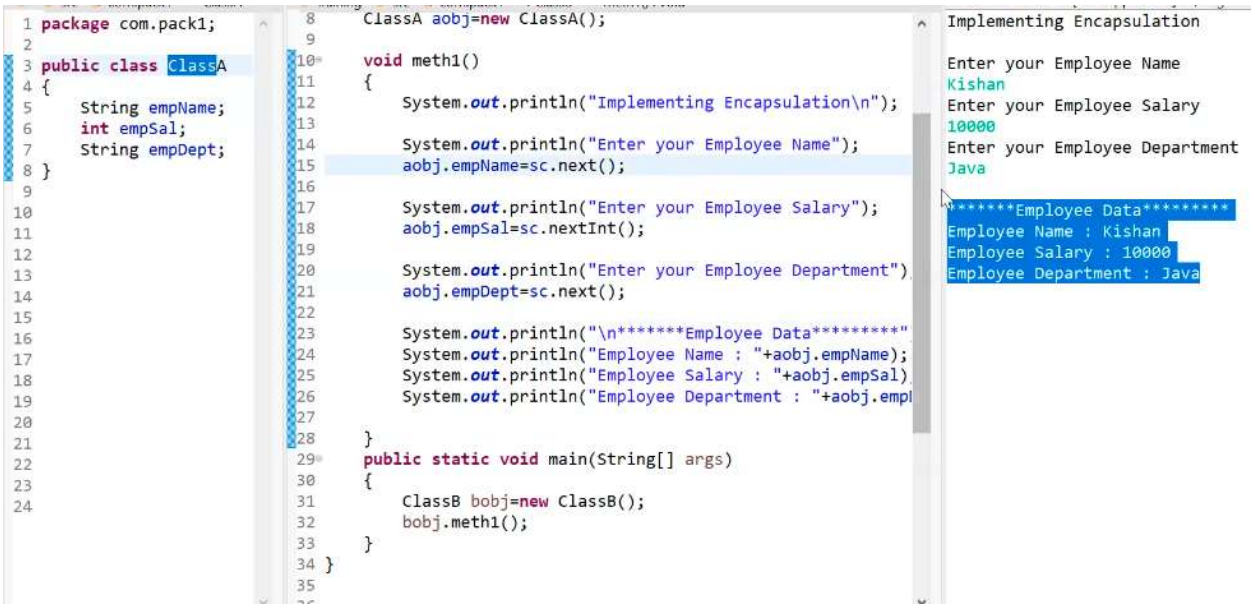
```
1  package com.pack1;
2
3  public class ClassA
4  {
5      String empName;
6      int empSal;
7      String empDept;
8  }
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
```

ClassB.java

```
1  package com.pack1;
2
3  import java.util.Scanner;
4
5  public class ClassB
6  {
7      Scanner sc=new Scanner(System.in);
8      ClassA aobj=new ClassA();
9
10     void meth1()
11     {
12         System.out.println("Implementing Encapsulation\n");
13
14         System.out.println("Enter your Employee Name");
15         aobj.empName=sc.next();
16
17         System.out.println("Enter your Employee Salary");
18         aobj.empSal=sc.nextInt();
19
20         System.out.println("Enter your Employee Department");
21         aobj.empDept=sc.next();
22
23         System.out.println("\n*******Employee Data*********");
24         System.out.println("Employee Name : "+aobj.empName);
25         System.out.println("Employee Salary : "+aobj.empSal);
26         System.out.println("Employee Department : "+aobj.empDept);
27
```

Console

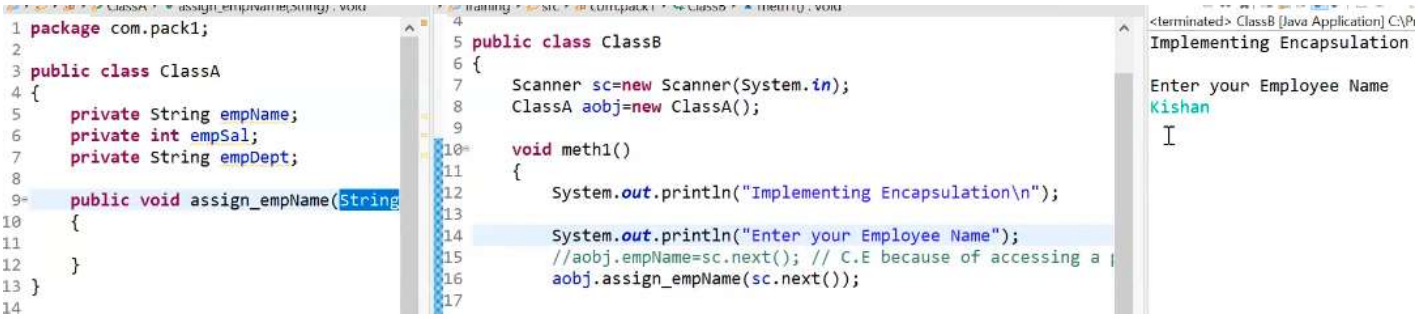
```
<terminated> ClassB [Java Applicatio
Implementing Encapsul

Enter your Employee N
Kishan
Enter your Employee S
10000
Enter your Employee D
Java

*******Employee Data*
Employee Name : Kisha
Employee Salary : 100
Employee Department :
```



```
package com.pack1;

public class ClassA
{
    String empName;
    int empSal;
    String empDept;
}
```

```
    ClassA aobj=new ClassA();

    void meth1()
    {
        System.out.println("Implementing Encapsulation\n");

        System.out.println("Enter your Employee Name");
        aobj.empName=sc.next();

        System.out.println("Enter your Employee Salary");
        aobj.empSal=sc.nextInt();

        System.out.println("Enter your Employee Department");
        aobj.empDept=sc.next();

        System.out.println("\n******Employee Data*********"
        System.out.println("Employee Name : "+aobj.empName);
        System.out.println("Employee Salary : "+aobj.empSal);
        System.out.println("Employee Department : "+aobj.empl

    }
    public static void main(String[] args)
    {
        ClassB bobj=new ClassB();
        bobj.meth1();
    }
}
```

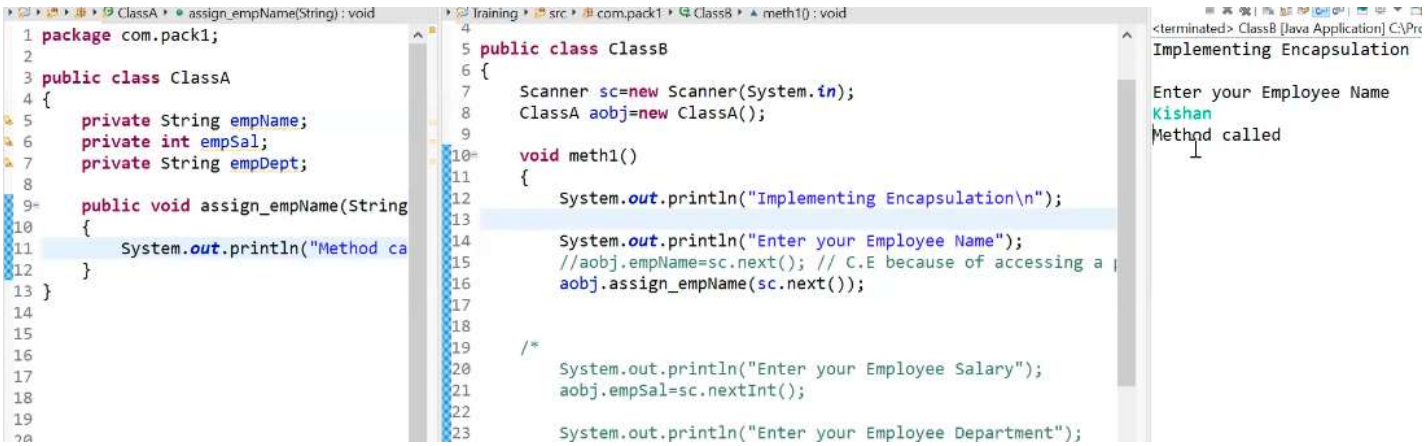
```
Implementing Encapsulation

Enter your Employee Name
Kishan
Enter your Employee Salary
10000
Enter your Employee Department
Java

******Employee Data*********
Employee Name : Kishan
Employee Salary : 10000
Employee Department : Java
```



```
package com.pack1;

public class ClassA
{
    private String empName;
    private int empSal;
    private String empDept;

    public void assign_empName(String
    {

    }
}
```

```
public class ClassB
{
    Scanner sc=new Scanner(System.in);
    ClassA aobj=new ClassA();

    void meth1()
    {
        System.out.println("Implementing Encapsulation\n");

        System.out.println("Enter your Employee Name");
        //aobj.empName=sc.next(); // C.E because of accessing a
        aobj.assign_empName(sc.next());
```

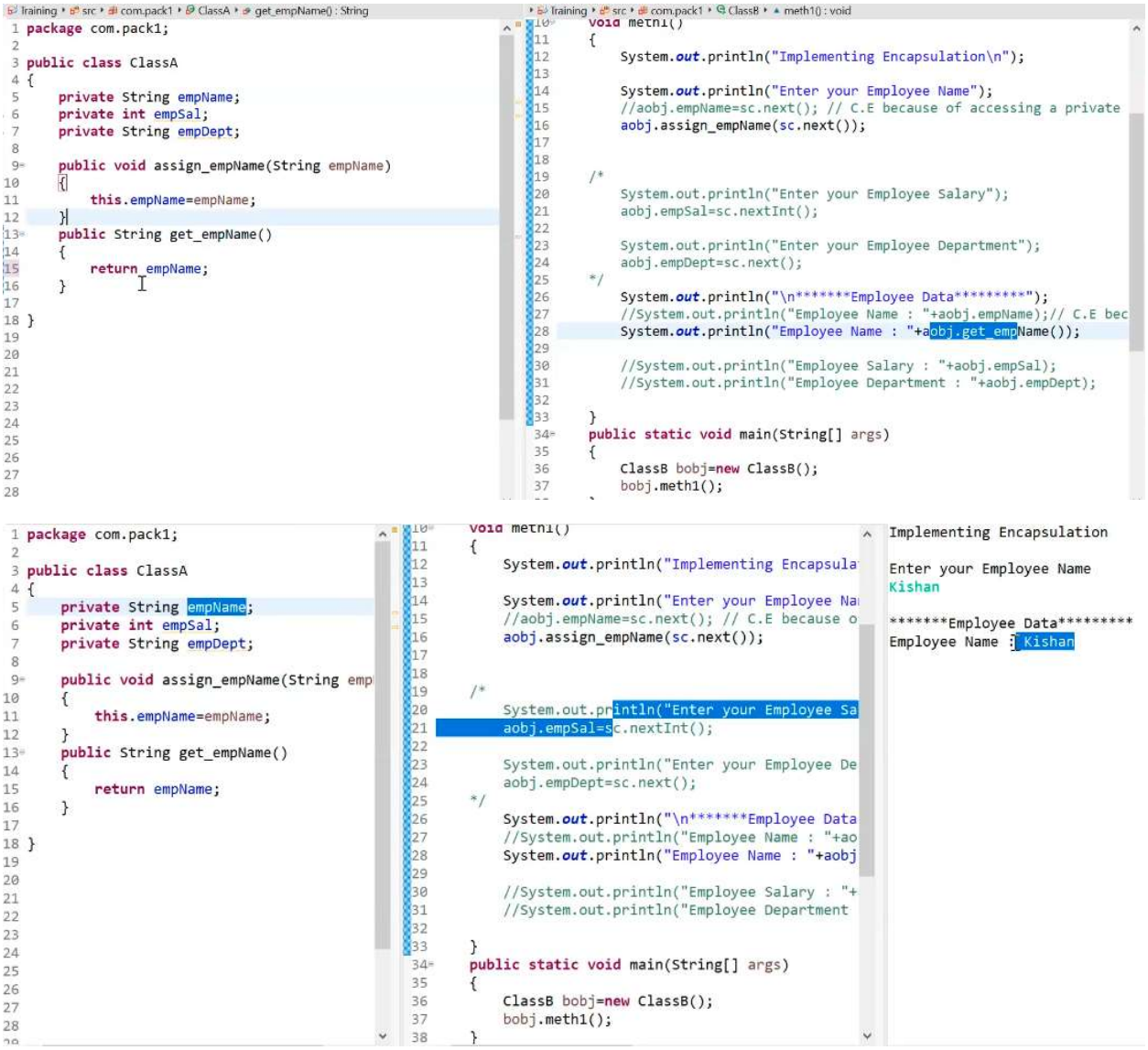
```
<terminated> ClassB [Java Application] C:\P
Implementing Encapsulation

Enter your Employee Name
Kishan
```



```
package com.pack1;

public class ClassA
{
    private String empName;
    private int empSal;
    private String empDept;

    public void assign_empName(String
    {
        System.out.println("Method ca
    }
}
```

```
public class ClassB
{
    Scanner sc=new Scanner(System.in);
    ClassA aobj=new ClassA();

    void meth1()
    {
        System.out.println("Implementing Encapsulation\n");

        System.out.println("Enter your Employee Name");
        //aobj.empName=sc.next(); // C.E because of accessing a
        aobj.assign_empName(sc.next());


    /*
        System.out.println("Enter your Employee Salary");
        aobj.empSal=sc.nextInt();

        System.out.println("Enter your Employee Department");
```

```
<terminated> ClassB [Java Application] C:\Pr
Implementing Encapsulation

Enter your Employee Name
Kishan
Method called
```



```
package com.pack1;

public class ClassA
{
    private String empName;
    private int empSal;
    private String empDept;

    public void assign_empName(String empName)
    {
        this.empName=empName;
    }
    public String get_empName()
    {
        return empName;
    }

}
```

```
    void meth1()
    {
        System.out.println("Implementing Encapsulation\n");

        System.out.println("Enter your Employee Name");
        //aobj.empName=sc.next(); // C.E because of accessing a private
        aobj.assign_empName(sc.next());


    /*
        System.out.println("Enter your Employee Salary");
        aobj.empSal=sc.nextInt();

        System.out.println("Enter your Employee Department");
        aobj.empDept=sc.next();
    */
        System.out.println("\n*******Employee Data*********");
        //System.out.println("Employee Name : "+aobj.empName);// C.E bec
        System.out.println("Employee Name : "+aobj.get_empName());

        //System.out.println("Employee Salary : "+aobj.empSal);
        //System.out.println("Employee Department : "+aobj.empDept);

    }
    public static void main(String[] args)
    {
        ClassB bobj=new ClassB();
        bobj.meth1();
    }
```

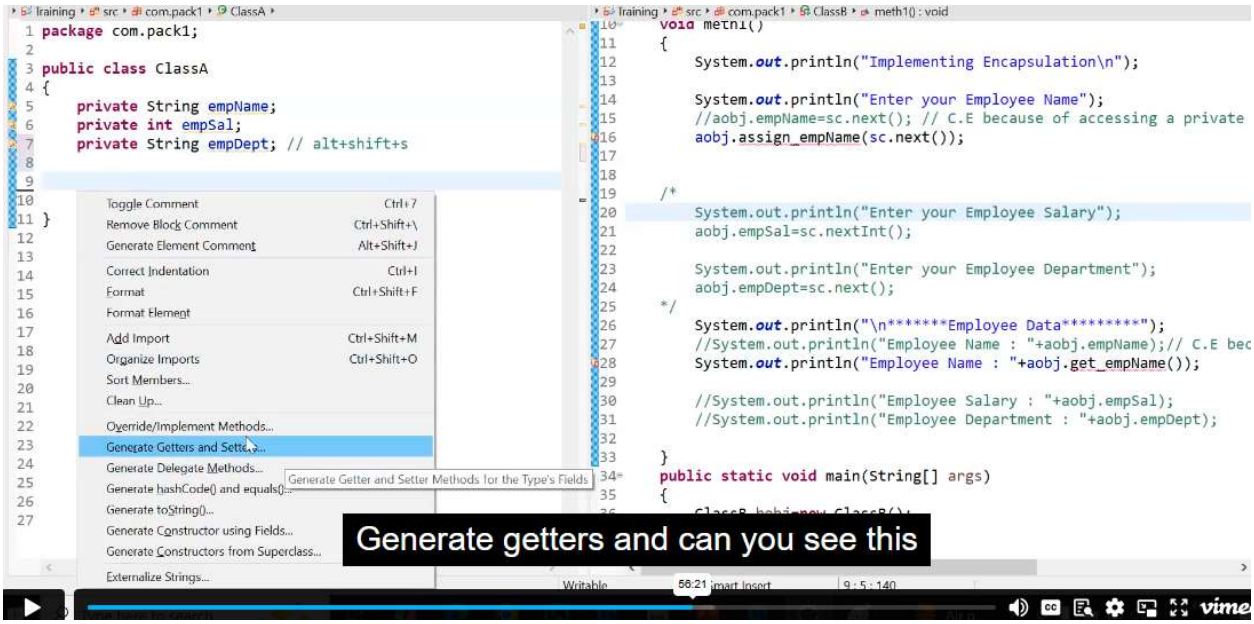
```
Implementing Encapsulation

Enter your Employee Name
Kishan

*******Employee Data*********
Employee Name : Kishan
```

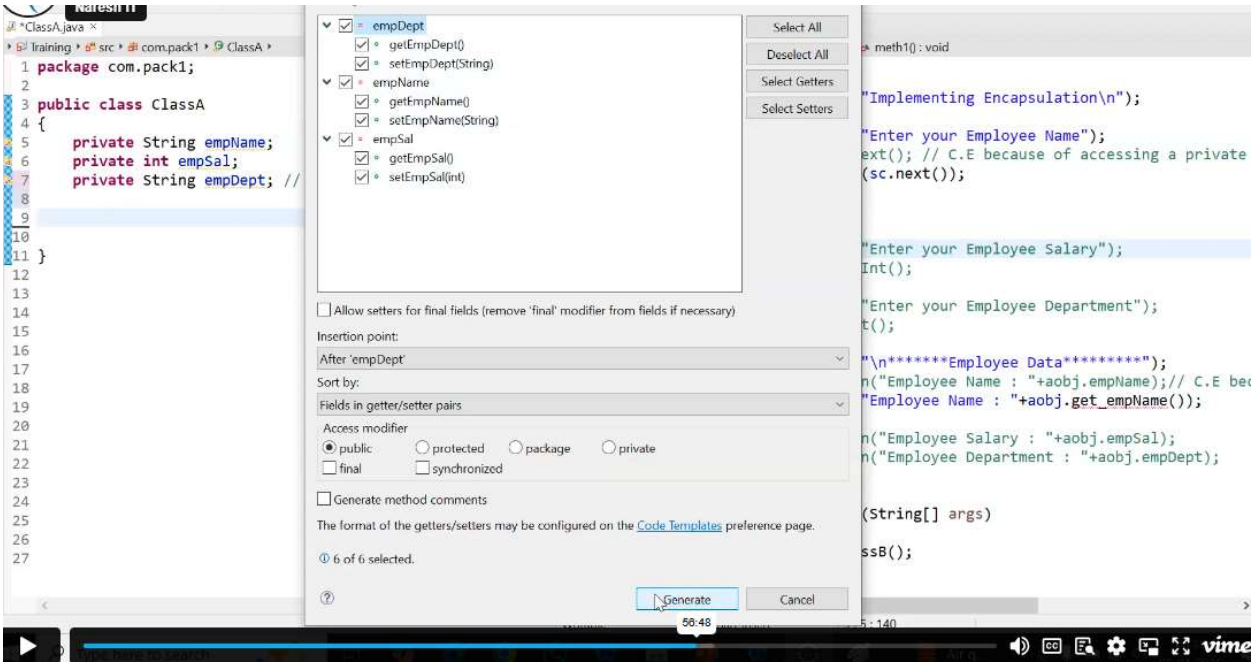
In java we have one class named bean class

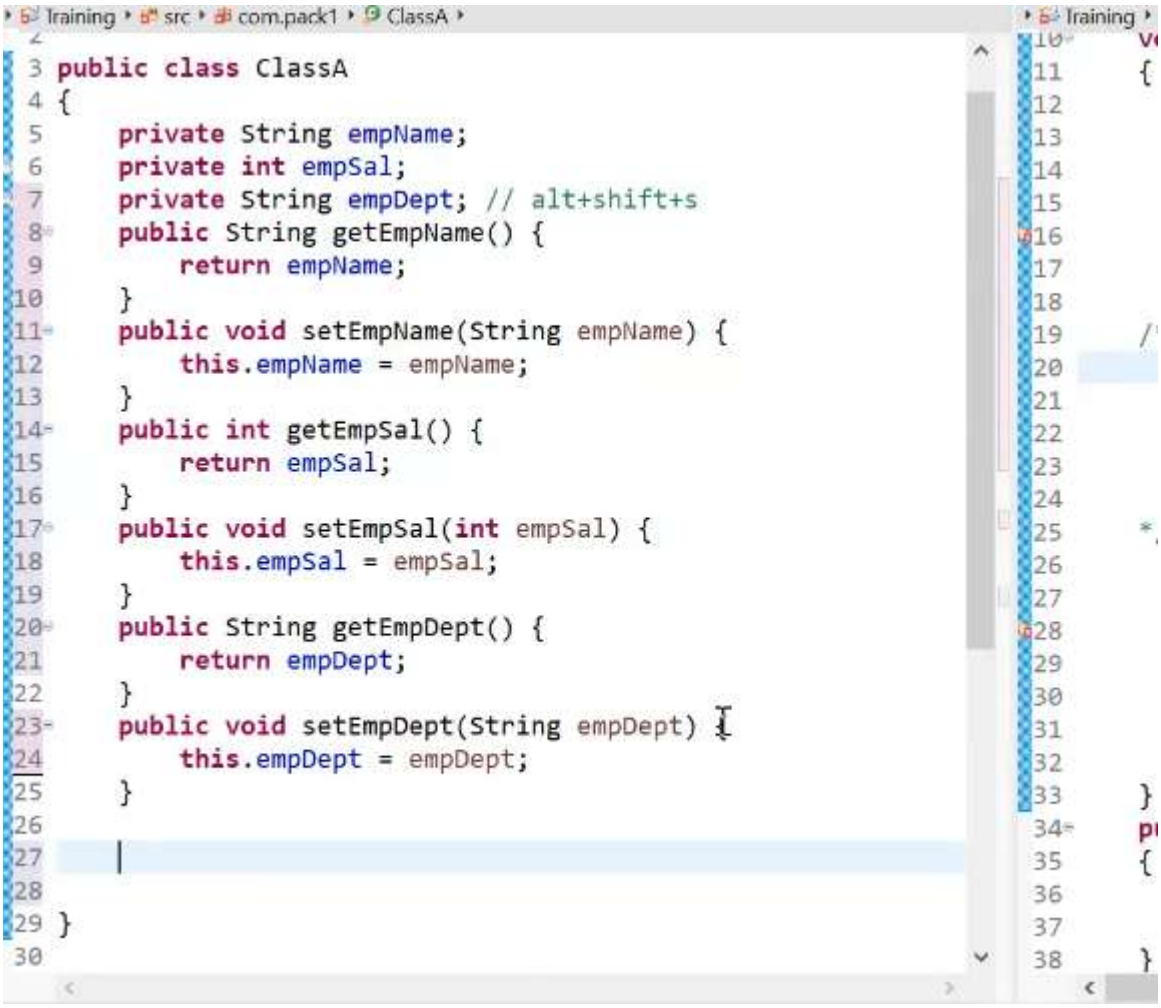
It has 4 properties

1. That class should implement serializable. (it should inherit serializable interface).
2. Make all those variables as private.
3. Use setters and getter methods

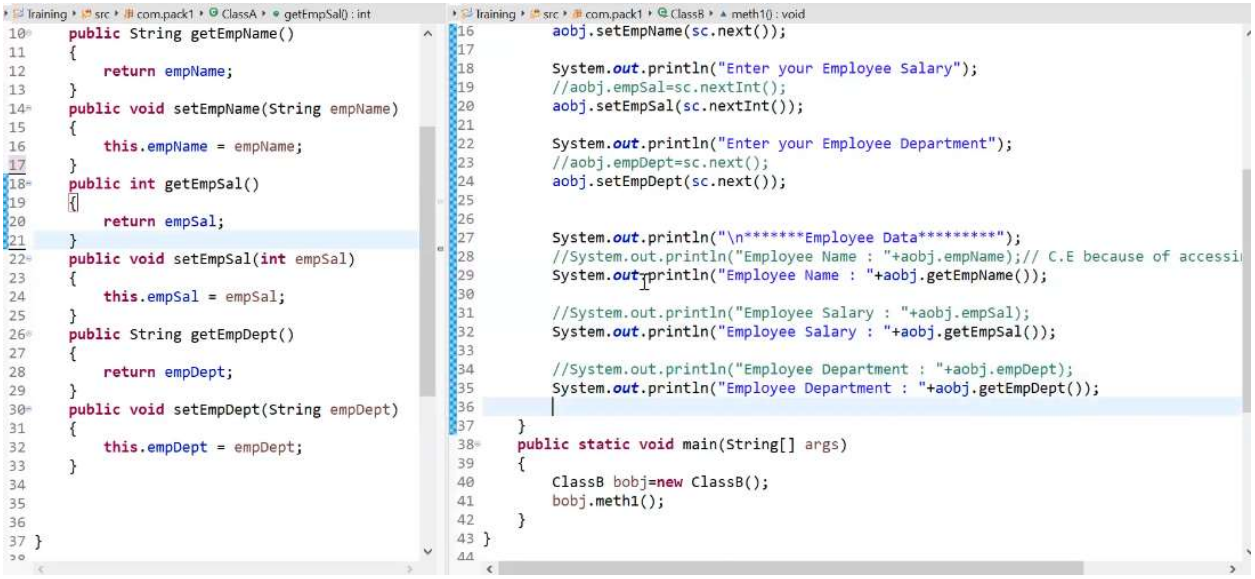
4. You need to have a public constructure





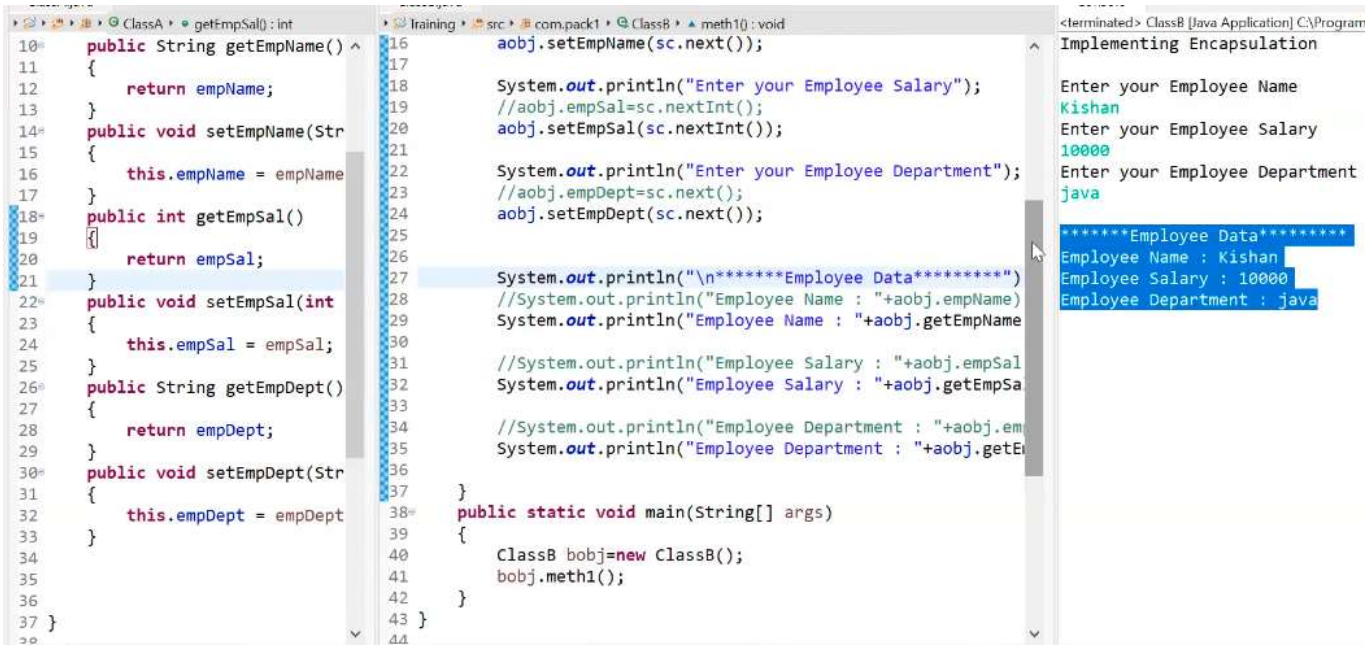


```
Training ▸ src ▸ com.pack1 ▸ ClassA ▸
 3 public class ClassA
 4 {
 5     private String empName;
 6     private int empSal;
 7     private String empDept; // alt+shift+s
 8     public String getEmpName() {
 9         return empName;
10     }
11     public void setEmpName(String empName) {
12         this.empName = empName;
13     }
14     public int getEmpSal() {
15         return empSal;
16     }
17     public void setEmpSal(int empSal) {
18         this.empSal = empSal;
19     }
20     public String getEmpDept() {
21         return empDept;
22     }
23     public void setEmpDept(String empDept) {
24         this.empDept = empDept;
25     }
26
27
28
29 }
30
```



```
Training ▸ src ▸ com.pack1 ▸ ClassA ▸ getEmpSal() : int
10     public String getEmpName()
11     {
12         return empName;
13     }
14     public void setEmpName(String empName)
15     {
16         this.empName = empName;
17     }
18     public int getEmpSal()
19     {
20         return empSal;
21     }
22     public void setEmpSal(int empSal)
23     {
24         this.empSal = empSal;
25     }
26     public String getEmpDept()
27     {
28         return empDept;
29     }
30     public void setEmpDept(String empDept)
31     {
32         this.empDept = empDept;
33     }
34
35
36
37 }
```

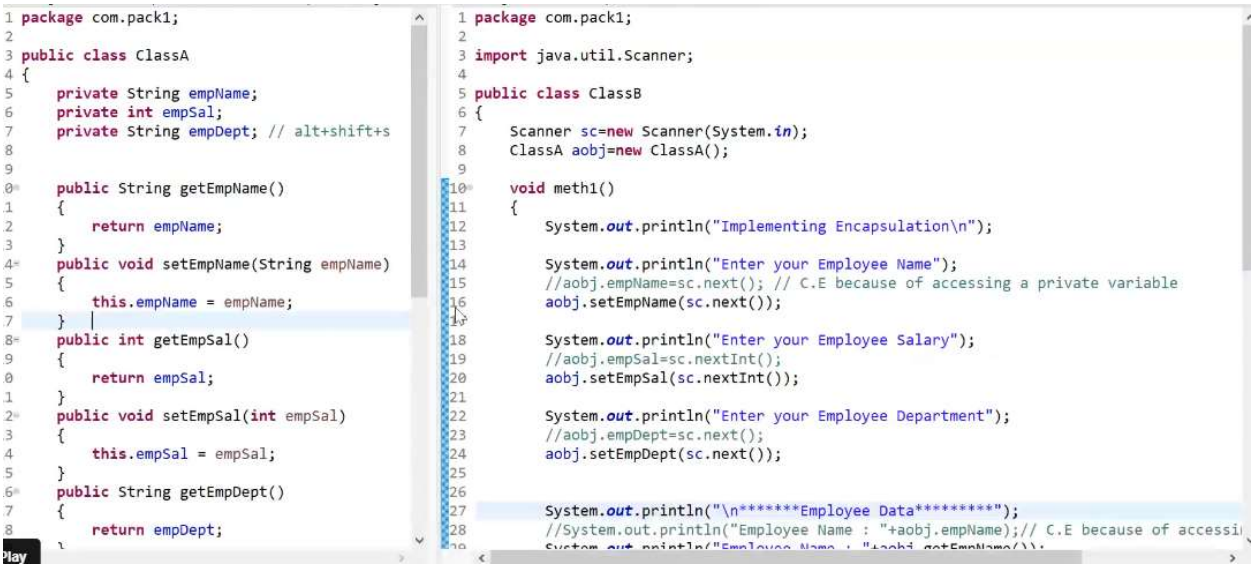
```
Training ▸ src ▸ com.pack1 ▸ ClassB ▸ meth1() : void
16         aobj.setEmpName(sc.next());
17
18         System.out.println("Enter your Employee Salary");
19         //aobj.empSal=sc.nextInt();
20         aobj.setEmpSal(sc.nextInt());
21
22         System.out.println("Enter your Employee Department");
23         //aobj.empDept=sc.next();
24         aobj.setEmpDept(sc.next());
25
26
27         System.out.println("\n*******Employee Data*********");
28         //System.out.println("Employee Name : "+aobj.empName);// C.E because of accessi
29         System.out.println("Employee Name : "+aobj.getEmpName());
30
31         //System.out.println("Employee Salary : "+aobj.empSal);
32         System.out.println("Employee Salary : "+aobj.getEmpSal());
33
34         //System.out.println("Employee Department : "+aobj.empDept);
35         System.out.println("Employee Department : "+aobj.getEmpDept());
36
37     }
38     public static void main(String[] args)
39     {
40         ClassB bobj=new ClassB();
41         bobj.meth1();
42     }
43 }
```



```
    public String getEmpName()
    {
        return empName;
    }
    public void setEmpName(Str
    {
        this.empName = empName
    }
    public int getEmpSal()
    {
        return empSal;
    }
    public void setEmpSal(int
    {
        this.empSal = empSal;
    }
    public String getEmpDept()
    {
        return empDept;
    }
    public void setEmpDept(Str
    {
        this.empDept = empDept
    }


}
```

```
        aobj.setEmpName(sc.next());

        System.out.println("Enter your Employee Salary");
        //aobj.empSal=sc.nextInt();
        aobj.setEmpSal(sc.nextInt());

        System.out.println("Enter your Employee Department");
        //aobj.empDept=sc.next();
        aobj.setEmpDept(sc.next());


        System.out.println("\n*******Employee Data*********")
        //System.out.println("Employee Name : "+aobj.empName)
        System.out.println("Employee Name : "+aobj.getEmpName

        //System.out.println("Employee Salary : "+aobj.empSal
        System.out.println("Employee Salary : "+aobj.getEmpSa

        //System.out.println("Employee Department : "+aobj.em
        System.out.println("Employee Department : "+aobj.getE

    }
    public static void main(String[] args)
    {
        ClassB bobj=new ClassB();
        bobj.meth1();
    }
}
```

```
<terminated> ClassB [Java Application] C:\Program
Implementing Encapsulation

Enter your Employee Name
Kishan
Enter your Employee Salary
10000
Enter your Employee Department
java

*******Employee Data*********
Employee Name : Kishan
Employee Salary : 10000
Employee Department : java
```

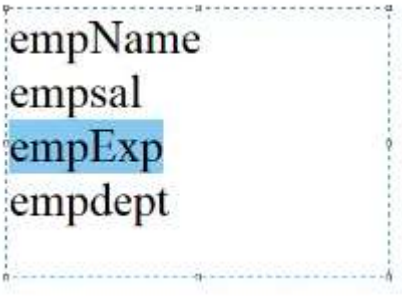


```
package com.pack1;

public class ClassA
{
    private String empName;
    private int empSal;
    private String empDept; // alt+shift+s


    public String getEmpName()
    {
        return empName;
    }
    public void setEmpName(String empName)
    {
        this.empName = empName;
    }
    public int getEmpSal()
    {
        return empSal;
    }
    public void setEmpSal(int empSal)
    {
        this.empSal = empSal;
    }
    public String getEmpDept()
    {
        return empDept;
    }
```

```
package com.pack1;

import java.util.Scanner;

public class ClassB
{
    Scanner sc=new Scanner(System.in);
    ClassA aobj=new ClassA();

    void meth1()
    {
        System.out.println("Implementing Encapsulation\n");

        System.out.println("Enter your Employee Name");
        //aobj.empName=sc.next(); // C.E because of accessing a private variable
        aobj.setEmpName(sc.next());

        System.out.println("Enter your Employee Salary");
        //aobj.empSal=sc.nextInt();
        aobj.setEmpSal(sc.nextInt());

        System.out.println("Enter your Employee Department");
        //aobj.empDept=sc.next();
        aobj.setEmpDept(sc.next());


        System.out.println("\n*******Employee Data*********");
        //System.out.println("Employee Name : "+aobj.empName);// C.E because of accessi
```

```java
package com.pack1;

import java.util.Scanner;

public class ClassB
{
    Scanner sc=new Scanner(System.in);
    ClassA aobj=new ClassA();

    void meth1()
    {
        System.out.println("Implementing Encapsulation\n");

        System.out.println("Enter your Employee Name");
        //aobj.empName=sc.next(); // C.E because of accessing a private variable
        aobj.setEmpName(sc.next());

        System.out.println("Enter your Employee Salary");
        //aobj.empSal=sc.nextInt();
        aobj.setEmpSal(sc.nextInt());

        System.out.println("Enter your Employee Department");
        //aobj.empDept=sc.next();
        aobj.setEmpDept(sc.next());


        System.out.println("\n*******Employee Data*********");
        //System.out.println("Employee Name : "+aobj.empName);// C.E because of accessing a private variable
        System.out.println("Employee Name : "+aobj.getEmpName());

        //System.out.println("Employee Salary : "+aobj.empSal);
        System.out.println("Employee Salary : "+aobj.getEmpSal());

        //System.out.println("Employee Department : "+aobj.empDept);
        System.out.println("Employee Department : "+aobj.getEmpDept());

    }
    public static void main(String[] args)
    {
        ClassB bobj=new ClassB();
        bobj.meth1();
    }
}
```

Assignment



Take 2 classes

If empExp is greater than 10 yrs or above increases the hike by 30%

If the exp is less than 10 yrs, the hike by 5%

Consider the above program as reference.