Session -2 (Day6)

1. **Create a class Vehicle and encapsulate the data members.**

Code:

```
public class Vehicle {
        private String name;
         private int id;

  public String getName() {
        return name;
   }
  public void setName(String name) {
       this.name = name;
  }
  public int getId() {
      return id;
  }
  public void setId(int id) {
      this.id = id;
 }
 }


  public class RunEncap {

     public static void main(String[] args) {
        Vehicle obj = new Vehicle();
        obj.setName("John");
        obj.setId(123);

        System.out.println("Name:" +obj.getName()+ " Id:"
+obj.getId());

    }
}
```

Output:

```
Name:John Id:123
```

2. **Create demo applications to illustrate different types of inheritance.**

Code:

Single Inheritance-

```
class Test  //parent class
{

     void sound()
    {
         System.out.println("the sound is good in old car");
     }

     static void enginePerformance()
    {
         System.out.println("40 yrs");
     }
}
public class SingleInheritance extends Test  //child class
{
    void design()
    {
        System.out.println("latest racing car model design
with good features");
    }
    public static void main(String[] args) {
        SingleInheritance si = new SingleInheritance();

        si.sound();
        Test.enginePerformance();
        si.design();
    }

}
```

Multilevel Inheritance-

```
class Test{                         //super parent class

     void sound() {
          System.out.println("the sound is good in old car");
     }

     void enginePerformance() {
          System.out.println("40 yrs");
     }
}

class Car extends Test{             //parent class

     void speed() {

          System.out.println("400 rpm");
     }
}

public class MultiLevelInheritance extends Car {
//child class
    void design()
    {
        System.out.println("latest racing car model design
with good features");
    }
  public static void main(String[] args)
  {
      MultiLevelInheritance mi= new MultiLevelInheritance();

      mi.sound();
      mi.design();
      mi.enginePerformance();
      mi.speed();

  }
}
```

Heirarchial Inheritance-

```
class Test{ //super parent

     void sound() {
          System.out.println("the sound is good in old car");
     }

     void enginePerformance() {
          System.out.println("40 yrs");
     }
}

class Car extends Test{  //child-1

     void speed() {

          System.out.println("400 rpm");
     }
}

public class HeirarchialInheritance extends Car{
    void design() {
        System.out.println("latest racing car model design
with good features");
    }
  public static void main(String[] args) {

      HeirarchialInheritance hi= new
HeirarchialInheritance();

      hi.sound();
      hi.design();
      hi.enginePerformance();
      hi.speed();

  }

}
```

3. Create one Animal class(superclass), Dog class(subclass) the same method in the subclass override the method in the superclass.

Code:

Output: