```
// Object Create :-> 1

 interface Vehicle {
    public void engine();
    public void breaks();
    public void wheels();
}

// Object Create :-> 2

class All_Vehicle {
    public static void sendVehicle(Vehicle vehicle) {

        vehicle.engine();
        vehicle. breaks();
        vehicle. wheels();
    }
}

// Object Create :-> 3

 class Bus implements  Vehicle {

        static {
            System.out.println("Bus class is loaded");
            System.out.println();
          }

          Bus() {
              System.out.println("Bus class object is created");
               System.out.println();
          }

           public void engine () {
                System.out.println("Bus engine capacity is 40 kmph");
                 System.out.println();
          }

            public void breaks () {
                System.out.println("Bus has two breaks");
                System.out.println();
          }

           public void wheels() {
           System.out.println("Bus will run on 4 wheels");
           System.out.println();
     }
}

// Object Create :-> 4


 class RedBus implements Vehicle {

     static {
            System.out.println("RedBus class is loaded");
            System.out.println();
          }

         RedBus() {
              System.out.println("RedBus class object is created");
               System.out.println();
          }

      public void engine() {
                System.out.println("RedBus engine capacity is 60 kmph");
                System.out.println();
          }


            public void breaks () {
                System.out.println("RedBus has also two breaks");
                System.out.println();
          }

         public void wheels() {
         System.out.println("RedBus will run on 8 wheels");
         System.out.println();
     }
}

// Object Create :-> 5

 class Volvo implements Vehicle {

    static {
        System.out.println("Volvo class is Loaded");
        System.out.println();
      }

          Volvo () {
              System.out.println("Volvo object is created");
```

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

            public void engine () {
                System.out.println("Volvo Engine capacity is 110 kmph");
                 System.out.println();
          }

            public void breaks () {
                System.out.println("Volvo has two breaks");
                 System.out.println();
          }

          public void wheels() {
          System.out.println("Volvo will run on also 4 wheels");
          System.out.println();
     }
}

// Object Create :-> 6

 class Bike implements Vehicle {

    static {
        System.out.println("Bike class is Loaded");
        System.out.println();
      }

          Bike () {
              System.out.println("Bike object is created");
              System.out.println();
          }

            public void engine () {
                System.out.println("Bike Engine capacity is 120 kmph");
                System.out.println();
          }

            public void breaks () {
                System.out.println("Bike has two breaks");
                System.out.println();
          }

          public void wheels() {
          System.out.println("Bike will run on 2 wheels");
          System.out.println();
    }
}



// Main method:->

import java.util.Scanner;

class VehicleDriver {
    public static void main(String[] args) throws ClassNotFoundException,
                                                     InstantiationException,
                                                       IllegalAccessException {
        All_Vehicle all_vehicle = new All_Vehicle();

        // Scanner API
            Scanner scn = new Scanner(System.in);

            System.out.println("Enter the Vehicle: ");
            String vehicleName = scn.nextLine();

            // Reflection API for loading & Instantiating

            Class cls = Class.forName(vehicleName);
            Object obj = cls.newInstance();

            if(obj instanceof Vehicle) {
                Vehicle vehicle = (Vehicle)obj;
                All_Vehicle.sendVehicle(vehicle);
    }

        else {
            System.out.println("Only pass Vehicle ");
        }
    }
}

// by shamshad
```