

Car Class

```
import java.io.*;

import java.util.* ;

import java.util.*;

class Car {
    int noOfGear;
    String color;
    // Write your constructor and printCarInfo method here.
    public Car(int noOfGear, String color){
        this.noOfGear = noOfGear;
        this.color = color;
    }
}

class RaceCar extends Car {
    int maxSpeed;
    // Write your constructor and printRaceCarInfo method here.

    public RaceCar(int noOfGear, String color, int maxSpeed)
    {
        super(noOfGear, color);
        this.maxSpeed = maxSpeed;
    }

    public void printInfo()
    {
        System.out.println("noOfGear: " + noOfGear);
        System.out.println("color: " + color);
        System.out.println("maxSpeed: " + maxSpeed);
    }
}

class Solution {
    public static void main(String args[]) {

        Scanner sc = new Scanner(System.in);
```

```

        int noOfGear = sc.nextInt();

        // [IGNORE]: Extra "end line" in previous line.
        sc.nextLine();

        String color = sc.nextLine();
        int maxSpeed = sc.nextInt();
        RaceCar raceCar = new RaceCar(noOfGear, color, maxSpeed);
        raceCar.printInfo();
    }
}

```

Multilevel Inheritance

```

import java.util.* ;
import java.io.*;

// Create the classes here
class GrandFather{
    //String grandFatherName = "Suresh";
    /*public void constructor1(String grandFatherName)
    {
        grandFatherName = grandFatherName;
    }*/
}

class Father extends GrandFather{
    //String fatherName = "Ramesh";

    /*public void constructor2(String grandFatherName, String
fatherName){
        super(grandFatherName);
        fatherName = fatherName;
    }*/
}

class Son extends Father{
    //String sonName = "Saurabh";

    /*public void constructor3(String grandFatherName, String
fatherName, String sonName){
        super(grandFatherName, fatherName);
        sonName = sonName;
    }*/
}

```

```

    }*/

    public void printName(String grandFatherName, String fatherName,
String sonName)
    {
        //super(grandFatherName, fatherName);
        System.out.println("sonname: " + sonName);
        System.out.println("fathername: " + fatherName);
        System.out.println("grandfather: " + grandFatherName);
    }
}

class Solution {

    public static void main(String args[]) {


        // Write your code here
        //GrandFather obj1 = new GrandFather();
        //Father obj2 = new Father();
        Son obj3 = new Son();
        obj3.printName("Suresh", "Ramesh", "Saurabh");
    }
}

```

How many types of inheritance in java?

Options:

- ☐ 2
- ☐ 1
- ☒ 3
- ☐ 5


 Correct Answer

What is the output of the following code?

```
class Base {  
  
    public void show() {  
        System.out.println("Base class show() method called");  
    }  
  
}  
  
class Derived extends Base {  
  
    public void show() {  
        System.out.println("Derived class show() method called");  
    }  
  
}  
  
public class Main {  
  
    public static void main(String[] args) {  
        Base b = new Derived();  
        b.show();  
    }  
  
}
```

Options:

- ☐ Base class show() method called
- ☒ Derived class show() method called
- ☐ Both of the above
- ☐ None of the above


 Correct Answer

What is the output of the following code?

```
class Base {  
  
    final public void show() {  
        System.out.println("Base class show() method called");  
    }  
}  
  
class Derived extends Base {  
  
    public void show() {  
        System.out.println("Derived class show() method called");  
    }  
}  
  
public class Main {  
  
    public static void main(String[] args) {  
        Base b = new Derived();  
        b.show();  
    }  
}
```

Options:

- ☐ Base class show() method called
- ☐ Derived class show() method called
- ☒ Compile time error
- ☐ Runtime error

 Correct Answer

What is the output of the following code?

```
class Base {  
  
    public void show() {  
        System.out.println("Base");  
    }  
  
}  
  
class Derived extends Base {  
  
    private void show() {  
        System.out.println("Derived");  
    }  
  
}  
  
public class Main {  
  
    public static void main(String[] args) {  
        Base b = new Derived();  
        b.show();  
    }  
  
}
```

Options:

- ☐ Base
- ☐ Derived
- ☒ Compile time error
- ☐ Runtime error

✓ Correct Answer