class A
no return and no argument

class B
no return and no argument

class Main

override
solve problem

```
import java.util.Scanner;
class A{
    void disp(){
        System.out.println("Class A");
    }
}
class B extends A{
    void disp(){
        super.disp();
        System.out.println("Class B");
    }
}
public class Polymorphism{
    public static void main(String[] args) {
        B b = new B();
        b.disp();
    }
}
```

Class A
Class B
—----------

class A
return and no argument

class B
return and no argument

class Main

override
solve problem

```
import java.util.Scanner;
```

```
class A{
    int disp(){
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter a: ");
        int a = sc.nextInt();
        System.out.println("class A");
        return a;
    }
}
class B extends A{
    int disp(){
        super.disp();
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter b: ");
        int b = sc.nextInt();
        System.out.println("Class B");
        return b;
    }
}
public class Polymorphism{
    public static void main(String[] args) {
        B b = new B();
        System.out.println(b.disp());
    }
}
```

Enter a: 45
class A
Enter b: 50
Class B
50
==========
class A
no return and argument

class B
no return and argument

class Main

override
solve problem

```java
import java.util.Scanner;
class A{
    void disp(int a){
        System.out.println("class A");
        System.out.println(a);
    }
}
class B extends A{
    void disp(int b){
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter a: ");
        int a = sc.nextInt();
        super.disp(a);
        System.out.println("Class B");
        System.out.println(b);
    }
}
public class Polymorphism{
    public static void main(String[] args) {

        B b1 = new B();
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter b: ");
        int b = sc.nextInt();
        b1.disp(b);

    }
}
```

Enter a: 30
class A
30
Class B
50
==========
class A
return and argument

class B
return and argument

class Main

override
solve problem

```java
import java.util.Scanner;
class A{
    int disp(int a){
        System.out.println("class A");
        return a;
    }
}
class B extends A{
    int disp(int b){
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter a: ");
        int a = sc.nextInt();
        System.out.println(super.disp(a));
        System.out.println("Class B");
        return b;
    }
}
public class Polymorphism{
    public static void main(String[] args) {

        B b1 = new B();
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter b: ");
        int b = sc.nextInt();
        System.out.println(b1.disp(b));

    }
}
```

Enter b: 60
Enter a: 30
class A
30
Class B
60
==========
class A
covariant return type

class B
covariant return type

class Main

override
solve problem

```
import java.util.Scanner;
class A{
    A disp(){
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter a: ");
        int a = sc.nextInt();
        System.out.println("class A");
        System.out.println(a);
        return this;
    }
}
class B extends A{
    B disp(){
        super.disp();
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter a: ");
        int b = sc.nextInt();
        System.out.println("Class B");
        System.out.println(b);
        return this;
    }
}
public class Polymorphism{
    public static void main(String[] args) {

        B b1 = new B();
        b1.disp();
    }
}
```

Enter a: 50
class A
50
Enter a: 20
Class B

20
==========
class A
no return and no argument

class B
no return and argument

class Main

override
solve problem

```
import java.util.Scanner;
class A{
    void disp(int a){
        System.out.println("class A");
        System.out.println(a);
    }
}
class B extends A{
    void disp(int b){
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter a: ");
        int a = sc.nextInt();
        super.disp(a);
        System.out.println("Class B");
        System.out.println(b);
    }
}
public class Polymorphism{
    public static void main(String[] args) {

        B b1 = new B();
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter b: ");
        int b = sc.nextInt();
        b1.disp(b);

    }
}
```

Enter b: 90

Enter a: 60
class A
60
Class B
90
==========
class A
no return and argument

class B
no return and no argument

class Main

override
solve problem

```
import java.util.Scanner;
class A{
    void disp(int a){
        System.out.println("class A");
        System.out.println(a);
    }
}
class B extends A{
    void disp(int b){
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter a: ");
        int a = sc.nextInt();
        super.disp(a);
        System.out.println("Class B");
        System.out.println(b);
    }
}
public class Polymorphism{
    public static void main(String[] args) {

        B b1 = new B();
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter b: ");
        int b = sc.nextInt();
        b1.disp(b);
```

```
    }
}
```

Enter b: 10
Enter a: 30
class A
30
Class B
10
==========

class A
 return and no argument :covariant type

class B
 return and argument :covariant type

class Main

override
solve problem

```
import java.util.Scanner;
class A{
    A disp(){
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter a: ");
        int a = sc.nextInt();
        System.out.println("class A");
        System.out.println(a);
        return this;
    }
}
class B extends A{
    B disp(){
        super.disp();
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter a: ");
        int b = sc.nextInt();
        System.out.println("Class B");
        System.out.println(b);
        return this;
    }
}
```

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

        B b1 = new B();
        b1.disp();
    }
}
```

class A
500
Enter a: 600
Class B
600
==========

overriding :single level

```
class A{
    int calculation(int a,int b){
     return a+b;
    }
 }
 class B extends A{
     int calculation(int a,int b,int c){
         System.out.println(super.calculation(50, 30));
         return a+b+c;
     }
 }
public class Polymorphism{
    public static void main(String[] args) {

        B b1 = new B();
        System.out.println(b1.calculation(50,40,70));
    }
}
```

80
160
==========

overriding :multilevel level

```
class A{
    int calculation(int a,int b){
     return a+b;
    }
 }
 class B extends A{
```

```
    int calculation(int a,int b,int c){
        System.out.println(super.calculation(80, 80));
        return a+b+c;
    }
}
class C extends B{
    int calculation(int a,int b,int c,int d){
        System.out.println(super.calculation(30, 30,30));
        return a+b+c+d;
    }
}
public class Polymorphism{
   public static void main(String[] args) {
       C c1 = new C();
       System.out.println(c1.calculation(50,50,50,50));
   }
}
```

160
90
200
==========

overriding :heirarichal level

```
class A{
   int calculation(int a,int b){
    return a+b;
   }
}
class B extends A{
    int calculation(int a,int b,int c){
        System.out.println(super.calculation(20, 60));
        return a+b+c;
    }
}
class C extends A{
    int calculation(int a,int b,int c,int d){
        return a+b+c+d;
    }
}
public class Polymorphism{
   public static void main(String[] args) {
```

```
        B b1 = new B();
        System.out.println(b1.calculation(60, 60, 60));
        C c1 = new C();
        System.out.println(c1.calculation(50,50,70,50));
    }
}
```

80
180
220
==========
overriding :multiple level
==========
overriding :hybrid level
==========
class A
no return and no argument
return and argument
no return and argument
return and no argument
covariant return type

class B
no return and no argument
return and argument
no return and argument
return and no argument
covariant return type

class Main
access

override
solve problem

```
class A{
    void sayHello(){
      System.out.println("Hello!...class A");
    }
    int add(int a,int b){
      return a+b;
    }
    void sub(int a,int b){
      System.out.println("Subtraction: "+(a-b));
    }
```

```
    int multi(){
      int a = 20,b=3;
      return a*b;
    }
    A divide(int a,int b){
      System.out.println("Division: "+(a/b));
      System.out.println("-----------------");
      return this;
    }
  }
  class B extends A{
      void sayHello(){
          super.sayHello();
          System.out.println("Add: "+super.add(70, 80));
          super.sub(90, 70);
          System.out.println("Multiply: "+super.multi());
          super.divide(100, 5);
          System.out.println("Hello!...class B");
        }
        int add(int a,int b){
          return a+b;
        }
        void sub(int a,int b){
          System.out.println("Subtraction: "+(a-b));
        }
        int multi(){
          int a = 30,b=3;
          return a*b;
        }
        B divide(int a,int b){
          System.out.println("Division: "+(a/b));
          return this;
        }
  }
public class Polymorphism{
    public static void main(String[] args) {
        B b1 = new B();
        b1.sayHello();
        System.out.println("Add: "+b1.add(50, 100));
        b1.sub(90, 30);
```

```
        System.out.println("Multiply: "+b1.multi());
        b1.divide(100, 2);

    }
}
```

Hello!...class A
Add: 150
Subtraction: 20
Multiply: 60
Division: 20

Hello!...class B
Add: 150
Subtraction: 60
Multiply: 90
Division: 50
==========
constant values :

class College
instance variable :
- name
- address
- contact number
- staff
- library
- email

display()
print all

class Student
instance variable :
- name
- address
- contact number
- fname
- mname
- idnumber
- email

display()
print all

class Main

access

```
class College{
    String name;
    String address;
      long contact;
      int staff;
      String library;
      String email;
    void display(String name,String address,long contact,int staff,String
library,String email){
      this.name=name;
      this.address=address;
      this.contact=contact;
      this.staff=staff;
      this.library=library;
      this.email=email;
      System.out.println("!--College--!");
      System.out.println("Name: "+name);
      System.out.println("Address: "+address);
      System.out.println("Contact: "+contact);
      System.out.println("Staff: "+staff);
      System.out.println("Library: "+library);
      System.out.println("Email: "+email);
      System.out.println("---------X---------");
    }
  }
  class Student extends College{
      String name;
          String address;
          long contact;
          String fname;
          String mname;
          int idnumber;
          String email;
      void display(String name,String address,long contact,String
fname,String mname,int idnumber,String email){

super.display("ABC","JODHPUR",32626522,20,"ABC","lokeshdeorajodhpur@gmail.
com" );
        this.name=name;
```

```
        this.address=address;
        this.contact=contact;
        this.fname=fname;
        this.mname=mname;
        this.idnumber=idnumber;
        this.email=email;
        System.out.println("!--Student--!");
        System.out.println("Name: "+name);
        System.out.println("Address: "+address);
        System.out.println("Contact: "+contact);
        System.out.println("Father's Name: "+fname);
        System.out.println("Mother's Name: "+mname);
        System.out.println("ID Number: "+idnumber);
        System.out.println("Email: "+email);
      }
  }
public class Polymorphism{
    public static void main(String[] args) {
        Student s = new Student();
        s.display("DEF", "JODHPUR",
1232656654,"PQRS","TUV",203,"LOKESH@gmail.com");
      }
  }
```

!--College--!
Name: ABC
Address: JODHPUR
Contact: 32626522
Staff: 20
Library: ABC
Email: lokeshdeorajodhpur@gmail.com
---------X---------
!--Student--!
Name: DEF
Address: JODHPUR
Contact: 1232656654
Father's Name: PQRS
Mother's Name: TUV
ID Number: 203
Email: LOKESH@gmail.com
==========
user input values :

class College
instance variable :
- name
- address
- contact number
- staff
- library
- email

display()
print all

class Student
instance variable :
- name
- address
- contact number
- fname
- mname
- idnumber
- email

display()
print all

class Main
access