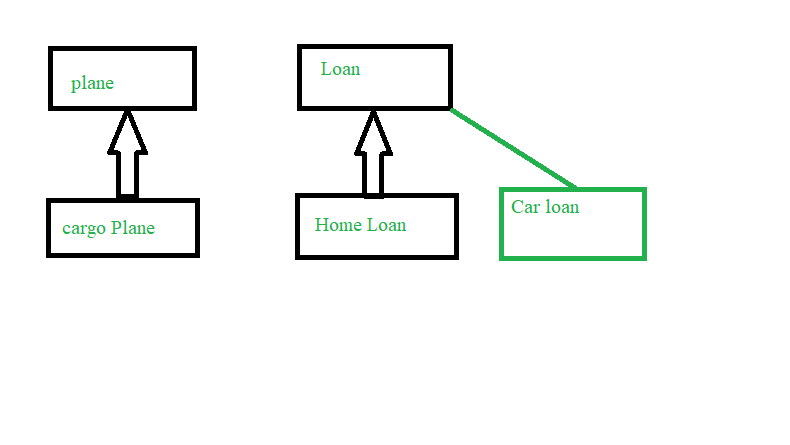
Inheritance:

IS-A relationship



Consider a plane which has many things and each plane will have some same set of Functionalities. We can write these functionalities in the Plane Class and write the other functions in the class extended from it.

Access Modifiers:

Public

Protected

Default

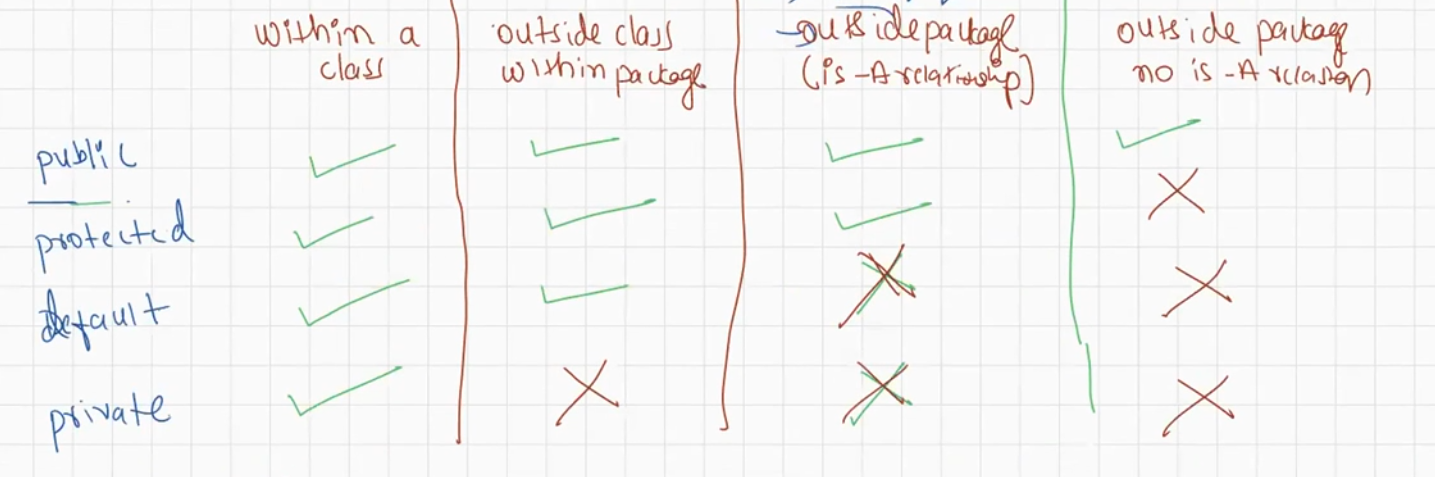
Private

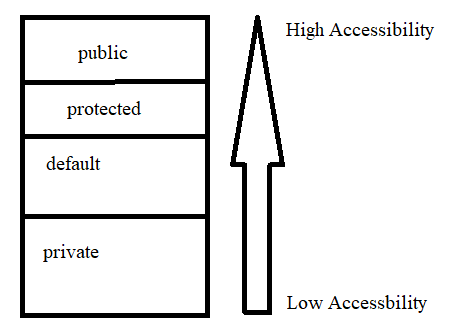
They are applicable for class,method,constructor,variable.

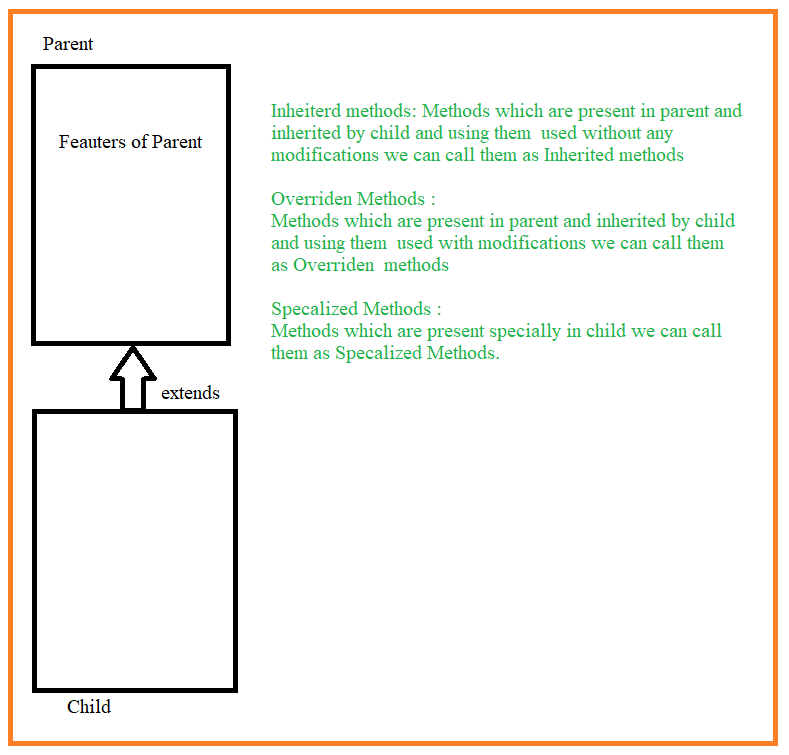
If we don’t specify any access specifier it will be by default access specifier.

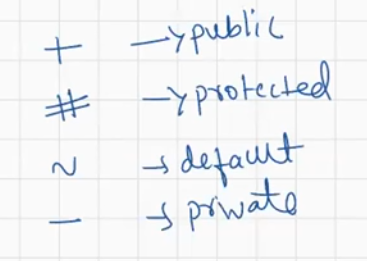
Project will be divided into multiple folders.

We can consider each folder as package.









A white board with blue writing

Description automatically generated

Once we extend the parent class the methods will be present in the child.

In the above example:

Inherited Methods:

land(),takeOff().

Overriden Methods:

fly()

specialized methods:

carryGoods()

carrypassengers()

A screenshot of a computer

Description automatically generated

From the screen shot we could see we have land(),takeOff() methods as well in the Cargo Plane Class.

Object is the parent of all the classes.

In Object class equals will compare the references.

**class** Plane

{

**public** **void** takeOff()

{

System.***out***.println("Plane Taking Off");

}

**public** **void** land()

{

System.***out***.println("Plane Landing");

}

**public** **void** fly()

{

System.***out***.println("Plane Flying");

}

}

**class** CargoPlane **extends** Plane

{

**public** **void** fly()

{

System.***out***.println("Cargo Plane Flying");

}

**public** **void** carryGoods()

{

System.***out***.println("Cargo Plane Carry Goods Method");

}

}

**class** PassengerPlane **extends** Plane

{

**public** **void** fly()

{

System.***out***.println("Passenger Plane Flying");

}

**public** **void** carryPassenger()

{

System.***out***.println("Passenger plane carrying ");

}

}

**public** **class** LaunchPlane {

**public** **static** **void** main(String[] args) {

CargoPlane cp=**new** CargoPlane();

cp.fly();//Overriden Method will be called

cp.carryGoods();//Specalized Method

cp.takeOff();//inherited Method

cp.land();//Inherited Method

PassengerPlane pp=**new** PassengerPlane();

pp.fly();

pp.carryPassenger();

pp.takeOff();

pp.land();

}

}

Output:

Cargo Plane Flying

Cargo Plane Carry Goods Method

Plane Taking Off

Plane Landing

Passenger Plane Flying

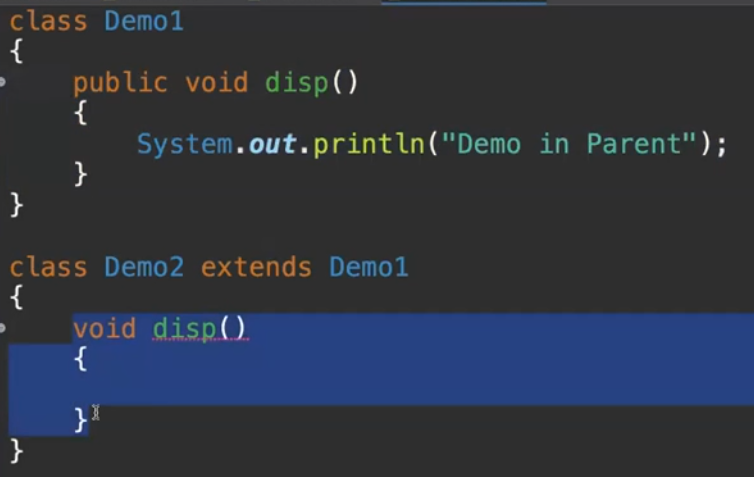
Passenger plane carrying

Plane Taking Off

Plane Landing

Rules to Override Method:

Rule 1: We cannot reduce the visibility of Overriden method.



Above is not allowed

But we can increase the visibility.

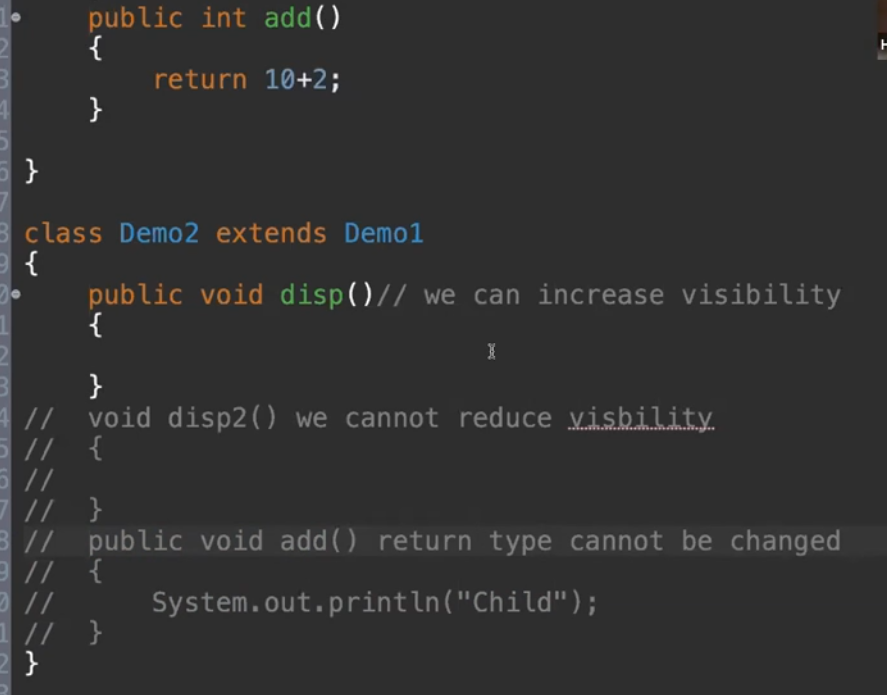
A screen shot of a computer

Description automatically generatedAllowed

Mostly all in-built methods are by default public.

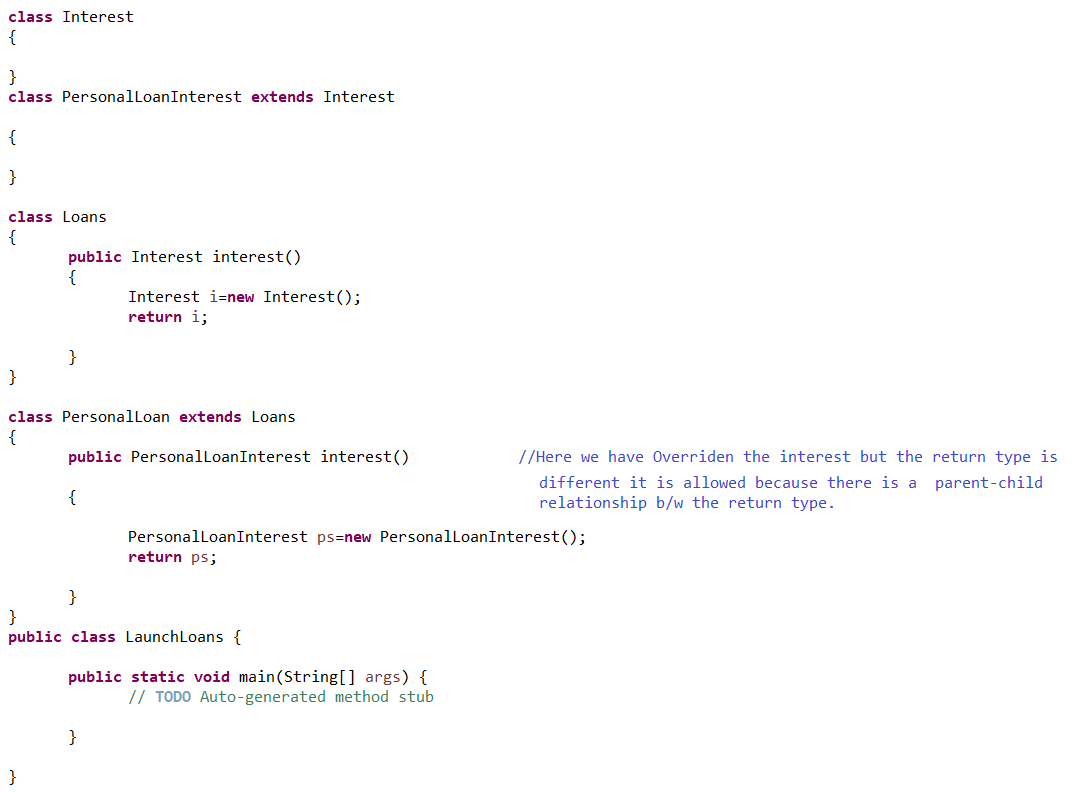
Rule 2:

Return type of the Over-riden method must be same as Overriding method(Parent Method)



Rule 3:

Return type of the Over-riden method can be different as Overriding method(Parent Method) if it is Co-variant return type(return type Is-A relationship).



We can return the Object as well.

Rule 4: Parameters of the Overridden method must be same as that of parent else it it will considered as Specialized method considering Method overloading.