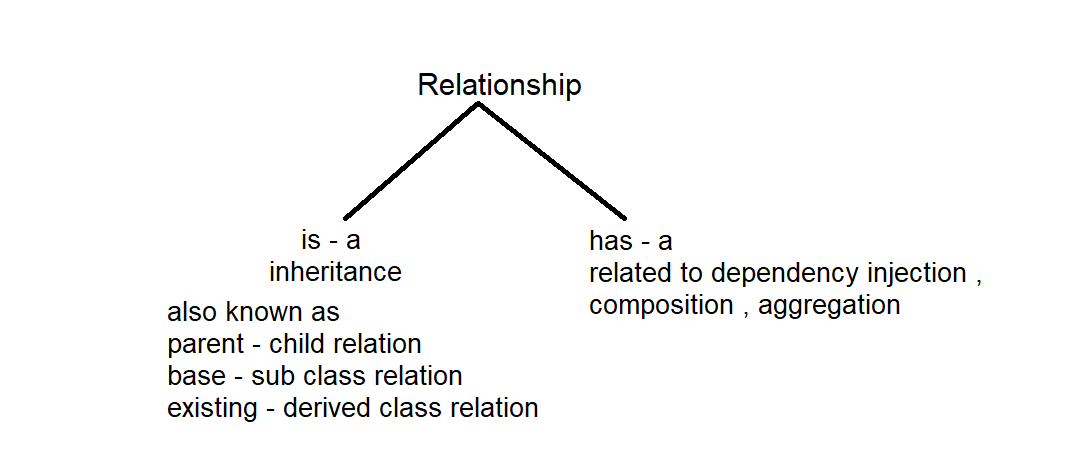
we can use code of one class in another class (code reusability) when both the classes are in relationship.



Inheritance: process of one class acquiring the properties and behaviour of another class is called inheritance

Relationship is created by using the keyword extends.

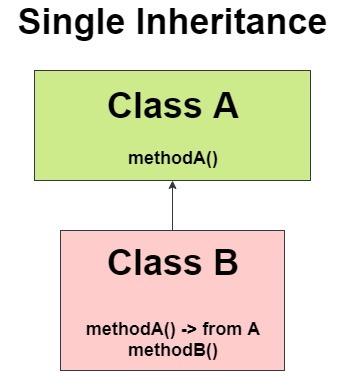
Eg: Inheritance\_Demo\_Eg1

Class before extends keyword is called child, sub, derived class.

The class after extends keyword is called parent, base, existing class

Keypoints :

1. Single level inheritance is allowed. (one class extends another class)

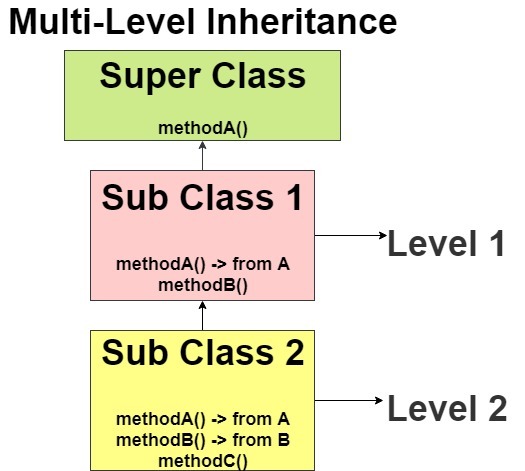


Eg: Inheritance\_Demo\_Eg1

1. Object class is parent of all the classes by default.

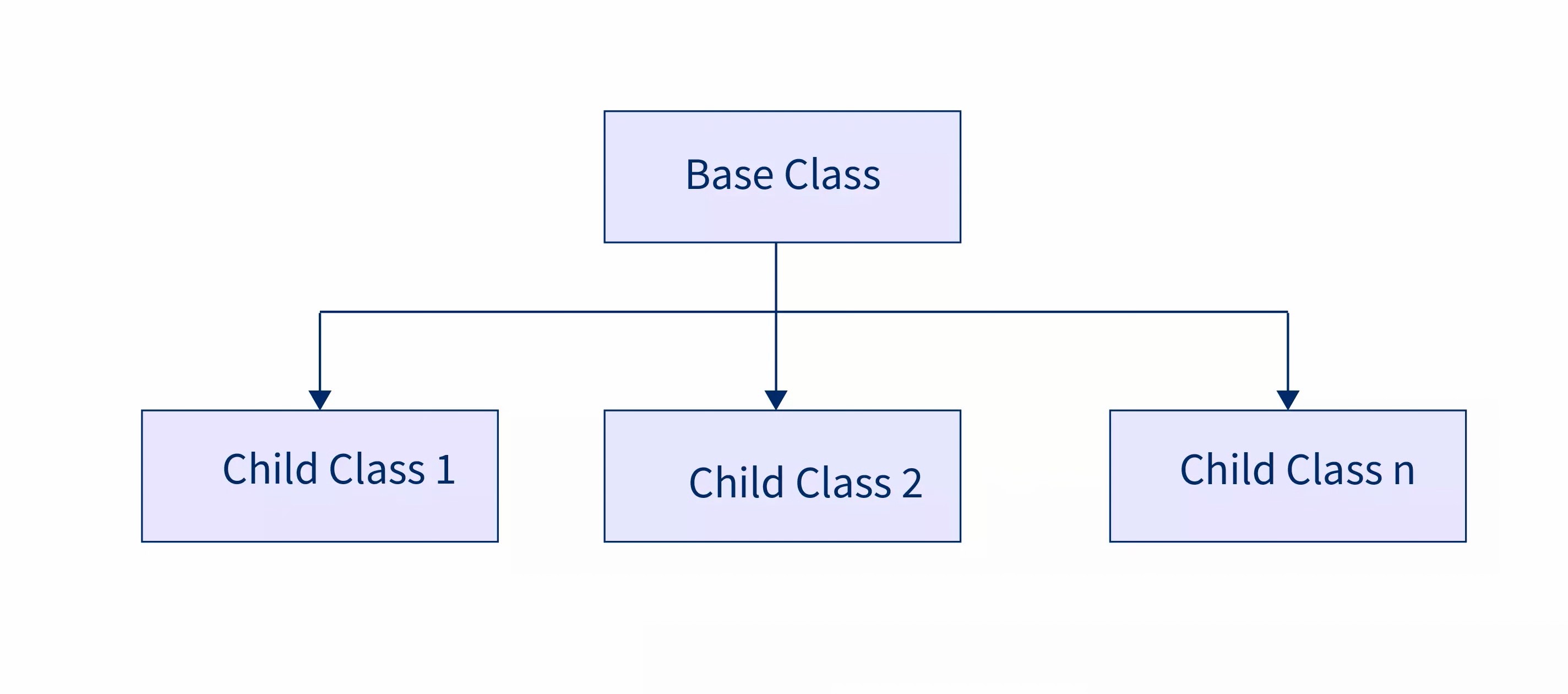
All the properties and methods of Object class can be used by our user defined classes, since they sub-classes of Object class.

1. Multilevel inheritance



Eg: Multilevel\_Inheritance\_Eg2

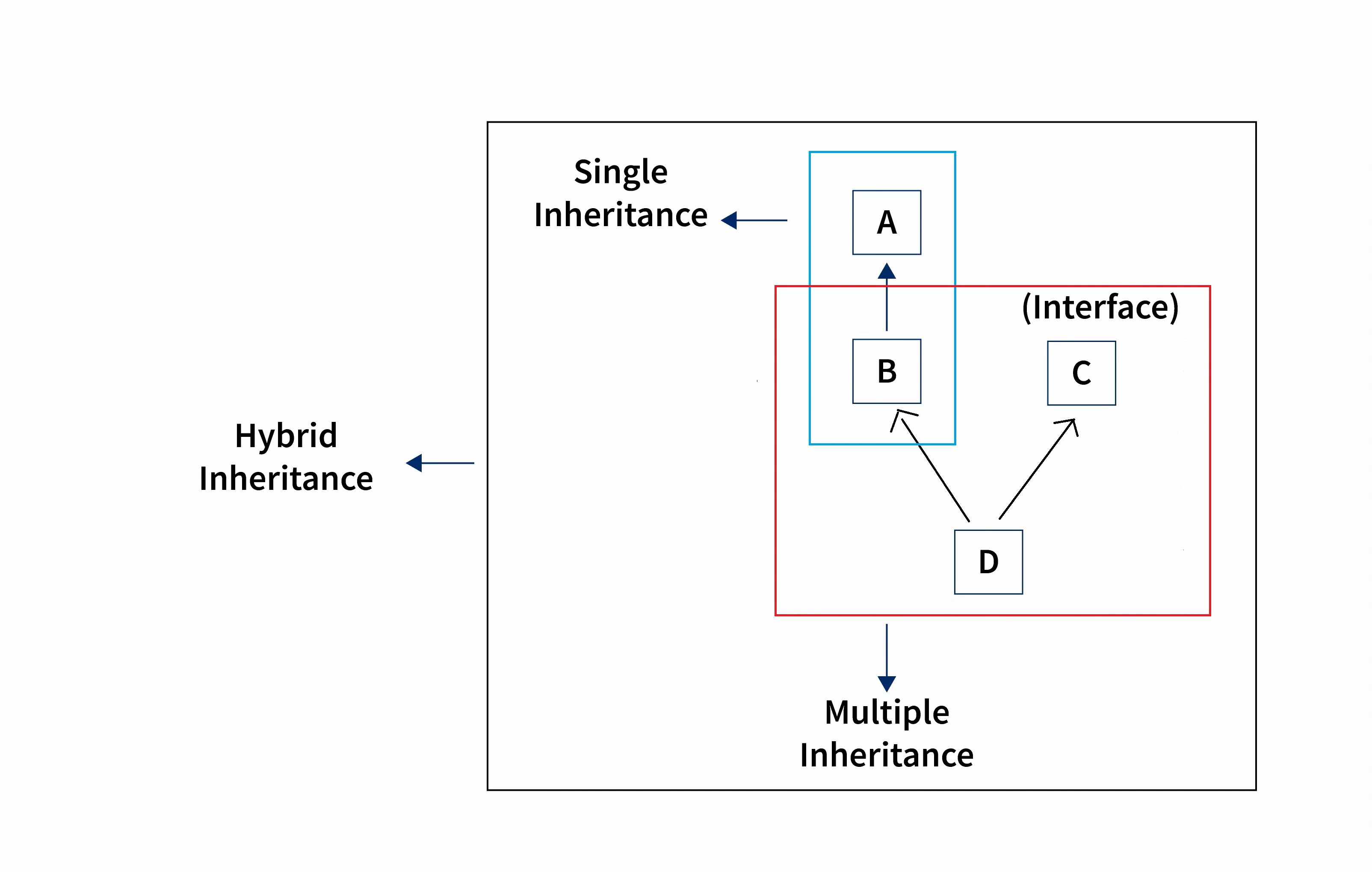
1. One parent/ base can have any no of child/sub classes

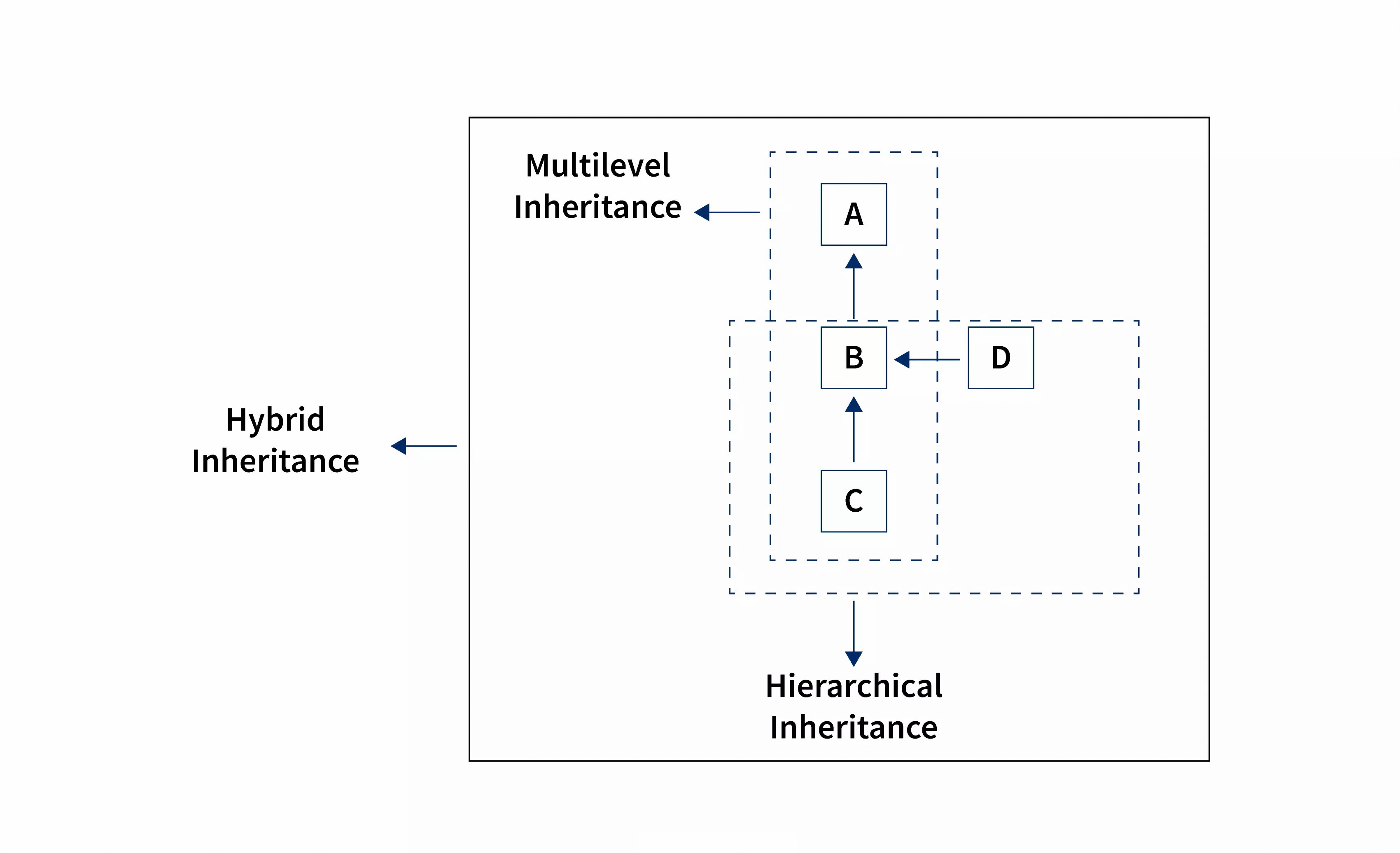


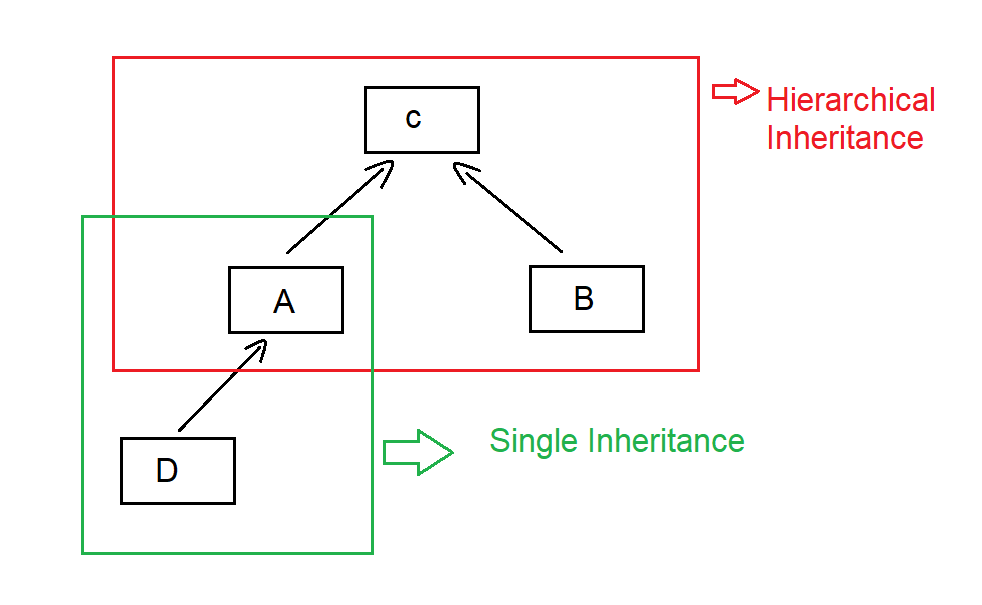
Eg: Hierarchical\_Inheritance\_Eg3

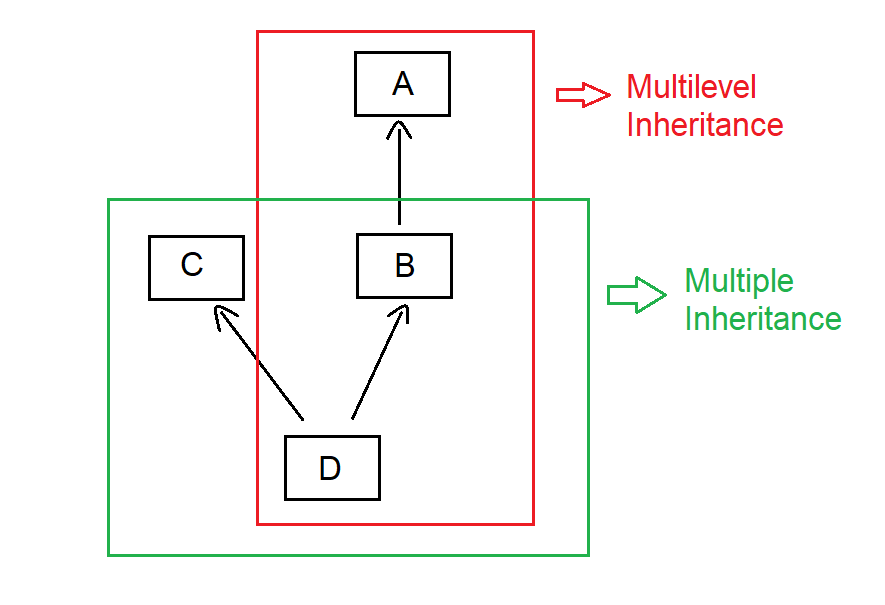
1. Hybrid inheritance: The term hybrid means made of more than one thing. Hybrid inheritance is combination / made of two (or) more types of inheritance

Different combinations of hybrid inheritance





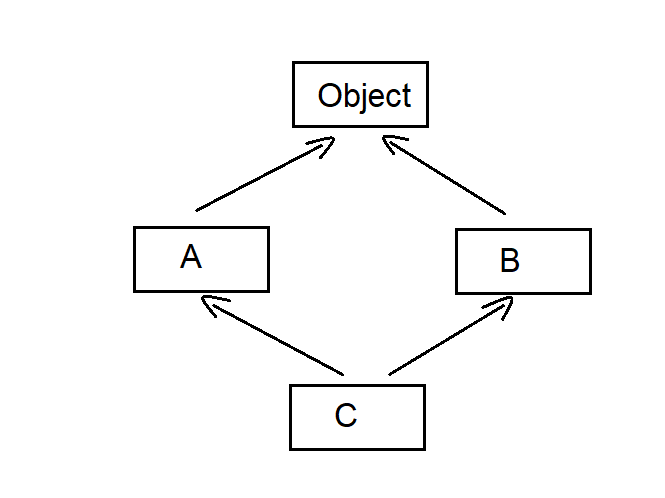




Single and multilevel inheritance is not possible. try to all the possible combinations on paper for better understanding.

1. Multiple inheritance is not allowed in java. because there might be a ambiguity that two parent classes have variables (or) methods with same name that raises the problem that which one to be picked for child.

Other names for multiple inheritance is diamond shape problem, compile time ambiguity is also same.



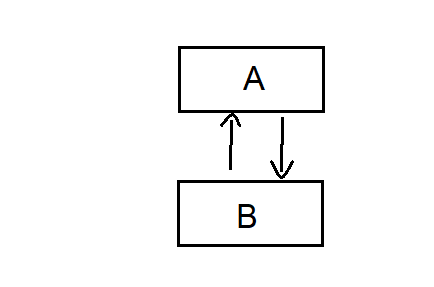
Since class A and B doesn’t have any parent by default object class is the parent for both class A and B.

Object class is present in java.lang package. Every class in Java is directly or indirectly derived from the Object class. If a class does not extend any other class then it is a direct child class of Object and if extends another class then it is indirectly derived. Therefore the Object class methods are available to all Java classes. Hence Object class acts as a root of the inheritance hierarchy in any Java Program.

Since the super class contains properties and methods of its own class and Object class, all of them can be used by the child classes defined with extends keyword.

Multiple inheritance is allowed in java only with interfaces, it is not possible through classes.

1. Cyclic inheritance is not allowed in java



Means either A nor B must be the parent. both cannot be the parents.

1. private members of a class doesn’t participate in the inheritance. it is done to preserve the encapsulation.

Making properties, methods private preserves encapsulation. because outside the class there is no direct access. Access is allowed through getter and setter because it is not direct.

Eg: Inheritance\_private

To access private methods in child class:

With the help of the java.lang.Class class and java.lang.reflect.Method class we can call private method from any other class.

1. Required method of Class Class

public Method getRequiredMethod() : it stores our private method in the form of object . And that object is collected by reference of Method class.

1. Required methods of Method Class

public void setAccessible( boolean status ) : if the boolean status is true . This method avoids java language checks. if boolean status is false it follows java language checks.

public Object invoke (invoke method, Object.. args ): It invokes the underlying method represented by Method object, on the specified object with specified parameters ( specified user-defined object of private method class with specific parameters. )

All of these methods throws multiple exceptions . they are handled by Exception class .

Eg:Private\_Method\_Inheritance .

1. constructors will not participate in inheritance.

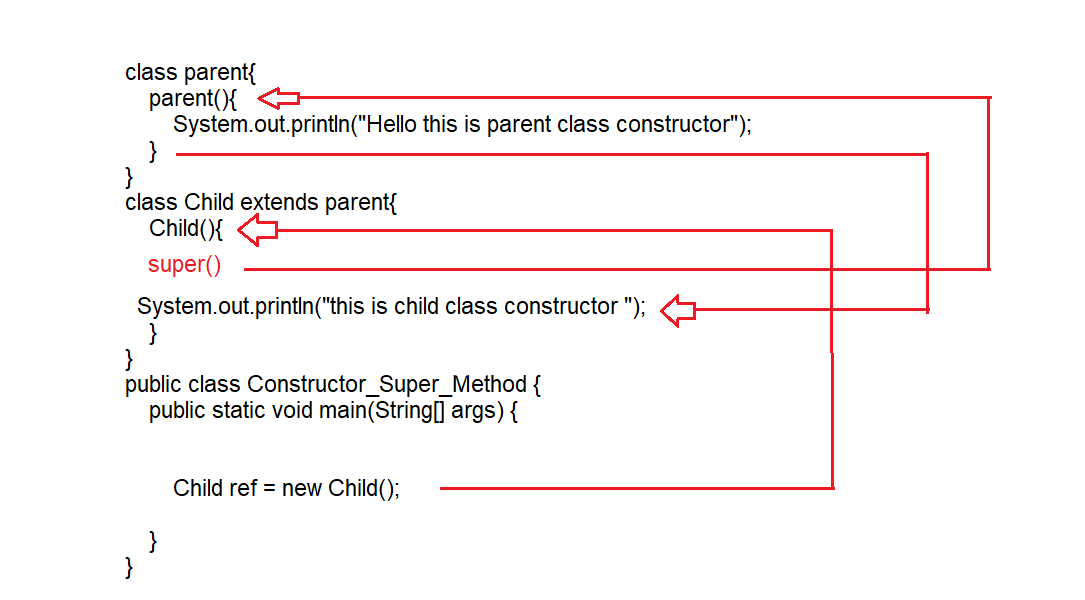
consider demo1 class extends demo2. And you create an object of

demo2 ref = new demo2()

constructor means when you had created an object, that time it is called. if it is getting inherited means you are creating object for demo2 and demo1 is executed.

However parent class constructor will be called because of super() (default method of constructor) call present in child class . it is not done because of inheritance.

Eg: Constructor\_Working\_During\_Inheritance\_Eg1

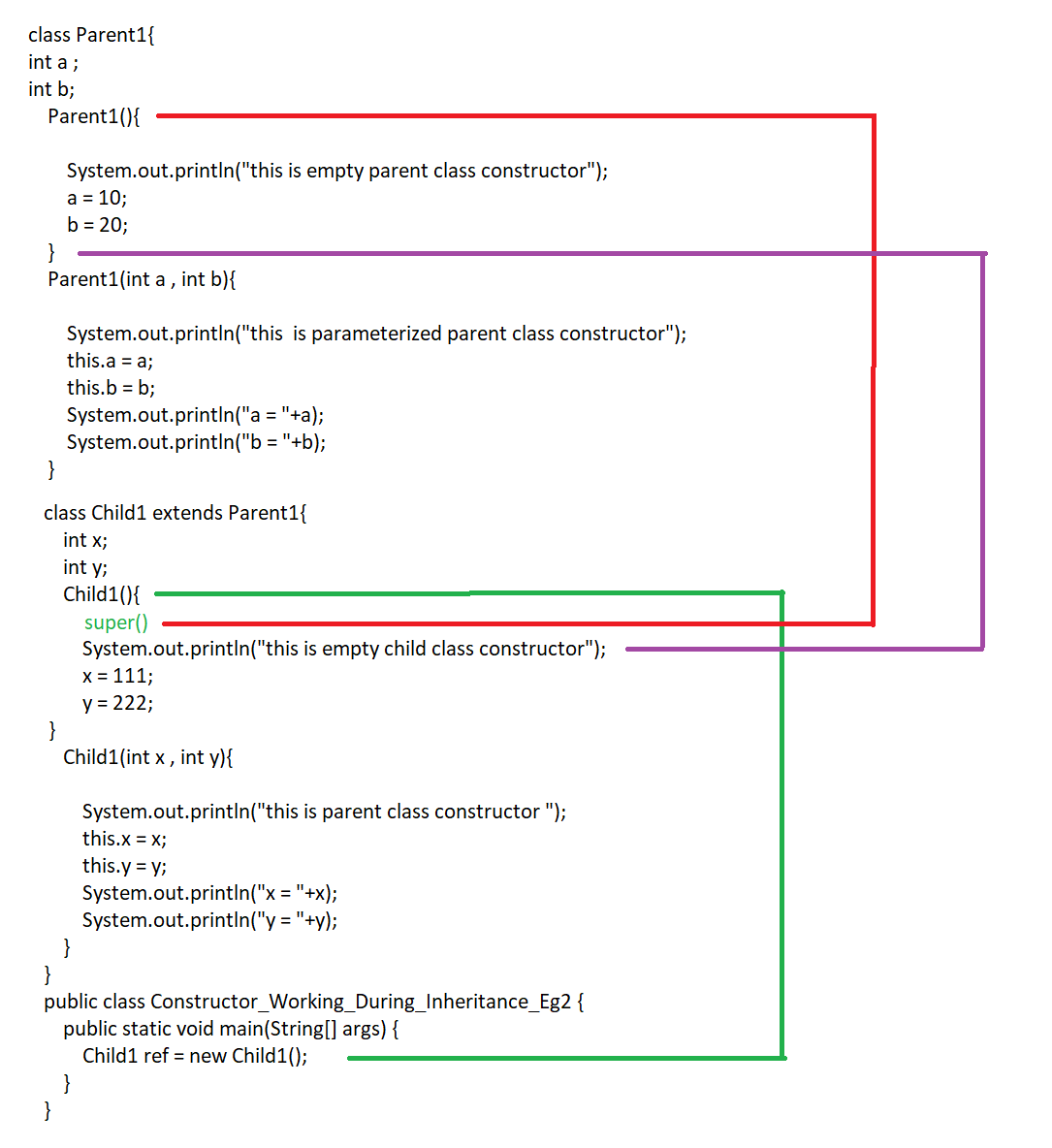


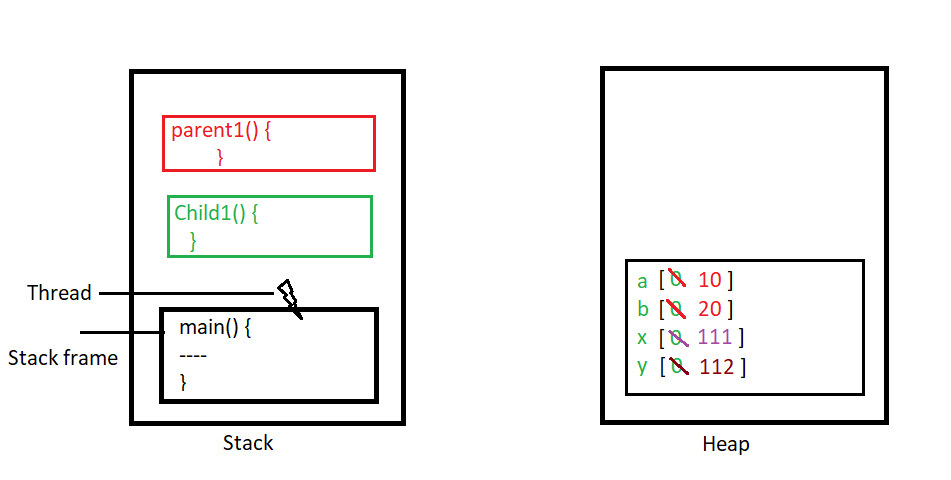
Explanation

Main method is executed line by line, when object is created control goes to Child class constructor it is executed. the 1st line of child class constructor will be super() by default . which calls parent class constructor, now control goes to parent class constructor, it will get executed. After its complete execution, control again comes to child class constructor to execute remaining code.

Note: Creating object of child class is equal to creating the object of the class

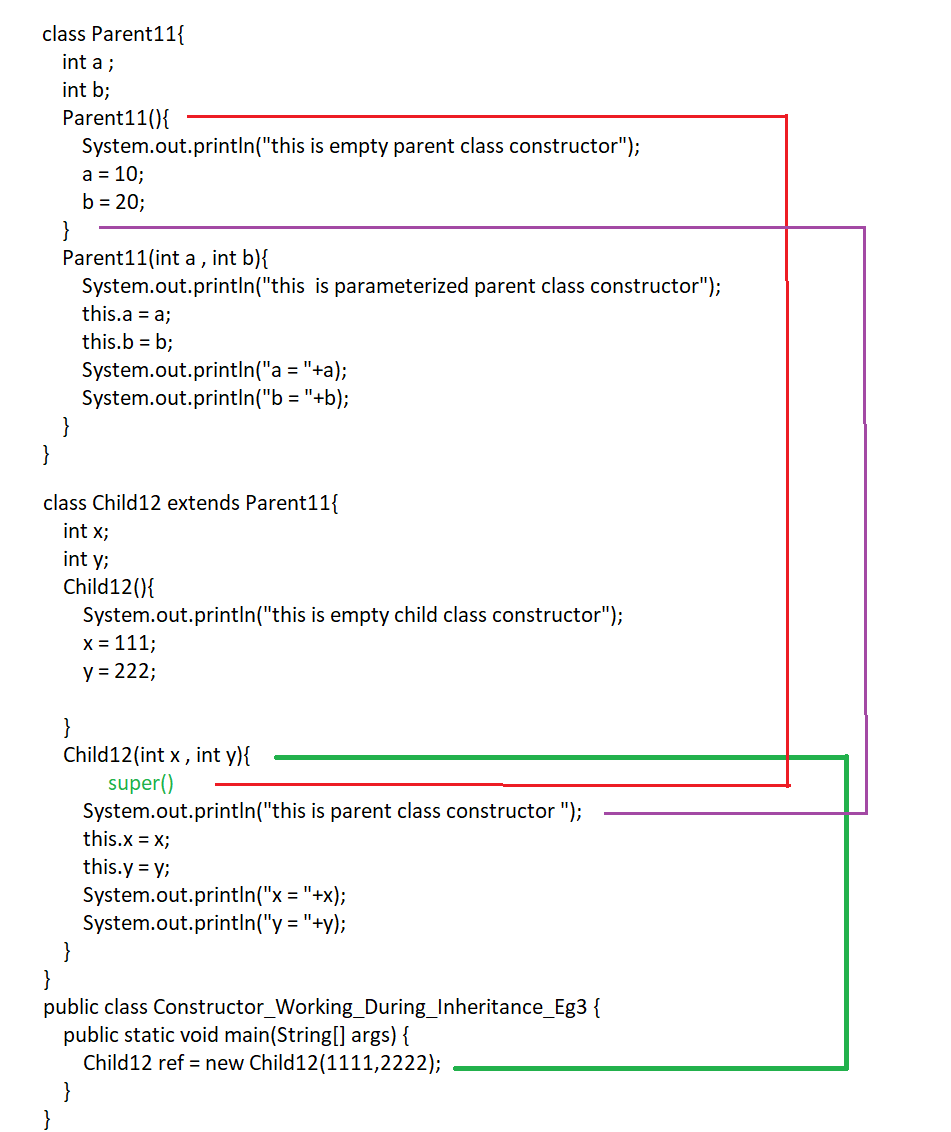
Eg : Constructor\_Working\_During\_Inheritance\_Eg2



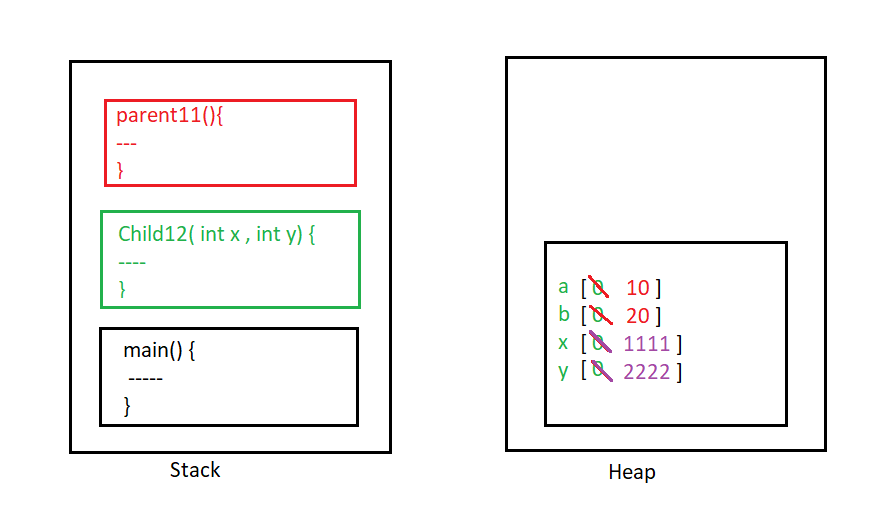


super() method is called by default by jvm .

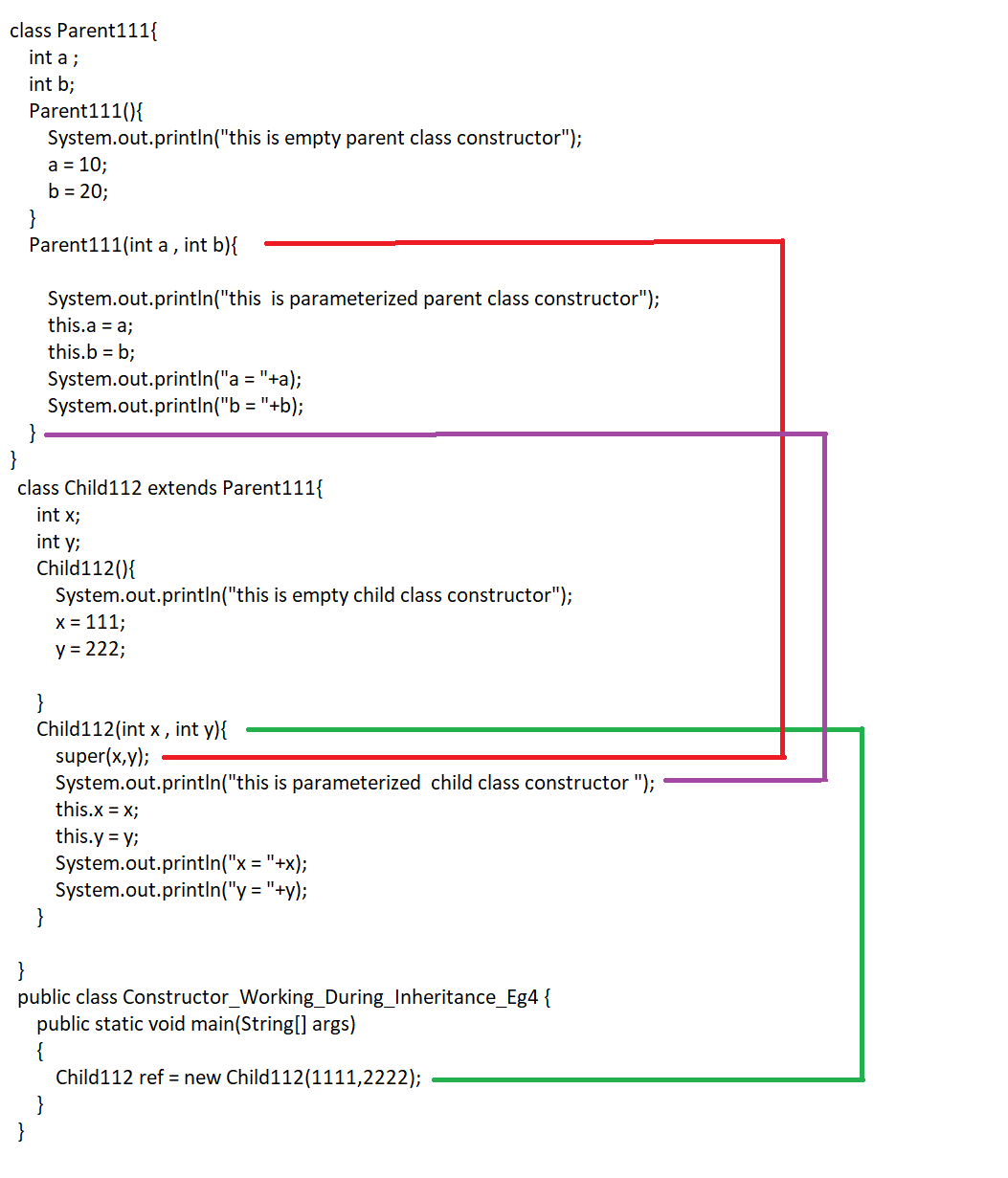
Eg : Constructor\_Working\_During\_Inheritance\_Eg3



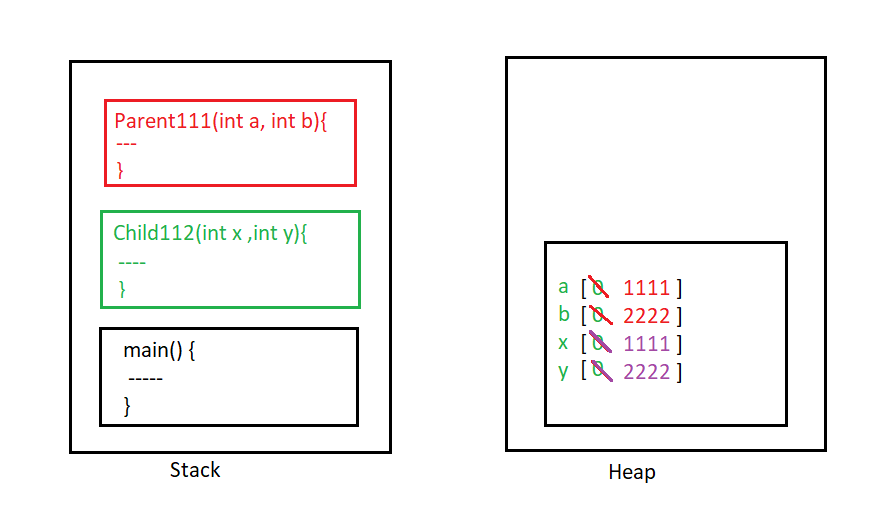
// super() is called default by jvm.



Eg : Constructor\_Working\_During\_Inheritance\_Eg4



// Here super(x,y) is written manually



// try examples with this().