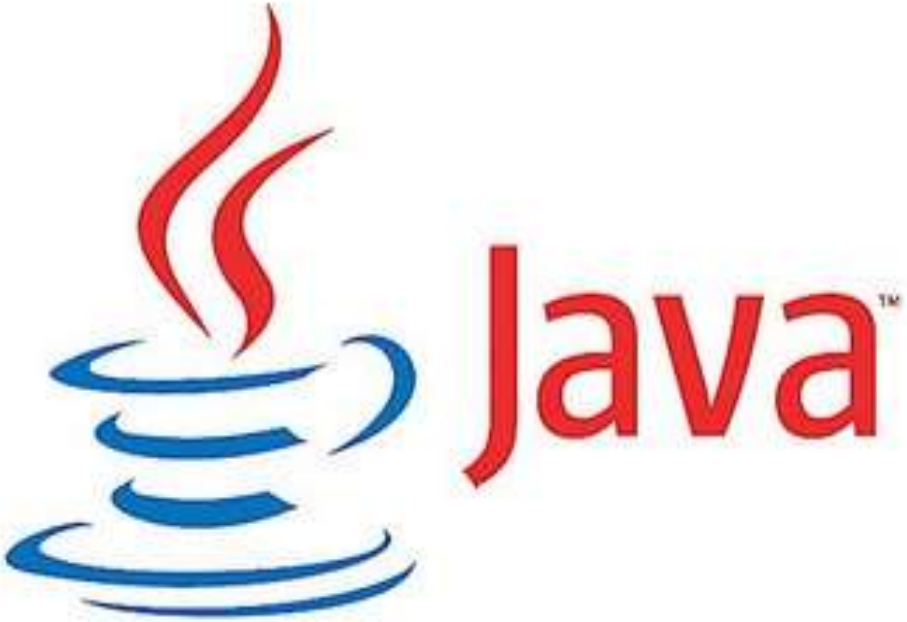
Bharathidasan University

Centre for Differently Abled Persons
Tiruchirappalli - 620024.

- Programme Name : Bachelor of Computer Applications
- Course Code : 23UCACC04
- Course Title : Programming in Java
- Semester : IV
- Unit : Unit III
- Compiled by : Dr. M. Prabavathy
  Associate Professor

  Ms. M. Hemalatha
  Guest Faculty


Java™

ABSTRACT CLASS

Abstract Class

- Abstraction is a process of **hiding the implementation** details and **showing only functionality** to the user.

- A class which is declared with the abstract keyword is known as an abstract class

It can have abstract and non-abstract methods (method with the body).

It can have constructors and static methods.

Syntax:

```
abstract className
{
        code;
}
```

Abstract Method

A method which is declared as abstract and does not have implementation is known as an abstract method.

Syntax:

```
abstract void MethodName();
```

Example:

```
abstract class Bike
{
    abstract void run();
}
```

```
class Honda extends Bike
{
   void run()
   {
         System.out.println("running safely");
   }

public static void main(String args[])
{
   Honda obj = new Honda();
   obj.run();
}
}
```

FINAL CLASS

Final Class

- The final keyword in java is used to restrict the user.
- The java final keyword can be used in many context.

1. Java Final Variable

If variable is declared as final, variable value cannot be changed

The values remain constant.

Example:

```
class Bike
{
        final int speedlimit=90;
        void run()
        {
                speedlimit=400;  //Compile time error
        }
public static void main(String args[])
{
        Bike obj1 =new  Bike();
        obj1.run();
   }
}
```

2. Java Final Method

If method declared as final, cannot override that method.

Example

```
class Bike
{
        final void run()
        {
                System.out.println("running");}
        }
}
```

```
class Honda extends Bike
{
   void run()
   {
         System.out.println("running safely with 100kmph");
   }
   public static void main(String args[])
   {
         Honda honda= new Honda();
         honda.run();
   }
}
```

The program throws Compile time error, because final method cannot be overridden.

3. Java Final Class

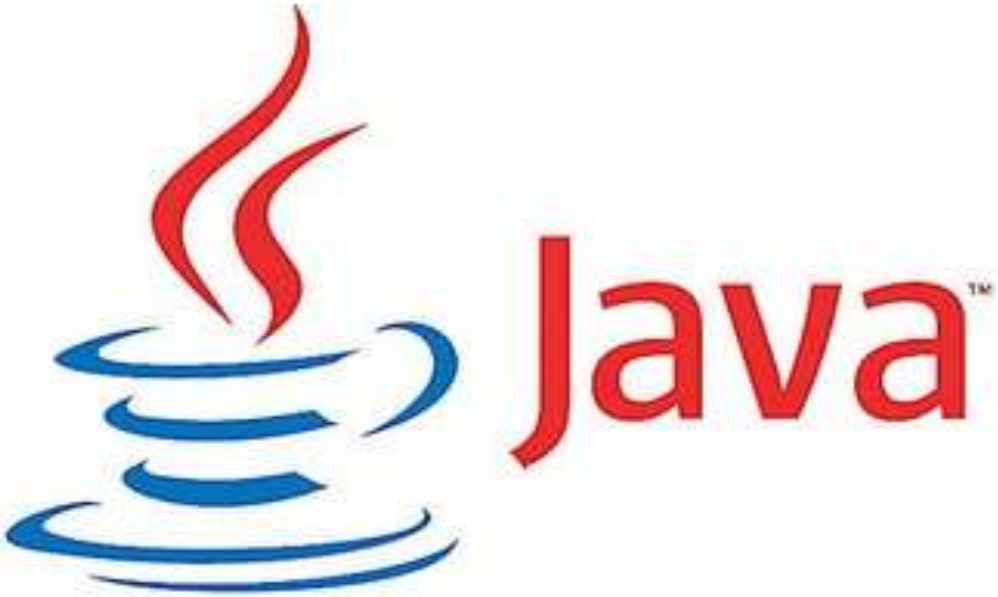
If any class declared as final, it cannot be extended.

Example

```
final class Bike
{
}
class Honda extends Bike
{
        void run()
        {
                System.out.println("running safely with 100kmph");
        }
```

```
    public static void main(String args[])
    {
        Honda obj= new Honda();
        obj.run();
    }
}
```

It throws compile time error as final class cannot be extended.


Java™

INHERITANCE

Inheritance

- Inheritance can be defined as the process where **one class accesses the properties (methods and fields) of another class.**

Super Class / Parent Class:

- a subclass inherits the properties from super class
- It is also called a base class or a parent class.

Sub Class / Child Class:

- inherits the properties from other class.
- called a derived class, extended class, or child class.

Syntax:

```
class Subclass-name extends Superclass-name
{
    //methods and fields
}
```

The **'extends'** keyword indicates deriving new class from an existing class.

Example:

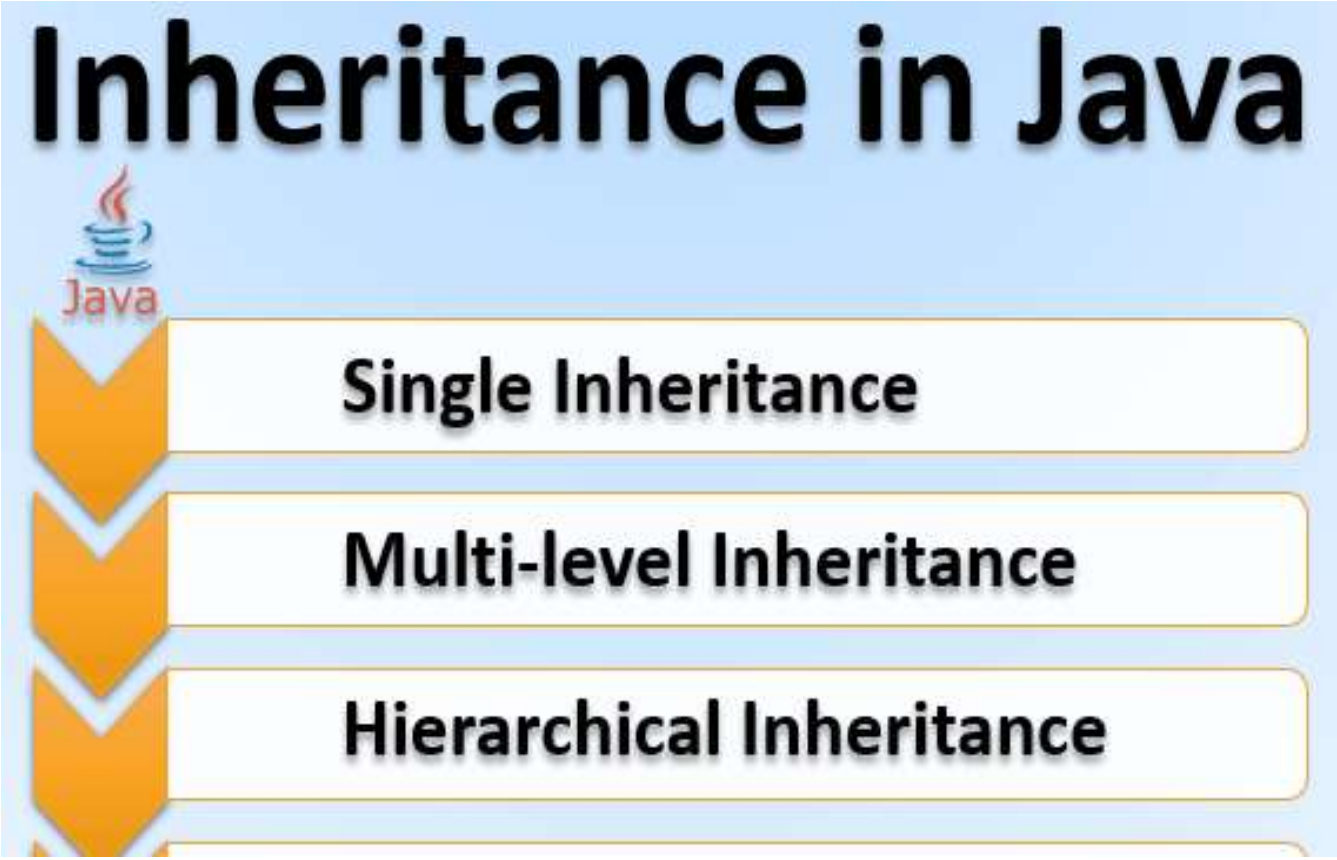
```
class A
{
        int a,b;
public void get( )
{
        a=12;
        b=5;
}
}
```

```
class B extends A
{
    int c;
    public void mul( )
    {
        c = a*b;
        System.out.println("Product = " + c);
    }

public static void main(String args[ ])
{
    B obj = new B( );
    obj.get( );
    obj.mul( );
}
}
```

Output:

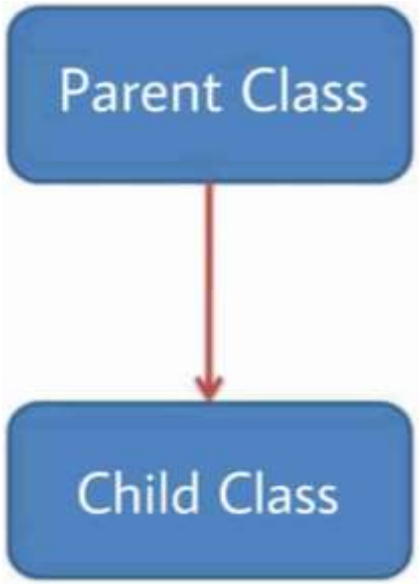
```
        Product  = 60
```

Types of Inheritance



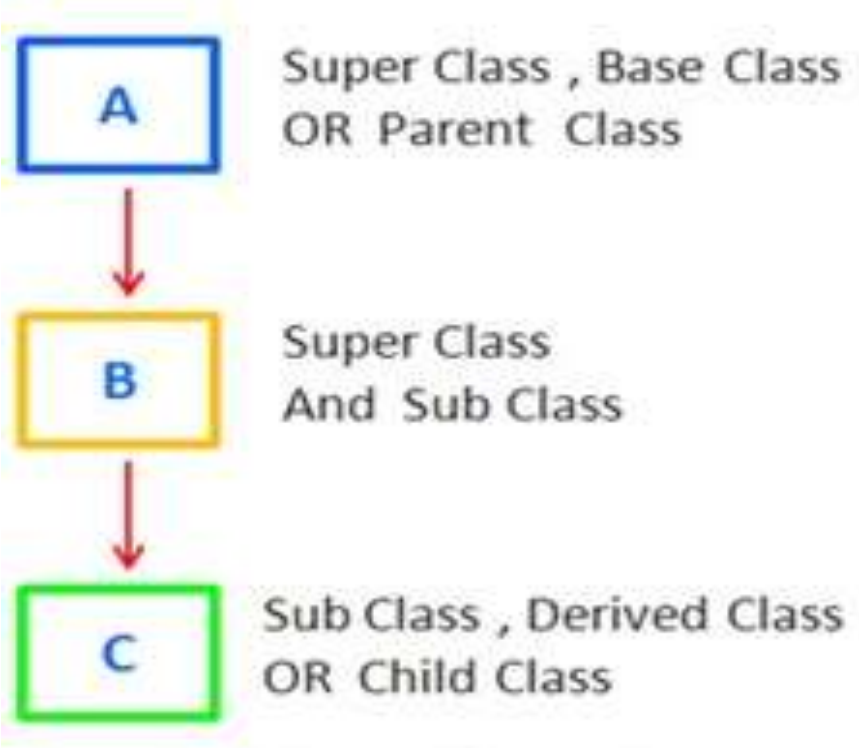
1. Single Inheritance

- When **an one class inherits from another class**, it is known as a single inheritance.



2. Multilevel Inheritance

- A class inherits **properties from a class** which **again has inherits properties.**



EXAMPLE:

```
class A
{
        int a;
        public void assign( )
        {
                a=12;
        }
}
class B extends A
{
        int b;
        public void get( )
        {
                b=5;
        }
}
```

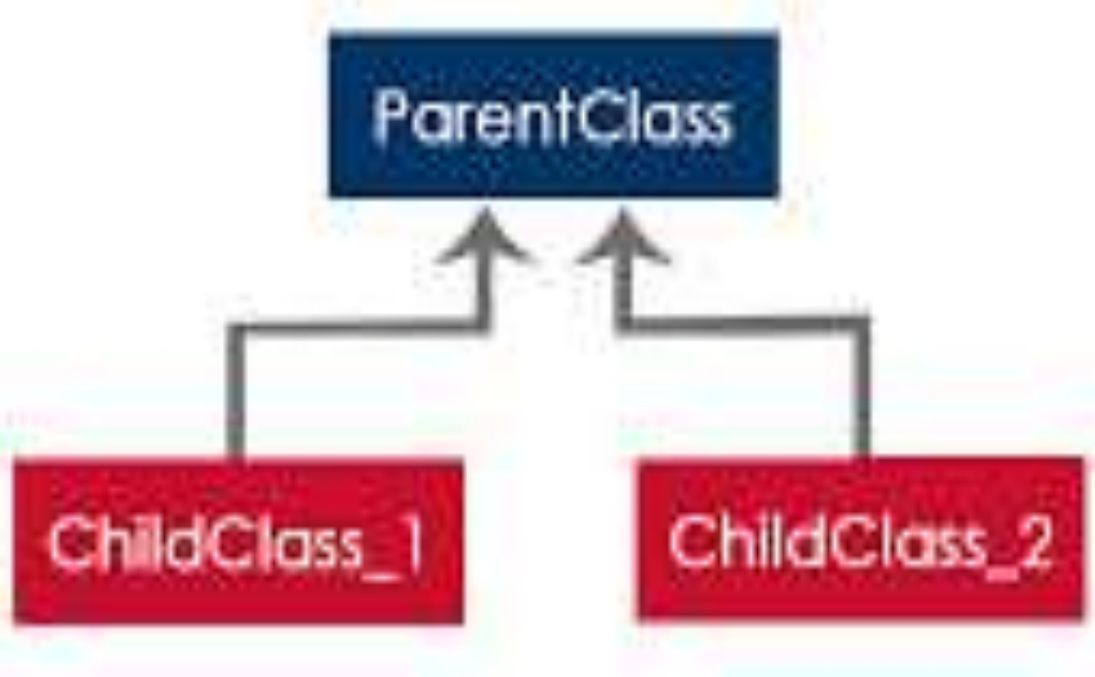
```
class C extends B
{
    int c;
    public void add( )
    {
            c = a+b;
            System.out.println(“ Sum = “ + c);
    }

public static void main(String args[ ])
{
    C obj = new C( );
    obj.assign( );
    obj.get( );
    obj.add( );
}
}
```

Output:

```
            Sum = 17
```

3. Hierarchical Inheritance

- When **two or more classes inherits a single class**, it is known as hierarchical inheritance.



Example:

```
class A
{
        int a,b;
        public void get( )
        {
                a=12;
                b=5;
        }
}
class B extends A
{
        public void add( )
        {
                System.out.println(“ Sum = “ + (a+b));
        }
}
```

```
class C extends A
{
    int d;
    public void mul( )
    {
            d = a*b;
            System.out.println("Product = " + d);
    }
public static void main(String args[ ])
{
    C obj = new C( );
    obj.get( );
    obj.mul( );
}
}
```

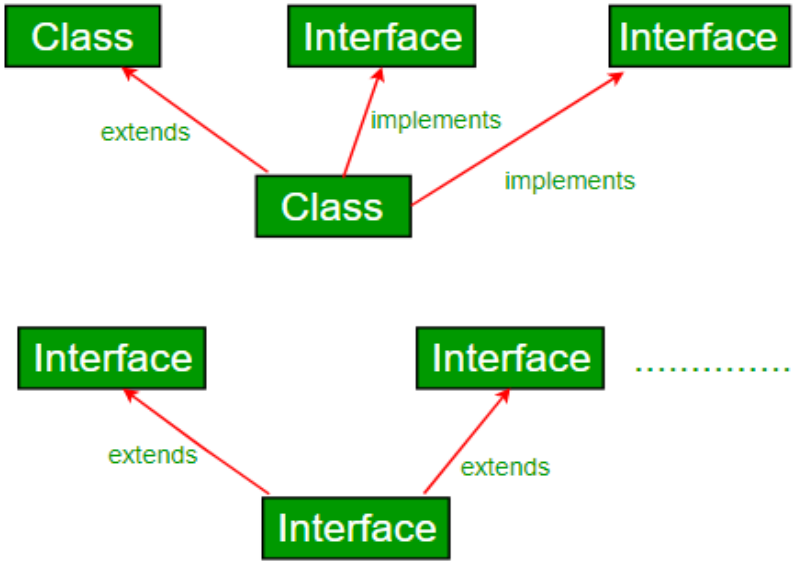
Output:

```
            Product  = 60
```

INTERFACES AND INHERITANCE

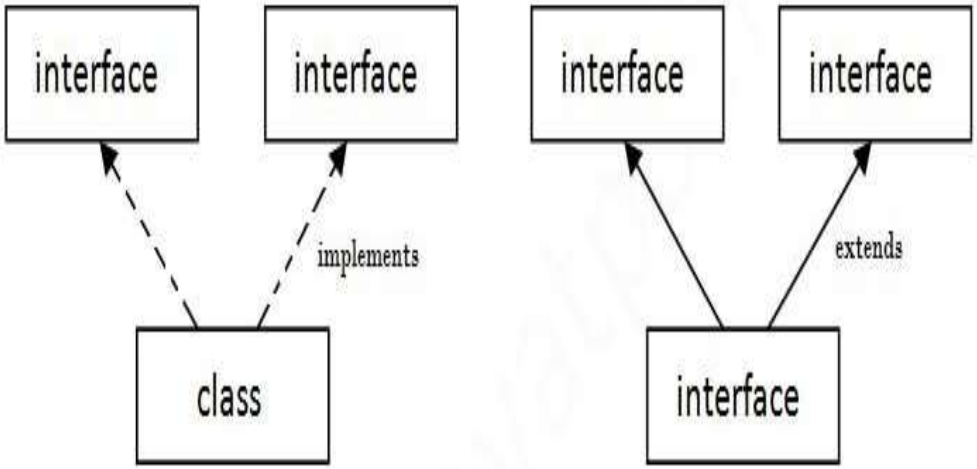
Interface and Inheritance

A class can extends another class and/ can implement one and more than one interface.



Multiple Inheritance

If a class implements multiple interfaces, or an interface extends multiple interfaces, it is known as multiple inheritance.



Example

```
interface Printable
{
   void print();
}
interface Showable
{
   void show();
}
```

```
class A7 implements Printable, Showable
{
   public void print()
   {
         System.out.println("Hello");
   }
   public void show()
   {
         System.out.println("Welcome");
   }
```

```
public static void main(String args[])
   {
        A7 obj = new A7();
        obj.print();
        obj.show();
   }
}
```

OUTPUT:

```
Hello
Welcome
```

POLYMORPHISM

Polymorphism

The word "poly" means many and "morphs“ means forms.

So **polymorphism** means **many forms**.

There are two types of polymorphism in Java

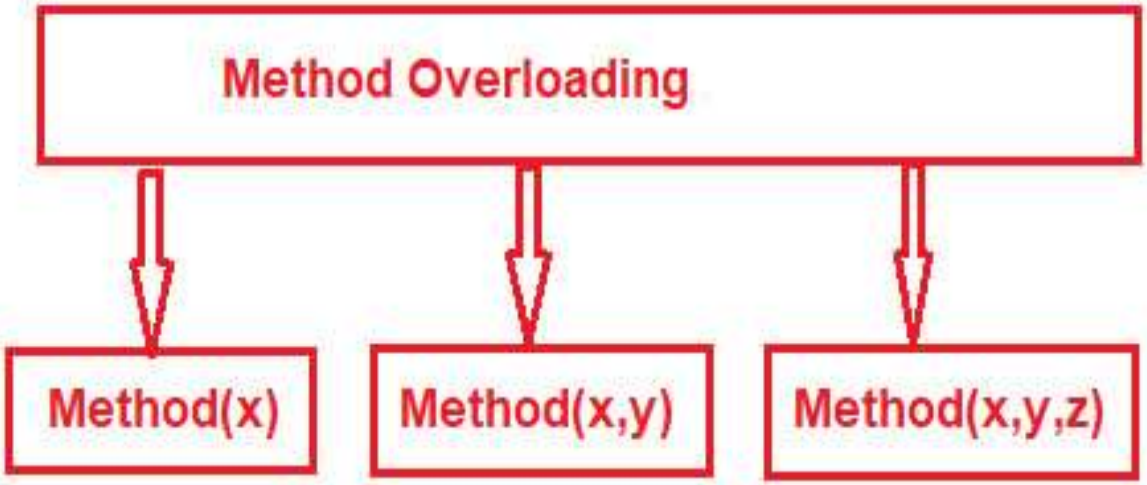
- Compile-time polymorphism
- Runtime polymorphism

Method Overloading

If a class has multiple methods having same name but different in parameters, it is known as Method Overloading.

There are two ways to overload the method in java

- By changing number of arguments
- By changing the data type



1. Method Name is going to be the same
2. Number, Type and order of Parameters is different

Example:

Method Overloading: changing no. of arguments

```
class Adder
{
    static int add(int a,int b)
    {
        return a+b;
    }

    static int add(int a,int b,int c)
    {
        return a+b+c;
    }
}
```

```
class TestOverloading1
{
public static void main(String[] args)
{
        System.out.println(Adder.add(11,11));
        System.out.println(Adder.add(11,11,11));
}
}
```

OUTPUT:

```
22
33
```

Example:

Method Overloading: changing data type of arguments

```
class Adder
{
   static int add(int a, int b)
   {
         return a+b;
   }
   static double add(double a, double b)
   {
         return a+b;
   }
}
```

```
class TestOverloading2
{
public static void main(String[] args)
{
        System.out.println(Adder.add(11,11));
        System.out.println(Adder.add(12.3,12.6));
}
}
```

OUTPUT:

```
22
24.9
```

Method Overriding

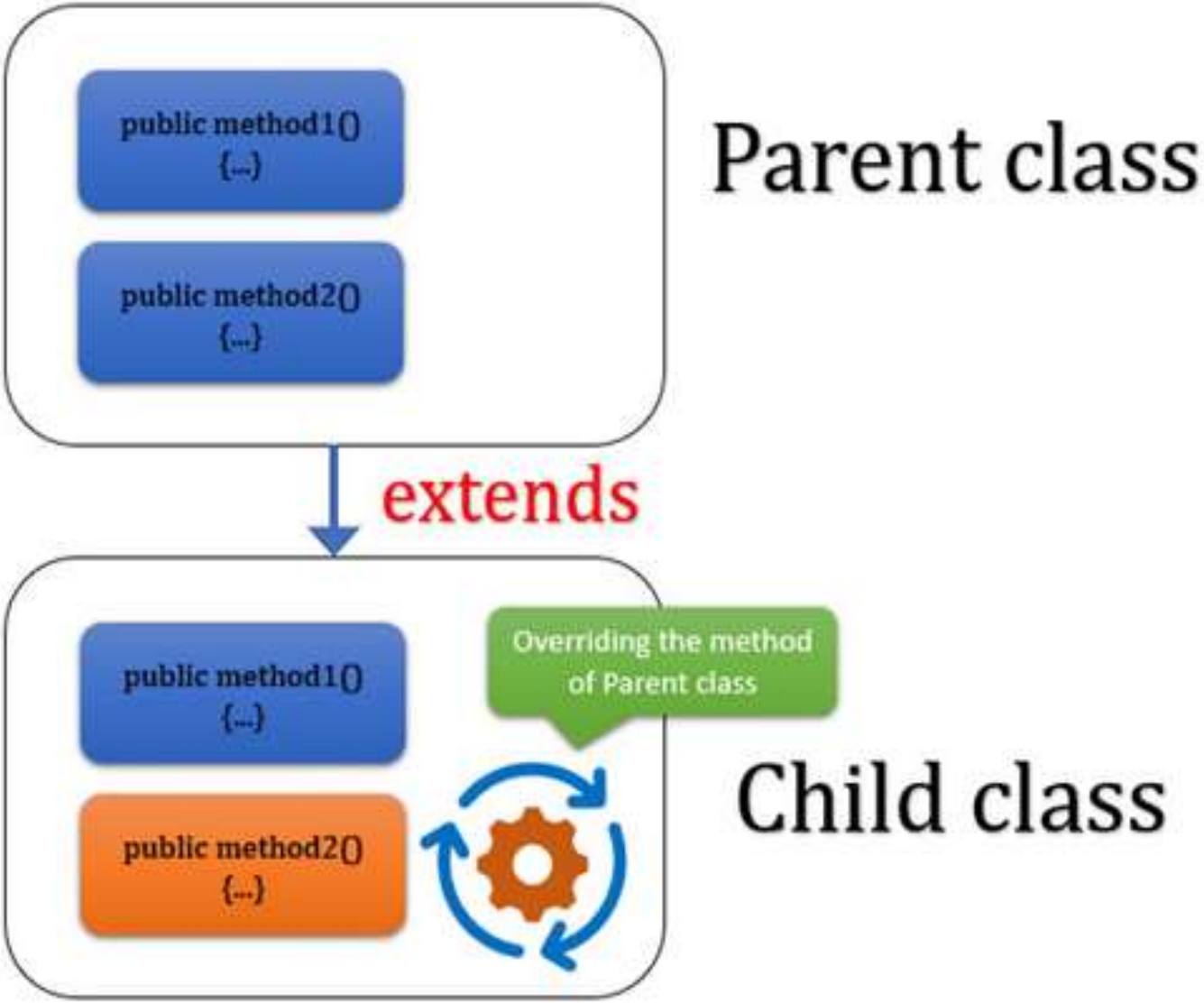
If subclass (child class) has the same method as declared in the parent class, it is known as method overriding.

It is used for runtime polymorphism

Rules for Java Method Overriding

The method must have the same name as in the parent class

The method must have the same parameter as in the parent class


public method1()
{...}
public method2()
{...}
Parent class
extends
public method1()
{...}
Overriding the method
of Parent class
public method2()
{...}
Child class

Example: Method Overriding

```
import java.io.*;
class A
{
        public void show()
        {
                System.out.print("Welcome");
        }
}
class B extends A
{
        public void show()
        {
                super.show();
                System.out.print(" to CDAP");
        }
}
```

```
class C
{
   public static void main(String args[])
   {
                B obj = new B();
                obj.show();
   }
}
```

Output:

Welcome to CDAP