

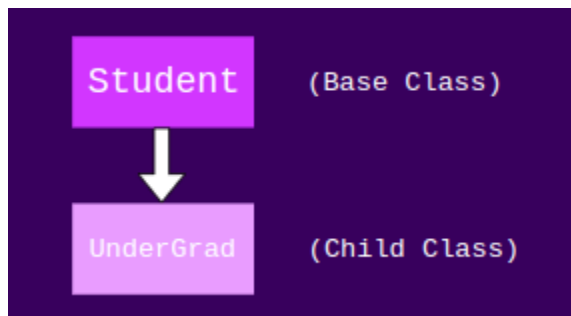
Inheritance

Inheritance provides a way to create a new class from an existing class. The new class is a *specialized* version of the existing class such that it inherits elements of the existing class.

Terms Used:

- **Superclass(Parent Class):** This class allows to re-use of its members in another class.
- **Derived Class(Child Class or Subclass):** This class is the one that inherits from the superclass.

Here, the classes can be built by using previously created classes. The derived class is declared but is followed by the word "extends" and then the name of the parent class



Note:

- ***Inheritance*** establishes an "is an" relationship between ***classes***.
- ***An object of the derived class "is an" object of the base class or the parent class. For example:***
 - **An UnderGrad is a Student**

An *object* of child class has:

- ***All members*** are defined in the child class.

- All *members* declared in the parent class.

An object of a child class can use:

- The members that are defined in the child class.
- The members that are defined in the parent class.

Creation of subclasses:

```
class Student{
    public String name;
    public int age;
}

class Undergrad extends Student{
    String sub;
    public Undergrad(){
        this.sub = "Java"
        this.name="Java";
        this.age=25;
    }
}
```

Types of Inheritance:

1. Single Inheritance:

Here, a class inherits the properties from a single class i.e. a single class extends another single class. In figure, class A is parent class and class B is child class.

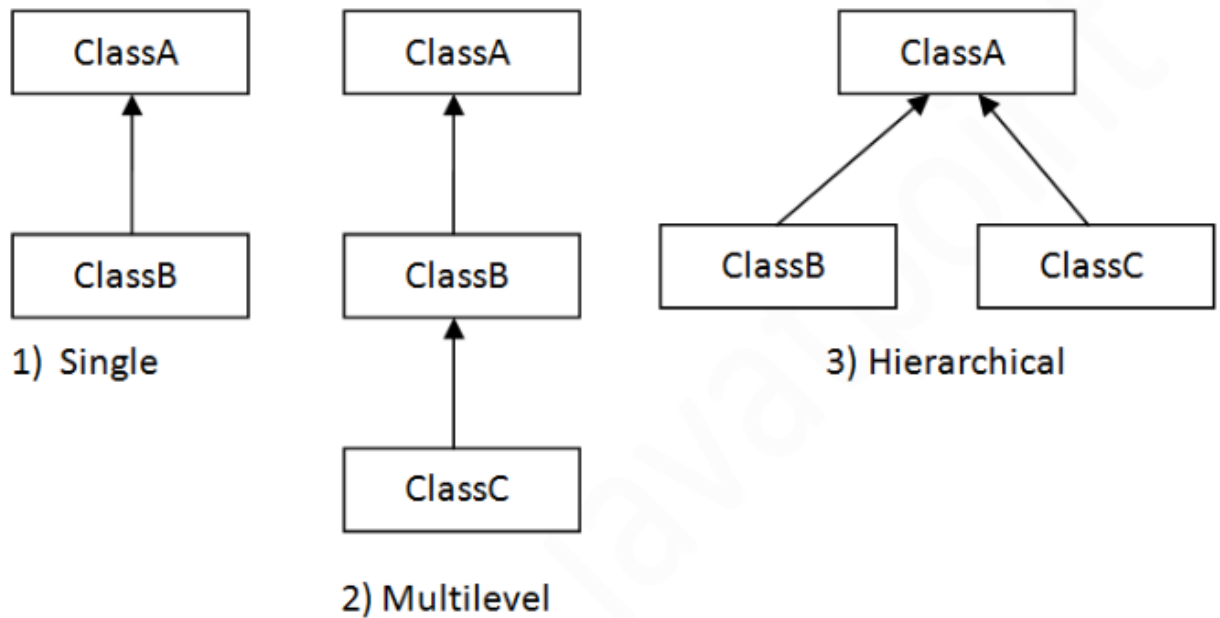
2. Multilevel inheritance :

Here, a derived class inherits a base class. Also, derived class act as a base class of other lower level class.

In figure, class A is the base class for class B. Also, class B serves as base class for derived class C.

3. Hierarchical inheritance

Here, two or more classes inherit a single class. From figure of hierarchical inheritance, class B and class C are derived class of class A.



Example 1a: Inheritance example without using constructor (Single inheritance)

```

class Vehicle {
    String carName="Bmw";
    void display() {
        System.out.println("I am a vehicle.");
    }
}

class Car extends Vehicle {
    void display() {
        System.out.println(carName + " is a car.");
    }
}

public class MainInheritance1 {
    public static void main(String[] args) {
        Vehicle vehicle = new Vehicle();
        vehicle.display();

        Car car = new Car();
        car.display();
    }
}

```

Output:

```

I am a vehicle.
Bmw is a car.

```

Example 1 b. Multilevel inheritance Example

```

class Batch1 {

    void display1()
    {
        System.out.println("Batch 1 is displayed");
    }
}

class Batch2 extends Batch1 {
    void display2()
    {
        System.out.println("Batch 2 is displayed");
    }
}
class Batch3 extends Batch2 {
    void display3() {
        System.out.println("Batch 3 is displayed");
    }
}
public class MainTest1b{
    public static void main(String[] args){
        Batch3 b=new Batch3();
        b.display1();
        b.display2();
        b.display3();
    }
}

```

Output of Program 1b.:

```

Batch 1 is displayed
Batch 2 is displayed
Batch 3 is displayed

```

Example 1 c. Hierarchical Inheritance Program Example

```
class Batch1 {  
  
    void display1()  
    {  
        System.out.println("Batch 1 is displayed");  
    }  
}  
  
class Batch2 extends Batch1 {  
    void display2()  
    {  
        System.out.println("Batch 2 is displayed");  
    }  
}  
class Batch3 extends Batch1 {  
    void display3() {  
        System.out.println("Batch 3 is displayed");  
    }  
}  
public class MainTest1c{  
    public static void main(String[] args){  
        Batch2 b1=new Batch2();  
        Batch3 b=new Batch3();  
        b1.display1();  
        b1.display2();  
        b.display1();  
        b.display3();//here b.display2 is not accepted  
    }  
}
```

Output:

```
Batch 1 is displayed  
Batch 2 is displayed  
Batch 1 is displayed  
Batch 3 is displayed
```

Example 1d. Program to display salary and bonus using inheritance taking Employee as base class and Programmer as derived class.

```
class Employee {  
    float salary=40000.0f;  
}  
class Programmer extends Employee  
{  
    int bonus=10000;  
    public static void main(String[] args)  
    {  
        Programmer p=new Programmer();  
        System.out.println("Salary is:"+p.salary);  
        System.out.println("Bonus is:"+p.bonus);  
    }  
}
```

Example 2: Inheritance with the use of constructor

```
class Animal {
    String name;

    Animal(String name) {
        this.name = name;
    }

    void eat() {
        System.out.println(name + " is eating.");
    }
}

class Dog extends Animal {
    Dog(String name) {
        super(name);
    }

    void bark() {
        System.out.println(name + " is barking.");
    }
}

class MainInheritance2 {
    public static void main(String[] args) {
        Dog dog = new Dog("Rocky");
        dog.eat();
        dog.bark();
    }
}
```

Output:

```
Rocky is eating.
Rocky is barking.
```


Example 3: Inheritance with the use of constructor and super keyword

```
class Person {
    String name;

    Person(String name) {
        this.name = name;
    }
}

class Student extends Person {
    int roll;

    Student(String name, int roll) {
        super(name);
        this.roll = roll;
    }

    void display() {
        System.out.println("Name: " + name);
        System.out.println("Roll Number: " + roll);
    }
}

public class MainInheritance3 {
    public static void main(String[] args) {
        Student student = new Student("Ram", 29);
        student.display();
    }
}
```

Output:

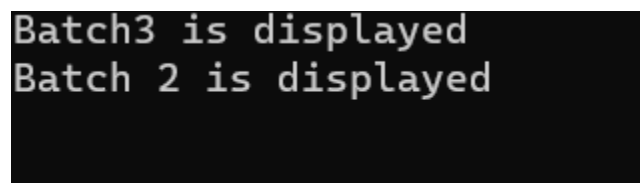
```
Name: Ram
Roll Number: 29
```

Example 4: Program using concept of constructors and inheritance

Example 5:

```
class Batch1 {
    void display1()
    {
        System.out.println("Batch 1 is displayed");
    }
}
class Batch2 extends Batch1 {
    void display1()
    {
        System.out.println("Batch 2 is displayed");
    }
}
class Batch3 extends Batch2 {
    Batch3() {
        System.out.println("Batch3 is displayed");
    }
}
public class MainTest5{
    public static void main(String[] args){
        Batch3 b=new Batch3();
        b.display1();
    }
}
```

Output:



```
Batch3 is displayed
Batch 2 is displayed
```

Question: How to print "Batch1 is displayed"???

Ans: Using constructor in Base class

Example 6: Illustrating inheritance concept with constructor

```
class Batch1 {
    Batch1() {
        System.out.println("Batch 1 is displayed"); }
    void display1()
    {
        System.out.println("Batch 1 is displayed");
    }
}

class Batch2 extends Batch1 {
    void display1()
    {
        System.out.println("Batch 2 is displayed");
    }
}

class Batch3 extends Batch2 {
    Batch3() {
        System.out.println("Batch 3 is displayed");
    }
}

public class MainTest5{
    public static void main(String[] args){
        Batch3 b=new Batch3();
        b.display1();
    }
}
```

Output:

```
Batch 1 is displayed  
Batch3 is displayed  
Batch 2 is displayed
```

Use of “super” in Java

- It is a reference variable used to refer to an immediate parent class object.
- When we create an instance of a subclass, the instance of superclass is created implicitly which is referred to by a super reference variable.

Notes:

- It refers to the immediate parent class instance variable.
- It is used to invoke the immediate parent class method.
- `super()` can be used to invoke immediate parent class constructor.

Example 7:

```

class Employee {
    int id;
    String name;
    Employee(int id, String name){
        this.id=id;
        this.name=name;
    }
}
class Programmer extends Employee {
    float salary;
    Programmer(int id,String name,float salary){
        super(id,name);
        this.salary=salary;
    }

    void display(){
        System.out.println("I am "+name+" and my id is " +id+". Also, I get salary of "+salary);
    }
}
class Person {
    public static void main(String[] args){
        Programmer p=new Programmer(1,"Ram",50000);
        p.display();
    }
}

```

Output:

```
I am Ram and my id is 1. Also, I get salary of 50000.0
```