Idris Ratlamwala
C31 - 2003145

Experiment 12: Programs on Inheritance

Theory :

Inheritance is an important pillar of OOP(Object-Oriented Programming). It is the mechanism in java by which one class is allowed to inherit the features(fields and methods) of another class.

Important terminology:

- **Super Class:** The class whose features are inherited is known as superclass(or a base class or a parent class).
- **Sub Class:** The class that inherits the other class is known as a subclass(or a derived class, extended class, or child class). The subclass can add its own fields and methods in addition to the superclass fields and methods.
- **Reusability:** Inheritance supports the concept of "reusability", i.e. when we want to create a new class and there is already a class that includes some of the code that we want, we can derive our new class from the existing class. By doing this, we are reusing the fields and methods of the existing class.

The **extends** keyword is used for inheritance.

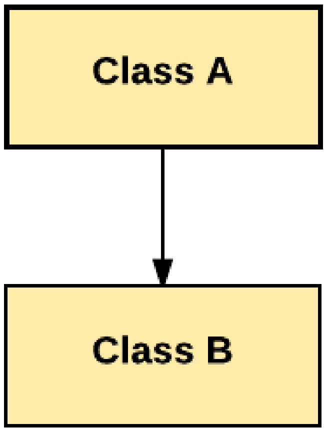
The syntax of Java Inheritance

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

Types of Inheritance :

Single Inheritance:

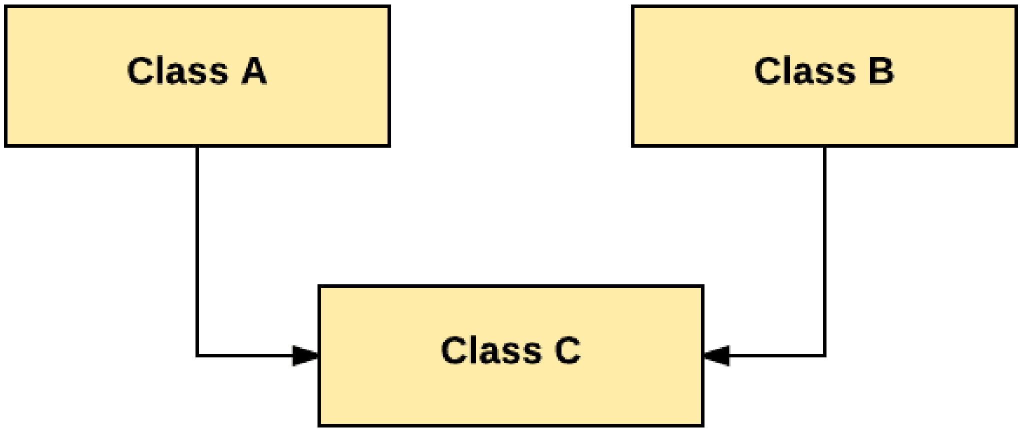
In Single Inheritance one class extends another class (one class only).



In above diagram, Class B extends only Class A. Class A is a super class and Class B is a Sub-class.

Multiple Inheritance:

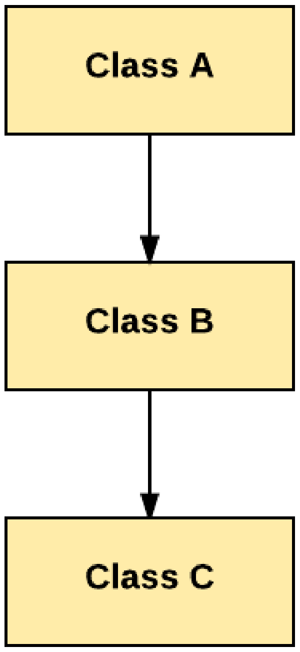
Multiple Inheritance is one of the inheritance in Java types where one class extending more than one class. Java does not support multiple inheritance.



As per above diagram, Class C extends Class A and Class B both.

Multilevel Inheritance:

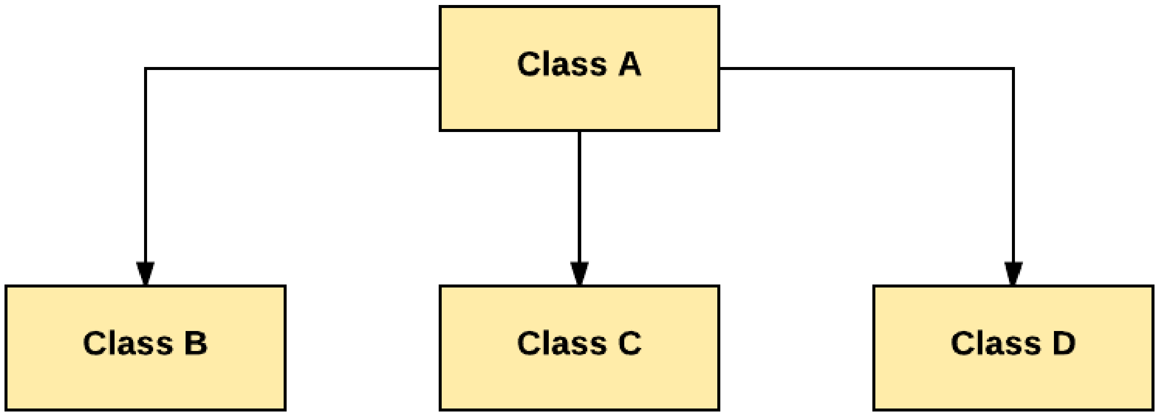
In Multilevel Inheritance, one class can inherit from a derived class. Hence, the derived class becomes the base class for the new class.



As per shown in diagram Class C is subclass of B and B is a of subclass Class A.

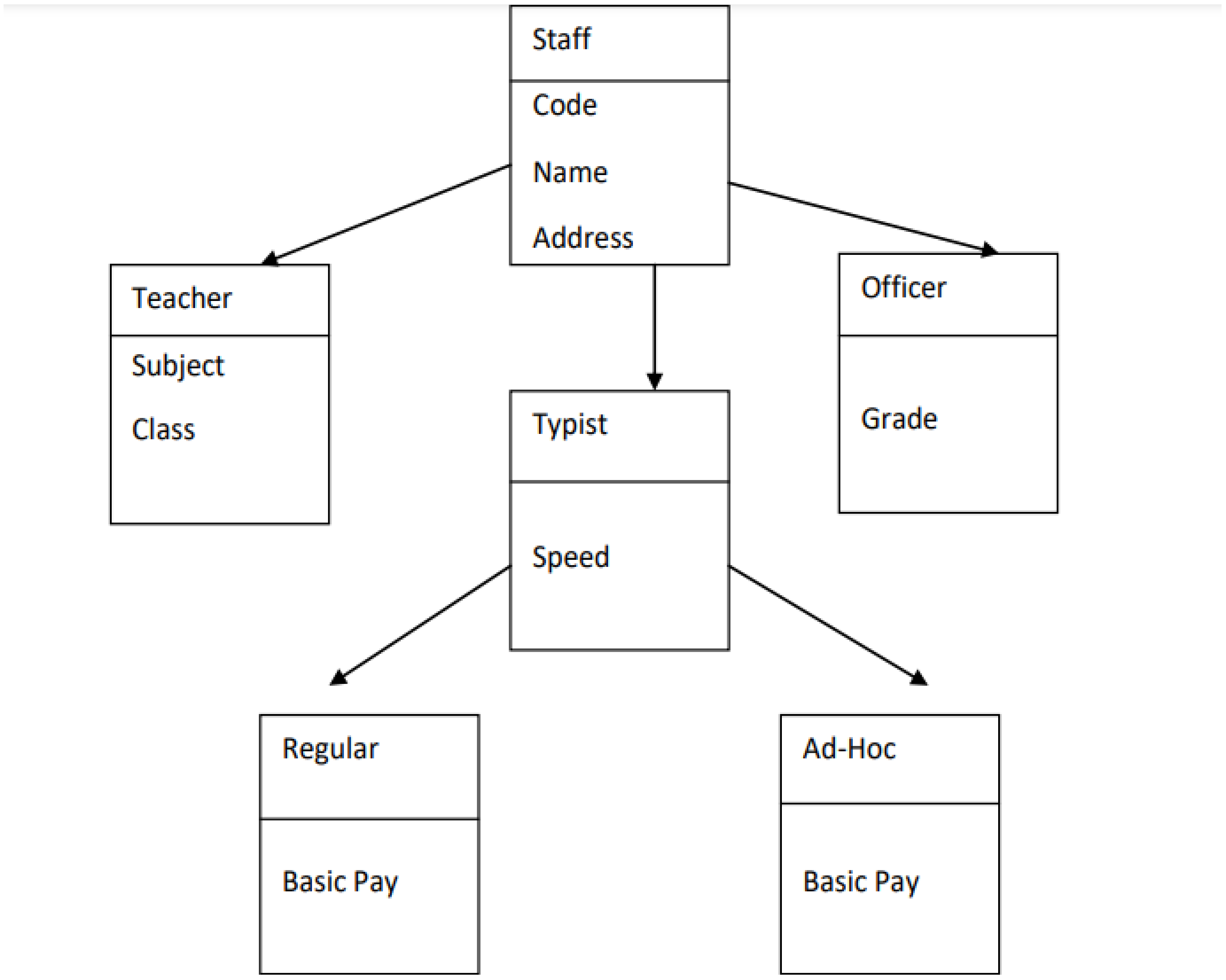
Hierarchical Inheritance:

In Hierarchical Inheritance, one class is inherited by many sub classes.



A.

Aim : Write a Java program to implement following Inheritance,

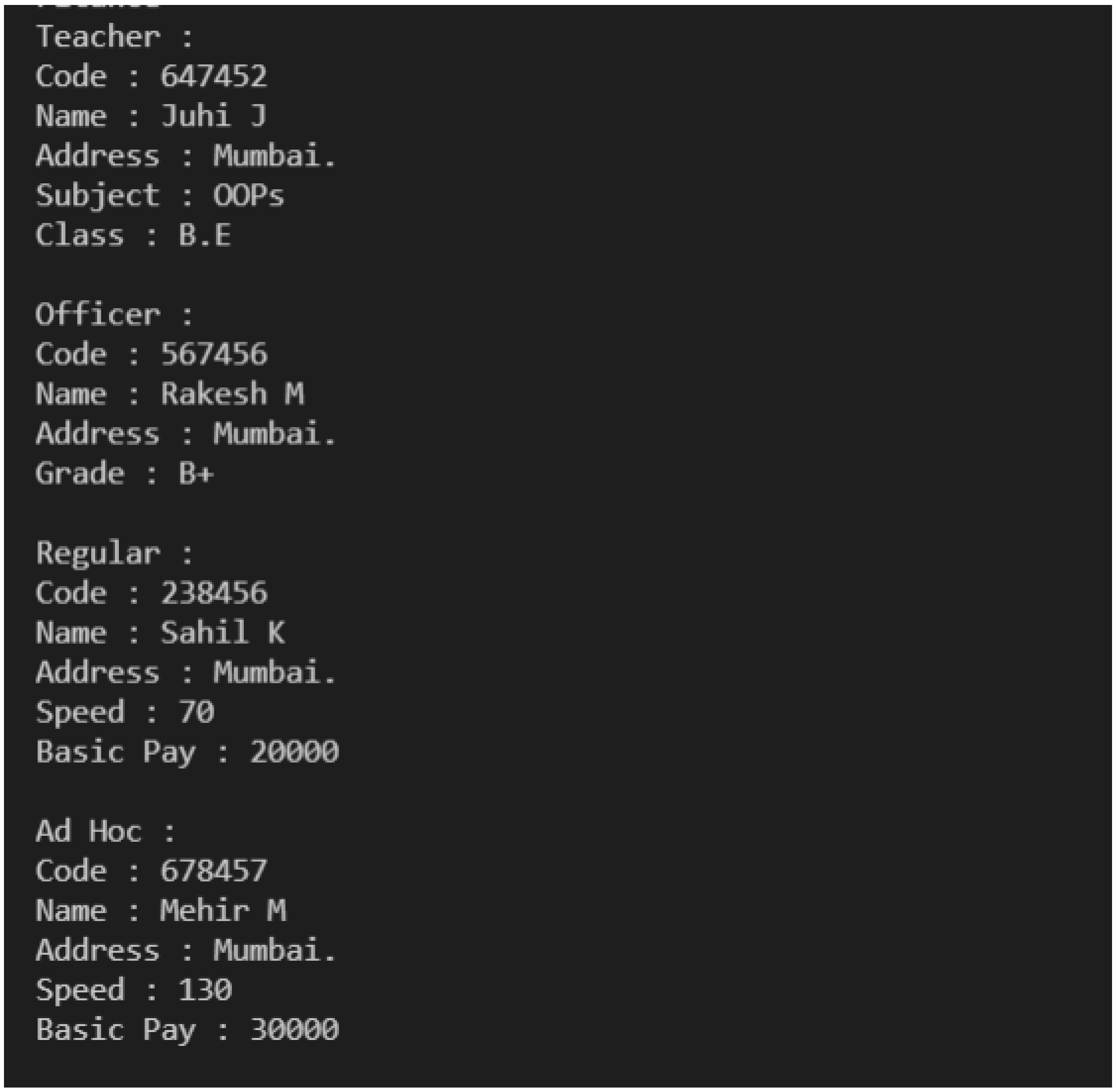


Program :

```
class Staff{
    int code;
    String name,address;

    Staff(int code,String name,String address){
        this.code=code;
        this.name=name;
        this.address=address;
    }
    void staffPrint(){
        System.out.println("Code : "+code+"\nName : "+name+"\nAddress : "+address);
```

```java
        }
    }

    class Teacher extends Staff{
        String subject,Class;

        Teacher(int code,String name,String address,String subject,String Class){
            super(code, name, address);
            this.subject=subject;
            this.Class=Class;
        }
        void print(){
            staffPrint();
            System.out.println("Subject : "+subject+"\nClass : "+Class+"\n");
        }
    }

    class Officer extends Staff{
        String grade;

        Officer(int code,String name,String address,String grade){
            super(code, name, address);
            this.grade=grade;
        }
        void print(){
            staffPrint();
            System.out.println("Grade : "+grade+"\n");
        }
    }

    class Typist extends Staff{
        int speed;
        Typist(int code,String name,String address,int speed){
            super(code, name, address);
            this.speed=speed;
        }
        void typistPrint(){
            staffPrint();
            System.out.println("Speed : "+speed);
        }
    }

    class Regular extends Typist{
        int basic_pay;
        Regular(int code,String name,String address,int speed,int basic_pay){
            super(code, name, address, speed);
            this.basic_pay=basic_pay;
        }
        void print(){
            typistPrint();
        System.out.println("Basic Pay : "+basic_pay+"\n");
    }
}

class Ad_Hoc extends Typist{
    int basic_pay;
    Ad_Hoc(int code,String name,String address,int speed,int basic_pay){
        super(code, name, address, speed);
        this.basic_pay=basic_pay;
    }
    void print(){
        typistPrint();
        System.out.println("Basic Pay : "+basic_pay+"\n");
    }
}


public class CollegeInheritance {
    public static void main(String[] args) {
        int code,speed,basicPay;
        String name,address,subject,Class,grade;

        name="Juhi J";code=647452;address="Mumbai.";subject="OOPs";Class="B.E";
        Teacher t1 = new Teacher(code, name, address, subject, Class);

        name = "Rakesh M"; code = 567456; grade = "B+";
        Officer o1 = new Officer(code, name, address, grade);

        name = "Sahil K"; code = 238456; speed = 70; basicPay = 20000;
        Regular r1 = new Regular(code, name, address, speed, basicPay);

        name = "Mehir M"; code = 678457; speed = 130; basicPay = 30000;
        Ad_Hoc a1 = new Ad_Hoc(code, name, address, speed, basicPay);

        System.out.println("Teacher :");
        t1.print();
        System.out.println("Officer :");
        o1.print();
        System.out.println("Regular :");
        r1.print();
        System.out.println("Ad Hoc :");
        a1.print();
    }
}
```

Output :



```
Teacher :
Code : 647452
Name : Juhi J
Address : Mumbai.
Subject : OOPs
Class : B.E

Officer :
Code : 567456
Name : Rakesh M
Address : Mumbai.
Grade : B+

Regular :
Code : 238456
Name : Sahil K
Address : Mumbai.
Speed : 70
Basic Pay : 20000

Ad Hoc :
Code : 678457
Name : Mehir M
Address : Mumbai.
Speed : 130
Basic Pay : 30000
```