OBJECT ORIENTED PROGRAMMING

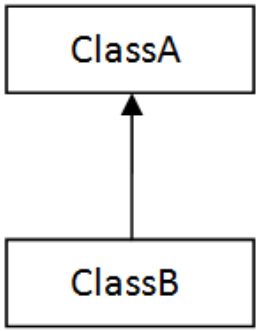
Affefah Qureshi

Department of Computer Science

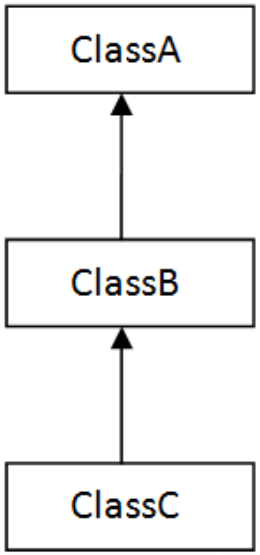
Iqra University, Islamabad Campus.

TYPES OF INHERITANCE IN JAVA

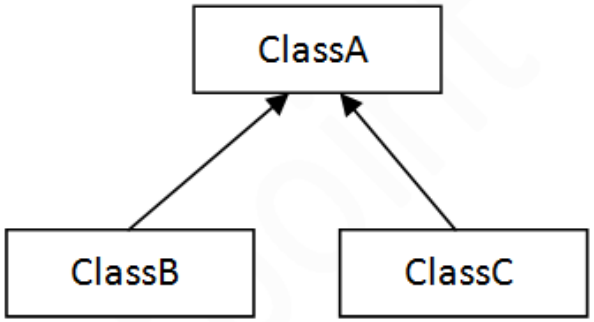
- On the basis of class, there can be three types of inheritance in java: single, multilevel and hierarchical.
- In java programming, multiple and hybrid inheritance is supported through interface only



1) Single

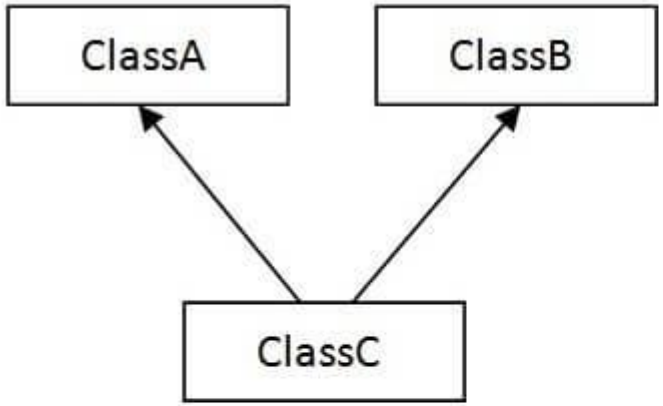


2) Multilevel

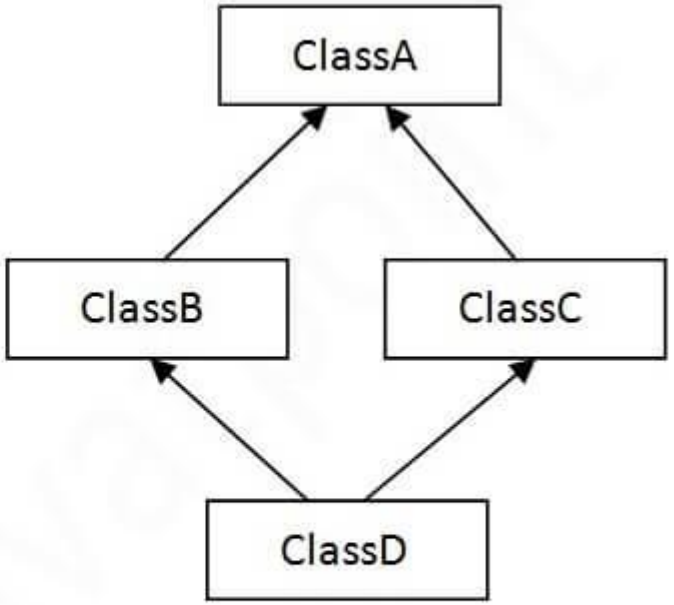


3) Hierarchical

When one class inherits multiple classes, it is known as multiple inheritance. For Example:

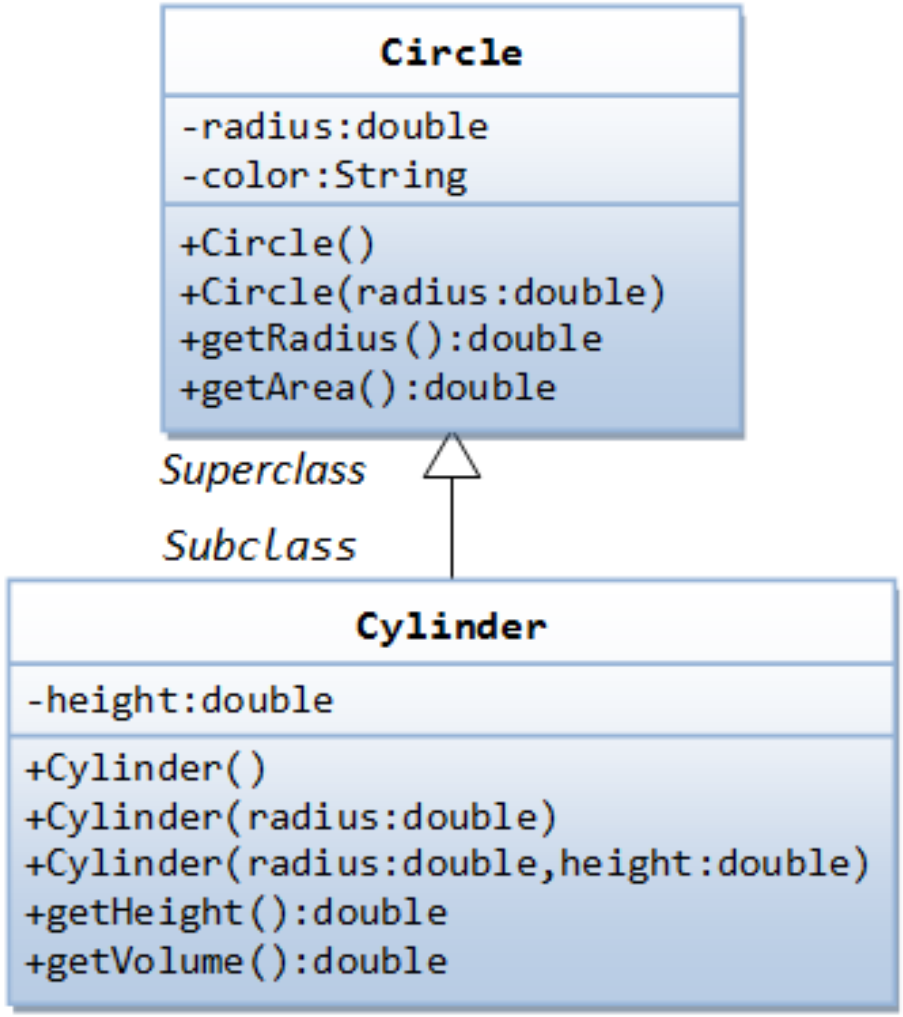


4) Multiple

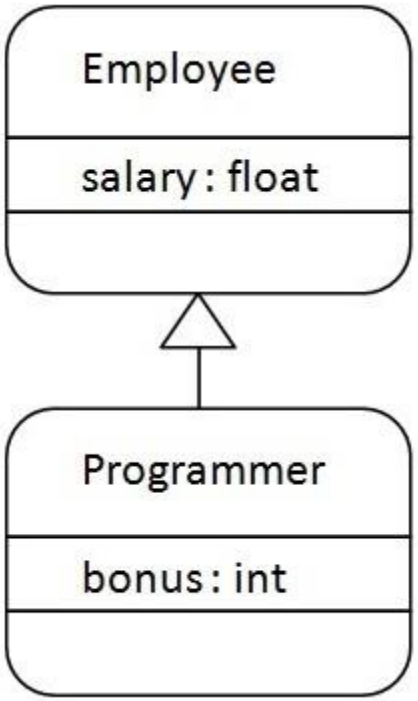


5) Hybrid

EXAMPLES FOR INHERITANCE (UML CLASS DIAGRAM)



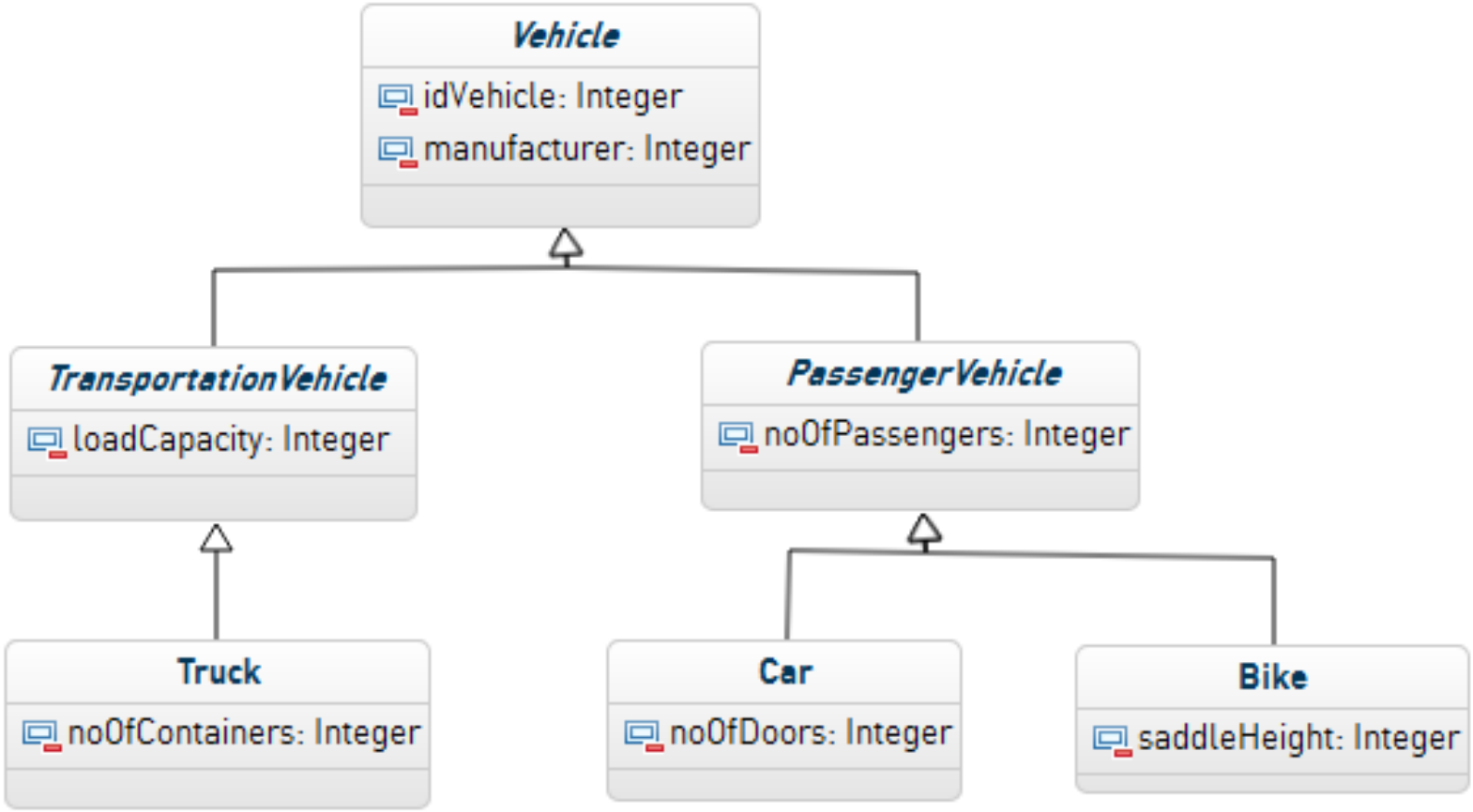
Can we Implement this design???




```
class Employee{
 float salary=40000;
}
class Programmer extends Employee{
 int bonus=10000;
 public static void main(String args[]){
   Programmer p=new Programmer();
   System.out.println("Programmer salary is:"+
p.salary);
   System.out.println("Bonus of Programmer is
:"+p.bonus);
}
}
```

Output:
Programmer salary is:40000.0
Bonus of programmer is:10000

MULTIPLE LEVELS OF INHERITANCE (UML CLASS DIAGRAM)



Can we Implement this design???

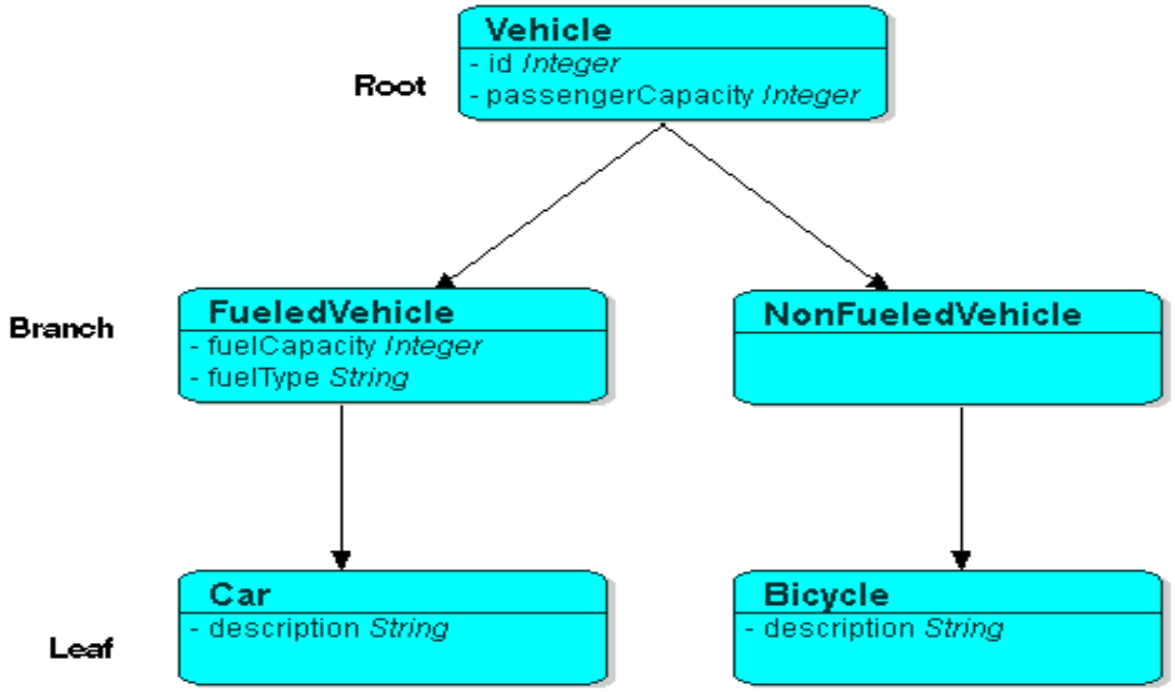
MULTI-LEVEL INHERITANCE

```
class Animal{
        void eat(){System.out.println("eating...");}
              }
class Dog extends Animal{
        void bark(){System.out.println("barking...");}
              }
class BabyDog extends Dog{
        void weep(){System.out.println("weeping...");}
}
class TestInheritance2{
public static void main(String args[]){
BabyDog d=new BabyDog();
d.weep();
d.bark();
d.eat(); }}
```

Output
weeping...
barking...
eating...

MULTI LEVELS OF INHERITANCE (UML CLASS DIAGRAM)

Java Inheritance Hierarchy:



Can we Implement this design???

```
class Animal{
void sound(){
System.out.println(“Make Sound Like...");}
}
class Dog extends Animal{
void sound(){
System.out.println("barking...");}
}
class Cat extends Animal{
void  sound(){
super.sound();
System.out.println(“Meow...");}
}

class TestInheritance{
public static void main(String args[]){
Dog d=new Dog();
Cat c=new Cat();
d.sound();
c.sound();
}}
```

WHY MULTIPLE INHERITANCE IS NOT SUPPORTED IN JAVA?

- To reduce the complexity and simplify the language, multiple inheritance is not supported in java.

A SUPERCLASS VARIABLE CAN REFERENCE A SUBCLASS OBJECT

- A reference variable **of a superclass** can be **assigned** a **reference to any subclass** derived from that superclass.

```
SuperClass referenceVariable=new SubClass();
Or
SubClass subClassReference=new SubClass();
SuperClass referenceVariable=subClassReference;
```

A SUPERCLASS VARIABLE CAN REFERENCE A SUBCLASS OBJECT

```
class Person{
  private String name;
  private int age;

public Person(String name, int age){
    this.name = name;
    this.age = age;
  }
  public void displayPerson() {
    System.out.println("Data of the Person class: ");
    System.out.println("Name: "+this.name);
    System.out.println("Age: "+this.age);
  }
}

public class Student extends Person {

  public String branch;

  public int Student_id;

  public Student(String name, int age, String branch,
   int Student_id){

    super(name, age);

    this.branch = branch;

    this.Student_id = Student_id;

}

public void displayStudent() {

  System.out.println("Data of the Student class: ");

  System.out.println("Name: "+this.name);

  System.out.println("Age: "+this.age);

  System.out.println("Branch: "+this.branch);

  System.out.println("Student ID: "+this.Student_id);

}
```

```
public static void main(String[] args) {
    Person person = new Student("Krishna", 20, "IT", 1256);
    person.displayPerson();
  }
}
```

Output:

Data of the Person class:
Name: Krishna
Age: 20

In this case, if you assign a Student object to reference variable of Person class as:

```
public static void main(String[] args) {
  Person person = new Student("Krishna", 20, "IT", 1256);
  person.displayStudent();
}
```

Compile time error

CAST SUPER CLASS OBJECT TO SUB CLASS

- To access the displayStudent() method, which is specific to the Student class, you would need to either cast the person object to a Student within the method call or declare the reference variable as Student type.

```
public static void main(String[] args) {
    Person person = new Student("Krishna", 20, "IT", 1256);
    ((Student) person).displayStudent(); // Casting person to Student and then calling displayStudent()
}
```

HOW TO CONVERT A SUB CLASS VARIABLE INTO A SUPER CLASS TYPE IN JAVA?

```
public static void main(String[] args) {
    Person person = new Person("Krishna", 20);
    //Converting super class variable to sub class type
 Student student = new Student("Krishna", 20, "IT", 1256);
    person = student;
    person.displayStudent();
  }
```

A SUPERCLASS VARIABLE CAN REFERENCE A SUBCLASS OBJECT

- Using this reference you can access the members of super class only
- If you try to access the sub class members a compile time error will be generated.