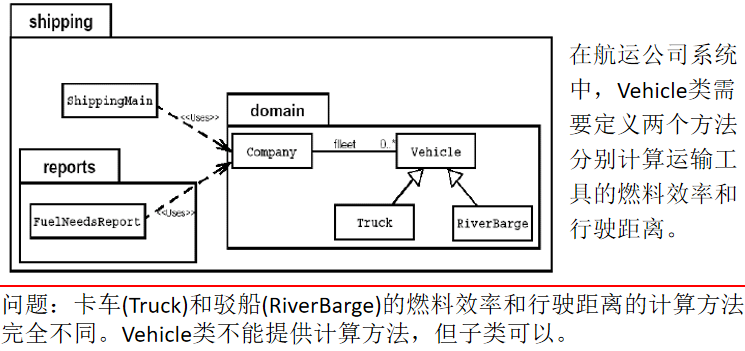
随着继承层次中一个个新子类的定义，类变得越来越具体，而父类则更一般，更通用。类的设计应该保证父类和子类能够共享特征。**有时将一个父类设计得非常抽象，以至于它没有具体的实例，这样的类叫做抽象类**。

* **抽象类**
* 用abstract关键字来修饰一个类时，这个类叫做抽象类；
* 用abstract来修饰一个方法时，该方法叫做抽象方法；
* 抽象方法：只有方法的声明，没有方法的实现。以分号结束。
* abstract int abstractMethod(int a);
* 含有抽象方法的类必须被声明为抽象类；
* 凡是类都有构造器；
* **抽象类不能被实例化**。**抽象类是用来被继承的**，**抽象类的子类必须重写父类的抽象方法，并提供方法体。若没有重写全部的抽象方法，仍为抽象类**。
* 注意：
* 不能用abstract修饰属性：
* 不能用abstract修饰私有方法：**子类不能够覆盖声明为private的方法的。**
* 不能用abstract修饰构造器：**应为构造器不能被重写**
* 不能用abstract修饰静态方法：
* 不能用abstract修饰final方法：
* **抽象类应用**
* 抽象类是用来模型化那些父类无法确定全部实现，而是由其子类提供具体实现的对象的类。



* 解决方案：Java允许类设计者指定：超类声明一个方法但不提供实现，该方法的实现由子类提供。这样的方法称为抽象方法。有一个或更多抽象方法的类称为抽象类。
* Vehicle是一个抽象类，有两个抽象方法。

public **abstract** class Vehicle{

//计算燃料效率的抽象方法

public **abstract** double calcFuelEfficiency();

//计算行驶距离的抽象方法

public **abstract** double calcTripDistance();

}

public class Truck **extends Vehicle**{

public double calcFuelEfficiency( ) {

//写出计算卡车的燃料效率的具体方法

}

public double calcTripDistance( ) {

//写出计算卡车行驶距离的具体方法

}

}

public class RiverBarge **extends Vehicle**{

public double calcFuelEfficiency( ) {

//写出计算驳船的燃料效率的具体方法

}

public double calcTripDistance( ) {

//写出计算驳船行驶距离的具体方法

}

}

* 注：**抽象类不能实例化** new Vehicle()是非法的。
* **思考**
* 问题1：为什么抽象类不可以使用final关键字声明？
* 问题2：一个抽象类中可以定义构造器吗？