抽象类

**为什么引入抽象类?**

抽象类 : 几何图形class Figure

子类: 矩形class Rectangle 圆class Circle三角形class Triangle 等...   属性: 曲边 ,直边, 几条边, 求面积, 求周长...

如果没有抽象类,父类提取出来的共性必须是唯一的,实现的,稳定的,那么求面积,求周长,都不能提取,当一个方法的功能是处理几何图形,参数是Figure f .当接收一个图形时,求其周长还要事先把 f 向下转型,很是麻烦.但是如果引入抽象类,可以把求周长的公式提取出来,但是不必实现,当然也无法实现.这样就不用向下转型了.增加了灵活性,而且子类继承了抽象类,必须把抽象类的所有抽象方法必须全实现,才能new实例,这样抽象类带来了另外一个好处:模板,规定子类必须要干什么!

**抽象类原则**

(1)抽象类用abstract修饰,抽象类不能new出实体

(2)抽象类提取出的共性,能实现的直接实现,不能实现的用abstract修饰,不必实现,由子类继承重写

(3)抽象类应该尽可能提取最多的共性,不用在乎能不能实现.

(4)抽象类不能new出实体,就是用来且必须被继承的,继承者必须重写其全部的抽象方法,才能new出实体,否则,继承者也是个抽象类.

(5)abstract修饰的类被继承才有意义,abstract修饰的方法被重写才能new出实体,而final修饰的类不能被继承,方法不能被重写,所以abstract和final不能同时使用.

(6)private修饰的方法子类无法触及,所以private和abstract也不通用.

(7)抽象类也有构造器,但是这个构造器并不是为了new出对象,而是给继承者的构造器调用的.

抽象方法和普通方法 形式上的区别:

public abstruct void abstructFun();

public void fun() {}

接口

/\*接口的权限是public或friendly,能继承一个或多个接口,不能继承类,不含构造器,不含初始化块\*/

interface JieKou extends 接口1,接口2... {

/\* 接口的所有成员都是public ,就像C语言的const修饰的变量,不再改变,就是"模板" \*/

public static final int VAR = 20;

/\*static 省空间,final 是规范就不要再修改,必须定义时直接初始化\*/

//int VAR = 20;

/\*普通方法必须是抽象的 public abstruct,不用实现 \*/

void f();

/\*类方法 public static ,必须实现 \*/

static void g() {

System.out.println("我是类方法");

}

/\*默认方法也必须实现 public \*/

default void print(String... msgs) {

for(String msg : msgs) {

System,out.println(msg);

}

}

/\*其他成员:内部类,内部接口,枚举类\*/

}

接口不能new出实体,接口可以作为引用变量,指向类new出的实体.

<1>定义引用变量,也可以进行强制类型转换

<2>调用接口的常量

<3>被其他类实现

一个类可以多继承很多接口,但是只能继承一个类.

class extends 类 implements 接口1,接口2... {

}

一个类继承了接口,必须实现接口中所有的抽象方法,不然仍旧只能当抽象类用.

继承接口的类重写接口的抽象方法时,权限只能是public,因为子类重写父类时,只能用相同或者更大的权限.

抽象类和接口的区别

