面向对象的特征：

1. 封装
2. 继承
3. 多态
4. 抽象

一. 封装：把类实现部分隐藏起来，选择性的暴露方法出来

封装的标准：高内聚，低耦合。

内聚：模块内部成员的关联程度

耦合：模块间的关联程度

访问修饰符控制作用范围：

1. public:公共的,公开的在任何地方都可以访问

```
Girl girl = new Girl();
//在类的外部:对象.属性
girl.name = "刀妹";
System.out.println(girl.name);
```

```
girl.sayHi();
```

2. protected:受保护的,在本类,在同包或子类中访问

```
girl.gender = "女";
System.out.println(girl.gender);
```

3. 省缺:在本类或同包中,可以访问

```
girl.age = 21;
System.out.println(girl.age);
```

4. private:只能在本类中访问

```
girl.setHeight(180);
```

```
System.out.println(girl.getHeight());
```

二. 包:package, 用来管理源代码

包以公司域名反写命名　eg:com.baidu

包的实质是一个有层次的文件夹, 点代表下一级

在包中的类, 第一行要加上包的路径

```
//eg:package com.lanou.a
```

在使用其他包 的类, 需要导包,

```
//eg; com.lanou.a.Dog
```

三. 继承:子类可以继承父类的属性和方法

Java中 的继承 的特点:

1. 单根继承:全部都是object的子类
2. 单向继承:一个类只能有一个父类, 一个类可以有多个子类
3. 构造方法不能被继承
4. 子类的构造方法内需要调用父类的构造方法
5. 子类的构造方法内没有调用父类的构造方法, 系统会默认调用父类的无参构造方法
6. 调用父类 的构造方法必须放在第一行
7. 当子类提供的方法, 不满足子类的需求时. 子类可以重写父类的方法

```
练习:封装zombie类
```

```
普通僵尸类
特征:总血量,失血量
行为:被打,死亡
Zombie zombie=new Zombie(50,3);
System.out.println("游戏开始");
```

```
while (zombie.getHP()>0) {
Thread.sleep(500);
zombie.hitted();
}
```

```
路障僵尸类
特征:总血量,失血量,装备
行为:被打,死亡,失去装备
```

```
当血量是总血量的一半时,失去装备,失去装备,是血量是以前的两倍
```

四.多态：一个事物的多种形态

重载，重写，覆盖的区别

1.重载：overload在一个类中方法名相同。参数个数和类型不同、

2.重写：override出现在父类和子类之间，当父类提供的方法不满足子类的需求时，可以重写父类的非静态方法

3.覆盖：重写父类的静态方法

编译时多态：方法重载

运营时多态：父类对象可以接受子类对象

```
Bird bird=new Bird();
Chicken chicken=new Chicken();
Duck duck=new Duck();
```

//调用方法时,先从本类中找方法,找不到,再去父类里找,直到找到object类

```
bird.fly();
chicken.fly();
duck.fly();
```

引用类型的数据转化

1. 向上转型:子类转父类,是安全的转化

```
bird=chicken;
```

2. 向下转型:父类转子类,是不安全的转化

```
duck=(Duck)bird;
```

对象instanceof类,//实例,判断对象是否是这个类的对象

```
if (bird instanceof Chicken) {
chicken = (Chicken) bird;
}
```

练习:

//足球运动类,属性:名字,行为:训练

//篮球运动类,属性:名字,行为:训练

```
Sport sport=new Sport("洋洋",19);
System.out.println(sport);
System.out.println(sport.toString())
```

五. 抽象方法:由abstract修饰方法

抽象方法的特征:

1. 抽向方法没有方法体,
2. 抽象方法必须放在抽象类中

抽象类:由abstract修饰的类

抽象类的特点:

1. 抽象程度:抽象类>类>对象
2. 抽象类不能实例化对象
3. 抽象类需要先转换为普通类,(继承抽象类),才能实例化.

```
Footballer footballer=new Footballer();子类化
```

4. 抽象类可以有抽象方法, 也可以有非抽象方法

5. 一个类继承于抽象类, a重写抽象类的抽象方法b. 本身也是抽象类.

//练习:

//创建抽象类(shape), 有抽象方法(计算面积, 计算周长)

//定义正方形类, 继承于形状类

//定义长 方形类, 继承于形状类

```
1  Rectangle rectangle=new Rectangle(5);
2  System.out.println( rectangle.acreage());
3  System.out.println(rectangle.perimeter());
4  接口:把公共的属性和方法封装成一个特定的功能集合
```

六. 接口的格式

/*访问修饰符interface接口类{

属性1

属性2

...

方法1

方法2

}

*/

接口的特点;

1. 接口中的方法 默认都是public abstract

2. 接口中的属性默认都是public static final

3. 一个类实现接口,有两种选择:a. 重写接口中的方法b. 本身也是抽象的

4. 接口不能实例化对象

练习:定义飞机类和马类,行为:拉货,拉人

类,接口的关系:

类和类之间:单继承　extends

类和接口之间:多实现　implements

接口和接口之间:多继承　extends