语法学习：将对象传入方法内然后赋值成null，为什么执行完方法之后，值不是null？

因为java只有一种传递参数的方式：**值传递**。在值传递中，实参的值被传给形参，方法体内对形参的任何赋值操作都不会影响到实参。

//解释为head=null不起作用？ 因为Java方法只有值传递，对于引用传递都是地址，

//形参与实参都指向同一个对象地址，形参试图改变指向地址，不改变实参；

//但是形参和实参对该指向的对象的操作是等价的，比如形参引用可以改变对象的属性。

## 例子1：参数为基本数据类型：

我们知道，传递int时，传递的就是值，这个很好理解。

public void add(int a){

a = a+1;

}

@Test

public void test3(){

int a = 2;

add(2);

**System.out.println(a);//2**

}

add方法不会改变a的值。

## 例子2：参数为引用类型

传递的尽管是个引用类型，本质上传递的也是**该引用指向的地址**。如例子:

public void modifyObject(Object obj){

obj = new Object();//IDEA默认提示，obj永远不会赋值为new Object()

}

@Test

public void test4(){

Object obj1 = new Object();

Object obj2 = obj1;

System.out.println(obj1 == obj2);//true指向一个对象

modifyObject(obj1);//试图让obj1指向新的Object，失败

System.out.println(obj1 == obj2);//true

obj1 = new Object();

System.out.println(obj1 == obj2);//false

}

那**传递引用**怎么会改变呢？

其实，传递引用不会改变该引用的指向的地址，但是在方法内部可以通过该引用修改引用所指向的对象，如**改变引用指向对象的属性**，如例子3。

## 改变引用指向对象的属性

public class **User**{

**String name;**

**String sex;**

public User(String name, String sex) {

this.name = name;

this.sex = sex;

}

@Override

public String toString() {

return "User{" + "name='" + name +

'\'' + ", sex='" + sex + '\'' + '}';

}

}

public void modifyName(User user){

user.name = "中国人";

}

@Test

public void test5(){

User user = new User("陈红佳","女");

System.out.println("user = " + user);

modifyName(user);

System.out.println("user = " + user);

}