参考文档：<https://blog.csdn.net/zj15527620802/article/details/88547914>

Hashcode是根据对象的内存地址来生成的。用来提高对象在散列标的查询效率

p1和p2的equals相同，我们认为是两个对象相等，但是这两个对象竟然同时出现再hashSet中，hashSet中是不会出现两个相同的元素的。

那问题在哪里？

hashset在添加一个元素的时候，会做如下判断：

1、如果添加元素的hashcode相等并且 对象equals为true或对象== 时，则认为是同一个元素，不添加到新元素中。

2、如果不符合上述条件，则认为是一个新元素，添加到set中。

所以，虽然p1和p2equals比较时相等，但是hashcode并不一样，所以在往set中添加的时候认为是两个不同的元素，所以才会出现了p1和p2同时在set中的情况。

****引申出几个经常在面试中问到的问题：****

     1、两个对象，如果a.equals(b)==true，那么a和b是否相等？

          相等，但地址不一定相等。

     2、两个对象，如果hashcode一样，那么两个对象是否相等？

          不一定相等，判断两个对象是否相等，需要判断equals是否为true。

重写hashcode方法，一般要重写equels方法。不然会引起HashMap，HashSet等这类散列表功能异常。

Equals方法判断两个对象是否相等

Hashcode重要用来在散列表中提高查询效率。

我们向一个散列表中添加某个元素，集合会首先调用hashCode方法，这样就可以直接定位它所存储的位置，

若该处没有其他元素，则直接保存。

若该处已经有元素存在，就调用equals方法来匹配这两个元素是否相同，

相同则不存，不同则散列到其他位置。

这样处理，当我们存入大量元素时就可以大大减少调用equals()方法的次数，极大地提高了效率。