equals()方法

equals是Object类提供的方法之一，众所周知，每一个java类都继承自Object类，
所以说每一个对象都有equals这个方法。而我们在用这个方法时却一般都重写这个方法。

equals()方法源代码

```
public boolean equals(Object obj) {
    return (this == obj);
}
```

从这个方法中可以看出，只有当一个实例等于它本身的时候，equals()才会返回true值。通俗地说，此时比较的是两个引用是否指向内存中的同一个对象，也可以称做是否实例相等。而我们在使用equals()来比较两个指向值对象的引用的时候，往往希望知道它们逻辑上是否相等，而不是它们是否指向同一个对象——这就是我们通常重写这个方法的原因。（String类内部重写过equals方法）

重写equals满足规则：

自反性：x.equals(x) 一定是true

对null：x.equals(null) 一定是false

对称性：x.equals(y) 和 y.equals(x)结果一致

传递性：a 和 b equals , b 和 c equals，那么 a 和 c也一定equals。

一致性：在某个运行时期间，2个对象的状态的改变不会不影响equals的决策结果，那么，在这个运行时期间，无论调用多少次equals，都返回相同的结果。

hashCode()方法

hashCode()返回该对象的哈希码值，该值通常是一个由该对象的内部地址转换而来的整数，它的实现主要是为了提高哈希表(例如java.util.Hashtable提供的哈希表)的性能。

hashCode()的返回值和equals()的关系如下：

如果x.equals(y)返回“true”，那么x和y的hashCode()必须相等。

如果x.equals(y)返回“false”，那么x和y的hashCode()有可能相等，也有可能不等。

equals()和hashCode()方法是用来在同一类中做比较用的，尤其是在容器里如set存放同一类对象时用来判断放入的对象是否重复。

Object类中定义了一个hashCode()方法来返回每个Java对象的哈希码，当从HashSet集合中查找某个对象时，Java系统首先调用对象的hashCode()方法获得该对象的哈希码表，然后根据哈希吗找到相应的存储区域，最后取得该存储区域内的每个元素与该对象进行equals方法比较；这样就不用遍历集合中的所有元素就可以得到结论

当我们重写一个对象的equals方法，就必须重写他的hashCode方法，不过不重写他的hashCode方法的话，Object对象中的hashCode方法始终返回的是一个对象的hash地址，而这个地址是永远不相等的。所以这时候即使是重写了equals方法，也不会有特定的效果的，因为hashCode方法如果都不想等的话，就不会调用equals方法进行比较了，所以没有意义了。