“==”比较双方引用是否相同。

如果是基本类型则表示**值相等**。

如果是引用类型（比较引用的对象在堆中的地址）则表示**地址相等**即是同一个对象。

equals方法比较值是否相同

所有类从Object类继承equals，比较的是是否是同一个对象。如果没有对equals方法进行重写，则比较的是引用类型的变量所指向的对象的地址；

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

不能作用于基本数据类型的变量。Object 的 equals 方法默认就是比较两个对象的hashcode，是同一个**对象的引用**时返回 true 否则返回 false。（默认equals等同于＂＝＂）。**但如String、Date，Integer等类对equals方法进行了重写的话，比较的是所指向的对象的内容。**

字符串相等的判断：

1. equals方法用来检测两个字符串内容是否相等。如果字符串s和t内容相等，则s.equals(t)返回true，否则返回false。
2. 要测试两个字符串除了大小写区别外是否是相等的，需要使用equalsIgnoreCase方法。
3. 判断字符串是否相等不要使用"=="。

equals方法的使用：

Object的equals方法容易抛空指针异常，应使用常量或确定有值的对象来调用 equals。null可以进行比较，但不能用来调用方法。

```
String str = null;
str.equals("SnailClimb")
//会抛出异常，改为“SnailClimb”.equals(str);
```

更推荐使用 java.util.Objects#equals(JDK7 引入的工具类)。

```
Objects.equals(null,"SnailClimb");// false
```

源码：

```
public static boolean equals(Object a, Object b) {
        // 避免出现空指针异常。
        return (a == b) || (a != null && a.equals(b));
    }
```