```java
// Example 5.3 joi/examples/EqualsDemo.java
//
//
// Copyright 2003 Bill Campbell and Ethan Bolker

// A class illustrating == and equals().
//
// %> java EqualsDemo
// Different objects, same field:
// e1 == e1LookAlike ->         false
// e1.equals( e1LookAlike ) ->  true
// Same object:
// e1 == e1Too ->               true
//
// Different ArrayLists with equal (but not ==) elements:
// alist0 == alist1 ->      false
// alist0.equals(alist1) -> true
//
// Different TreeMaps with equal keys
// mapping to equal (but !=) values:
// tmap0 == tmap1 ->      false
// tmap0.equals(tmap1) -> true
//
// tmap0.toString() -> {sillykey = EqualsDemo value 1}
// tmap1.toString() -> {sillykey = EqualsDemo value 1}
// Are these Strings == ?    false
// Are these Strings equal ? true

import java.util.ArrayList;
import java.util.TreeMap;

public class EqualsDemo
{
    // Properties of an EqualsDemo object.
    //
    // Override equals: two of these objects are equal if
    // their integer field has the same (i.e. ==) value.
    //
    // When you override equals it’s customary to override
    // toString too, so that equal objects return equal
    // Strings, so we do.

    private int field;

    public EqualsDemo( int field )
    {
        this.field = field;
    }

    public boolean equals( Object other )
    {
        return (other instanceof EqualsDemo)
            && (this.field == ((EqualsDemo)other).field);
    }

    public String toString()
```

```java
    {
        return " EqualsDemo value " + field;
    }

    public static void main( String[] args )
    {
        Terminal t = new Terminal();

        // EqualsDemo object == vs equals()
        EqualsDemo e1          = new EqualsDemo( 1 );
        EqualsDemo e1LookAlike = new EqualsDemo( 1 ); // same field.
        EqualsDemo e1Too       = e1;                  // same object

        t.println("Different objects, same field:");
        t.println("e1 == e1LookAlike ->         " + (e1 == e1LookAlike));
        t.println("e1.equals( e1LookAlike ) ->  " + e1.equals( e1LookAlik
        t.println("Same object:");
        t.println("e1 == e1Too ->               " + (e1 == e1Too));
        t.println();

        // Arrays and Maps
        ArrayList alist0 = new ArrayList();
        ArrayList alist1 = new ArrayList();

        alist0.add( e1);
        alist1.add( e1LookAlike );

        t.println( "Different ArrayLists with equal (but not ==) elements
        t.println( "alist0 == alist1 ->      "  + (alist0 == alist1));
        t.println( "alist0.equals(alist1) -> "  + alist0.equals(alist1));
        t.println();

        TreeMap tmap0 = new TreeMap();
        TreeMap tmap1 = new TreeMap();

        tmap0.put( "sillykey ", e1 );
        tmap1.put( "sillykey ", e1LookAlike );

        t.println( "Different TreeMaps with equal keys" );
        t.println( "mapping to equal (but !=) values:" );

        t.println( "tmap0 == tmap1 ->      " + (tmap0 == tmap1));
        t.println( "tmap0.equals(tmap1) -> " + tmap0.equals(tmap1));
        t.println();

        // Test Strings for == and equal
        String s0 = tmap0.toString();
        String s1 = tmap1.toString();
        t.println( "tmap0.toString() -> " + s0);
        t.println( "tmap1.toString() -> " + s1);
        t.println( "Are these Strings == ?    " + (s0 == s1) );
        t.println( "Are these Strings equal ? " + (s0.equals(s1)) );
    }
}
```