```java
package hyperDap.base.types.value;

/**
 * An immutable pair of number values. While there is no programmatic relation between the two
 * values, they are assumed to be related usually as a yValue dependent on an xValue, as if
 * representing a single point on a one dimensional function.
 *
 * @author soenk
 *
 * @param <T> The subclass of {@link Number} that the values should be stored in.
 */
public final class ValuePair<T extends Number> {

  private final T xValue; // independentValue
  private final T yValue; // dependentValue

  public ValuePair(T independentValue, T dependentValue) {
    this.xValue = independentValue;
    this.yValue = dependentValue;
  }

  /**
   * Returns the first of the two stored values, which is considered the independent value or
   * {@code xValue}.
   *
   * @return {@code xValue}
   */
  public T getX() {
    return this.xValue;
  }

  /**
   * Returns the second of the two stored values, which is considered the dependent value or
   * {@code yValue}.
   *
   * @return {@code yValue}
   */
  public T getY() {
    return this.yValue;
  }

  /**
   * Checks whether another Object {@code o} has <b>exactly</b> the same values as this instance.
   * <p>
   * Currently only applies to {@link ValuePair} instances, with arrays of the form
   * {@code <xValue,yValue>} and {@link java.util.Collection Collections} of this form are
   * considered for the future.
   *
   */
  @Override
  public boolean equals(Object o) {
    ValuePair<T> pair;
    try {
      pair = this.castToThis(o);
    } catch (ClassCastException e) {
      return false; // Could add option for {xValue,yValue} array or collection.
    }
    if (pair.getX().equals(this.xValue) == false) {
      return false;
    }
    if (pair.getY().equals(this.yValue) == false) {
      return false;
    }
    return true;
  }

  /**
```

```
68       * Helper method to cast Objects to {@link ValuePair} of the same type {@code <T>}
         as this
69       * instance
70       *
71       * @param o
72       * @return
73       */
74      @SuppressWarnings("unchecked")
75      private ValuePair<T> castToThis(Object o) {
76        return (ValuePair<T>) o;
77      }
78
79      //
80      //
         ********************************************************************************
         *********************************
81
82      /**
83       * A main for testing purposes that should be removed upon completion.
84       *
85       * @param args
86       */
87      public static void main(String[] args) {
88
89      }
90    }
91
```