**package** generics;

**public** **class** UnorderedPair<T> **extends** Pair<T>

{

**public** UnorderedPair()

{

setFirst(**null**);

setSecond(**null**);

}

**public** UnorderedPair(T firstItem, T secondItem)

{

setFirst(firstItem);

setSecond(secondItem);

}

**public** **boolean** equals(Object otherObject)

{

**if** (otherObject == **null**)

**return** **false**;

**else** **if** (getClass( ) != otherObject.getClass( ))

**return** **false**;

**else**

{

UnorderedPair<T> otherPair =

(UnorderedPair<T>)otherObject;

**return** (getFirst( ).equals(otherPair.getFirst( ))

&& getSecond( ).equals(otherPair.getSecond( )))

||

(getFirst( ).equals(otherPair.getSecond( ))

&& getSecond( ).equals(otherPair.getFirst( )));

}

}

**public** **static** **void** main(String[] args)

{

UnorderedPair<String> name = **new** UnorderedPair<String>("Kim", "Tom");

UnorderedPair<String> name2 = **new** UnorderedPair<String>("Tom", "Kim");

System.***out***.println("first=" + name.getFirst());

System.***out***.println("second=" + name.getSecond());

**if** (name.equals(name2))

System.***out***.println(name.toString() + " is the same as " + name2.toString());

**else**

System.***out***.println(name.toString() + " is NOT the same as " + name2.toString());

UnorderedPair<Integer> numbers = **new** UnorderedPair<Integer>(1,2); // Automatic Boxing

UnorderedPair<Integer> numbers2 = **new** UnorderedPair<Integer>(2,1); // Automatic Boxing

System.***out***.println("first=" + numbers.getFirst());

System.***out***.println("second=" + numbers.getSecond());

**if** (numbers.equals(numbers2))

System.***out***.println(numbers.toString() + " is the same as " + numbers2.toString());

**else**

System.***out***.println(numbers.toString() + " is NOT the same as " + numbers2.toString());

}

}

Output:

first=Kim

second=Tom

[first:Kim second:Tom] is the same as [first:Tom second:Kim]

first=1

second=2

[first:1 second:2] is the same as [first:2 second:1]