**package** generics;

**public** **class** TwoTypePair<T1, T2>

{

**private** T1 first;

**private** T2 second;

**public** TwoTypePair()

{

first = **null**;

second = **null**;

}

**public** TwoTypePair(T1 firstItem, T2 secondItem)

{

first = firstItem;

second = secondItem;

}

**public** **void** setFirst(T1 newFirst)

{

first = newFirst;

}

**public** **void** setSecond(T2 newSecond)

{

second = newSecond;

}

**public** T1 getFirst()

{

**return** first;

}

**public** T2 getSecond()

{

**return** second;

}

**public** String toString()

{

**return** ( "[first: " + first.toString() + " second: " + second.toString() +"]" );

}

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

{

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

**return** **false**;

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

**return** **false**;

**else**

{

TwoTypePair<T1, T2> otherPair =

(TwoTypePair<T1, T2>)otherObject;

**return** (first.equals(otherPair.first)

&& second.equals(otherPair.second));

}

}

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

{

TwoTypePair<String, Integer> name = **new** TwoTypePair<String, Integer>("Kim", 30);

TwoTypePair<String, Integer> name2 = **new** TwoTypePair<String, Integer>("Kim", 25);

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());

}

}

Output:

first=Kim

second=30

[first: Kim second: 30] is NOT the same as [first: Kim second: 25]