# 15.04 Virtual Lecture Notes

Ms. Mills wants to be able to compare her jeans. So she is going to modify her BasicJean class to implement the Comparable<T> interface.

First she creates a copy of BasicJean and calls it BasicJean2. Next, she adds to the first line of the class.

public abstract class BasicJean2 implements Comparable<BasicJean2>

When implementing Comparable<T>, the T is replaced with the class name that implements it. In this case, we replace T with BasicJean2. Any time you implement Comparable<T>, you must do the same thing.

Secondly, we add method compareTo(). We could have let any class that extends BasicJean2 implement it, but since all of Ms. Mills' clothing items are going to be compared using size, we will implement it here. BasicJean2 adds a pantSize instance variable, a modified constructor, and a new method getSize().

```
private int numPockets; private int pantSize;

/**
  * Constructor for objects of class BasicJean
  */ public BasicJean2(int n, int s) {
        // initialise instance variables numPockets = n; pantSize = s;
} public abstract void design();

public int getNumPockets() {
        return numPockets; } public void setNumPockets(int n) {
      numPockets = n; } public int getSize() {
      return pantSize; }
```

Now, according to the rules for writing a correct compareTo() method, it must satisfy the following three rules.

1 Return a negative number if, when comparing two pairs of jeans, a and b, a has a smaller size than b.
2 Return a zero if a has the same size as b.
3 Return a positive integer if a has a larger size than b.

Consequently, we write our method like this:

```
public int CompareTo(BasicJean2 obj) { if (pantSize < obj.pantSize)
return 1; else if (pantSize == obj.pantSize) return 0; else return 1; }
```

Now any class that inherits from BasicJean can use this method.

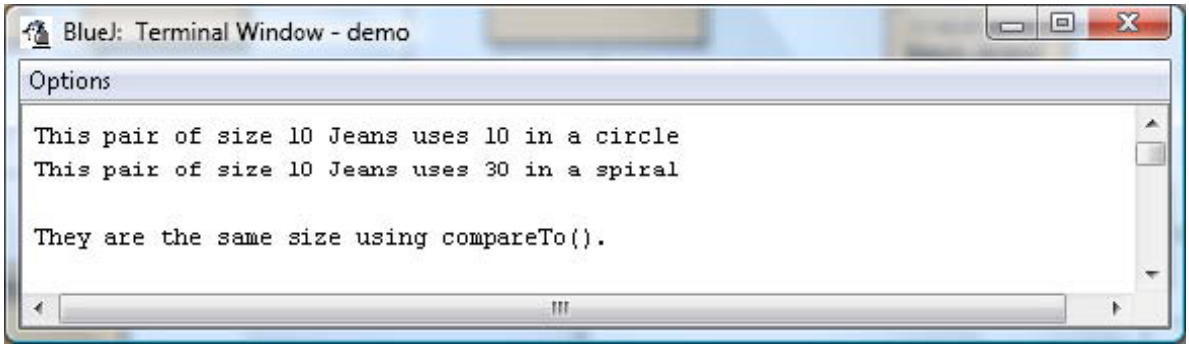
To test it out, we copied BeadedJean and renamed it BeadedJean2. BeadedJean2 extends the BasicJean2 class. BeadedJean2's constructor is modified to call BasicJean2 constructor appropriately.

```
public BeadedJean2(int np, int nb, int s)
{ // initialise instance variables super(np, s); numBeads = nb;
}
```

We will now modify the toString() method to output a size. Notice that since compareTo() only looks at size, two pairs of Beaded jeans are the same if they have the same size.

So if we create a new test class called Tester2 that can test it out, it will only compare them based on size alone. So a BeadedJean with a circle and size 10 will be considered the same as one with a spiral and size 10.

Take a look at the output of Tester2: Now, because of the Beaded design, we would not say that two size 10 jeans are necessarily the same. compareTo() just tells us they are the same size. Because they would also have to the same pattern in order to be equal, we would need to write an equals() method to reflect that. If we could order the jeans based upon Beaded design, then we could just override the compareTo() method, but that does not make sense to Ms. Mills.

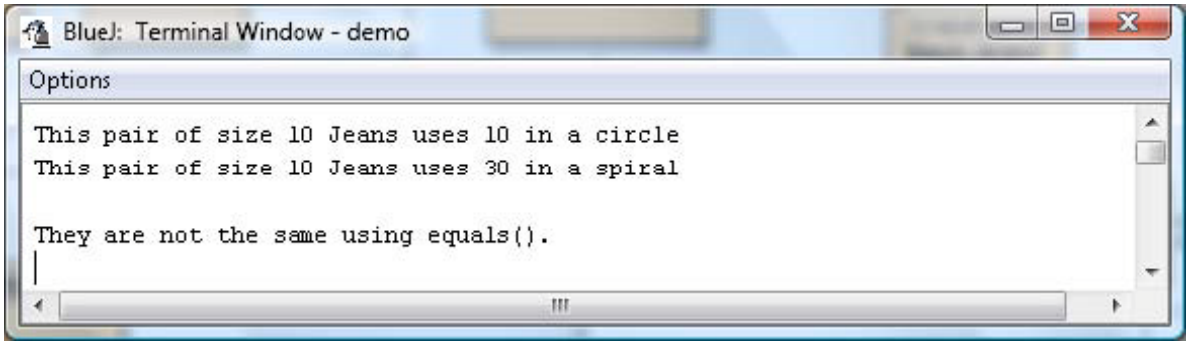



So, we will just write a new equals() method that returns true if compareTo() returns 0 and the patterns are the same. If we could have ordered them based on design and size, then we would have to override compareTo() and then make equals() return true or false based upon compareTo() returning a 0 or not.

```
public boolean equals(BeadedJean2 obj) { if ((compareTo(obj)
== 0) && (pattern.equals(obj.pattern))) return true; else return
false; }
```

After adding it and then making tester3 to test for the new test for equality, we get this output:

- Download the BasicJean2.java file to your module 15 demo programs directory and open it.
- Make sure you understand it before you continue.
- Download the BeadedJean2.java file to your module 15 demo programs directory and open it.




- Make sure you understand it before you continue.
- Download the tester2.java file to your module 15 demo programs directory and open it.
- Make sure you understand it before you continue.
- Download the tester3.java file to your module 15 demo programs directory and open it.
- Make sure you understand it before you continue.