APPENDIX D - INPUT FROM KEYBOARD

Terry Marris 7 June 2001

Reading input from a keyboard in a DOS environment is useful. Java does not provide an easy way of doing this. One way is to have this Java file in your current working directory.

```
/* Keyboard.java
   Terry Marris  9 Mar 2000
 */

import java.io.*;

public class Keyboard {
  public static String readString()
  {
    byte buffer[] = new byte[80];
    String inputString = null;
    try {
      System.in.read(buffer);
      inputString = new String(buffer);
      inputString = inputString.trim();
    }
    catch (IOException e) {
      System.out.println(e);
    }
    return inputString;
  }
}
```

An example of its use is given below. The program enters two numbers and outputs their sum.

```
/* TestInput.java
   Terry Marris  2 August 2000
*/

public class TestInput {
  public static void main(String[] args)
  {
    System.out.print("Enter a number -> ");
    String s1 = Keyboard.readString();  // read a string
    Integer n1 = new Integer(s1);       // String to Integer
    int num1 = n1.intValue();           // Integer to int

    System.out.print("Enter another number -> ");
    String s2 = Keyboard.readString();
    Integer n2 = new Integer(s2);
    int num2 = n2.intValue();

    System.out.println("Their sum is " + (num1 + num2));
  }
}
```

An example run is

```
Enter a number -> 123
Enter another number -> 456
Their sum is 579
```

Notice the use of *System.out.print()* (rather than *System.out.println()*) in

```
System.out.print("Enter a number -> ");
String s1 = Keyboard.readString();
```

This ensures that the input is on the same line as the prompt (look at the example run). If you used

```
System.out.printline("Enter a number -> ");
String s1 = Keyboard.readString();
```

the output would look like

```
Enter a number ->
123
```

(with the input on the line below the prompt) which is very ugly.

Sometimes you might find that, when testing a class, output scrolls off the screen. One way to hold the output so that you have a chance to read it is to use the *readString()* method.

First, create your own *pause()* method in your test class. It might look something like this.

```
public void pause()
{
  System.out.print("Press return to continue ... ");
  Keyboard.readString();
}
```

Then, in your *main()* method for example, you would write *pause()* wherever you want the screen display to stop scrolling. For example

```
public static void main(String[] args)
{
  Integer five = new Integer(5);
  Integer seven = new Integer(new String("7"));

  System.out.println((five.compareTo(seven) < 0));
  System.out.println(!five.equals(seven));
  pause();

  System.out.println((12 == five.intValue() +
                      seven.intValue()));
  pause();

  String s1 = five.toString();
  String s2 = seven.toString();
  System.out.println(!s1.equals(s2));
}
```