IST 256
Lab
Week 9, Wednesday, March 24, 2010

1. Writing a program that reads string data from a file

In this program, we will write a Java application that can read words from a file and put each word into a String variable.

a. Start by **creating a Java application** and name it something like FileReadWords. As usual, NetBeans will bring up a program source code window where there is a class called Main and a main method where we will put our code to read a file.

b. Save a data file for the program to read. From your browser, save the file called xanadu.txt from Monday and put it into the folder for your NetBeans project. (In Firefox, to save a file, right click on the link and select Save Link as …) Then navigate to the folder where you keep your projects and go into the newly created folder for FileReadWords and save the xanadu.txt file there.

c. Write the application that reads words from a file.
At the top of the program, right after the package statement, add import statements:

```
import java.io.*;
import java.util.Scanner;
```

From Monday's web page click on the link called WordFileRead. Copy the part of the main method from the line that starts "Scanner s … " on down to the end, but not including the last two brackets.

Paste this program into your NetBeans main method. Fix the tabs, spacing and matching brackets. Observe the use of the Scanner functions hasNext() to see if there is another word to read, and next() to actually read the word as a String.

Run the program and observe its results. By default, the Scanner is using "whitespace" to separate words. What do you think is meant by whitespace?

d. Cause an error.
In the program, change the name of the file and try to run the program. Observe the error that you get and write the name of the error here:

e. Change the file name back to be the correct name.

2. Writing a program that reads integer data from a file

In this program, we will write a similar Java application that can read integers from a file and put each one into an int variable.

a. Start by **creating a Java application** and name it something like FileReadInts. As usual, NetBeans will bring up a program source code window where there is a class called Main and a main method where we will put our code to read a file.

b. Save a data file for the program to read. From your browser, save the file called Feb05temps.txt from today and put it into the folder for your new NetBeans project

c. Write the application that reads words from a file.
At the top of the program, right after the package statement, add import statements:

```
import java.io.*;
import java.util.Scanner;
```

From the program that you just ran, copy the same code into the new application's main method.

Change the name of the file to be "Feb05temps.txt".

Instead of a String variable to read items from the file, declare an int variable to read integer numbers from the file. Change the Scanner functions to use:

```
s.hasNextInt()
s.nextInt()
```

Change the code as necessary to use these functions to read values into the int variable and to print them.

Run the program and observe the results.

3. Writing a GUI application that reads from a file and processes the results

a. Start by **creating a Java GUI application** and name it something like FileReadTemps. Then for that project, create a new jFrameForm to get the GUI window and also set the GUI to be the main class of the project.

b. Save a data file for the program to read. Copy the file called Feb05temps.txt from the folder for the FileReadInts project and put it into the folder for your new NetBeans project. (Or recopy it from the web page.)

c. Create the GUI:
Make a button that will read the data from the file, and a label to report the status of the file read. Also make a button to compute the average of the temperatures in the file and a label to put the result. The form can look something like this:

```
        |__ Read Temperatures from File Feb05temps.txt__|    File Status

        |__ Compute Average Temperature__|                   Average
```

For each button, right click on Event -> Action -> actionPerformed to get a button actionPerformed method.

d. Start the GUI application:
At the top of the file, after the package statement, put the same import statements.

Then just before the first actionPerformed method, declare an array of temperatures that is of size 31:

```
         //   will hold at least 31 temperatures
        int [] temperatures = new int[31];
         // number of temperatures actually stored in array (between 0 and 31)
        int numtemps = 0;
```

e. Write the button method that reads words from a file.
Inside the button method for the button that reads from the file, include the following code. (Note that the code doesn't fit on one page, so you will have to get some of it from the additional page.)

```
BufferedReader inputStream = null;
        Scanner inputScanner = null;
        // count the number of temperatures read from the file
        int count = 0;
        // variable to hold each temperature
        int temp;

        try
        {
           // create the input stream and the scanner
           inputStream = new BufferedReader(new FileReader("Feb05temps.txt"));
           inputScanner = new Scanner(inputStream);

           // loop as long as there is still input in the file to read
           while (inputScanner.hasNext())
           {
              // test if the next token in the file is an integer
              if (inputScanner.hasNextInt())
              {
                 // read the next integer in the file
                 temp = inputScanner.nextInt();
                 // store the temperature in the array and increment the count
                 temperatures[count] = temp;
                 count++;

                 // for debugging, we print each number as we read it:
                 System.out.println("Day " + count + ":  " + temp);
              }
              else
              {
                 // if it's not an integer, skip over it
                 inputScanner.next();
              }
           }  // end while loop to read from file
           // after file is read, save the number of items
           numtemps = count;
        }
        catch (IOException e)
        {
           // print the exception
           System.out.println(e.toString());
           // end the program with an error status
           System.exit(-1);
        }
```

```
finally
    // good idea to always close the input file
    {  inputScanner.close(); }
    // display a status message
    fileStatusLabel.setText("File read done");
```

Run this program and test this button. There will be a class discussion about the control flow of this section of code. And note the use of the System.exit method to force the program to quit if there are I/O errors.

f. Write the button method that computes the average.
In the second button method, use the array temperatures and the number numtemp to compute the average of the array.

Computing the average will be a little different from normal in that not all the elements of the array may be defined. You must write the loop that sums the elements of the array to go from 0 to less than numtemps. Then divide the sum to compute the average and put it into the label.

If you have time, you can control the number of digits that print after the decimal point in the average. Add a decimal formatter to this button method that prints only 3 digits after the decimal point (for example).

```
        DecimalFormat df = new DecimalFormat("00.000");
```

Then use the function df.format to convert the average to a String with this format and display it in the label.

g. Test your program
What is the average of the temperatures in this array?