IST 256
Lab
Week 7 – Wednesday, October 13, 2010

1. Working with arrays in an application

In this program, we will use a Java application to create an array and write a program to compute the average of the array. To put values in the array, we will use the Java random number generator. This is a program that if you give it a number bound, say 100, it can generate a sequence of random numbers that are all between 0 and 99.

A. Start writing the program
Create a new Java application and call it something like ArrayFunctions.

We can place other methods in this program, such as this method which will create an array of random numbers. First **place this import statement between the package statement and the public class Main line**, which will make the Java Random class available to the program.

```
import java.util.Random;
```

Then place this method definition **between the public class Main line and the public state main method header**.

```
/*
 * This method creates an array of random integers.
 *    The random numbers are created by a Java random number generator:
 *    given 100, the numbers are uniformly distributed between 0 and 99.
 * Parameter:  sizearray specifies the number of elements of the array
 * Result:     returns an array of random integers
 */
private static int [] getRandomNumbers (int sizearray)
{
   // create the array of the specified size
   int [] numberarray = new int[sizearray];
   // create the Java random number generator
   Random randomGenerator = new Random();

   // for each element of the array, get the next random number
   for (int index = 0; index < sizearray; index++)
   {
      numberarray[index] = randomGenerator.nextInt(100);
   }
   return numberarray;
}
```

There are several interesting features about this method, including the fact that it can return an entire array as a value by giving the return type of integer array, “int [ ]”.

Now for the body of the main method, place the following code, which uses the getRandomNumbers method to create an array of values and display them.

```
// create an array of length 10 that is initialized with random integers
int [ ] numbers = getRandomNumbers(10);

// print these numbers
for (int i = 0; i < numbers.length; i++)
{
   System.out.println("Num " + i + ":   " + numbers[i]);
}
```

Test this program several times and observe the numbers that are generated. Write one set of numbers here (from **one run** of the program).

Array Values:										

B. Add code to compute the average of the array
Use the solution developed in class to add code to compute the average of all the values in the array. Add this code to the main method after your other code.

Test this program at least three times and write three different average numbers here (from **three runs** of the program).

Three runs of the program:	
First average:	
Second average:	
Third average:	

2. Creating a GUI application to display a sequence of numbers and the squares of those numbers

This program is designed for you to get more experience working with arrays and with methods. Start a new project and name it something like TableDisplay. Then create a jFrame Form and name it something like TableDisplayGUI and set it to be the main class of the project.
Create a GUI design that has a button to display the table of numbers as computed. We are first going to compute the squares of all the numbers.

|__ Display Table __| *(button)*

Numbers Squares *(two labels as column headers)*
numbers results *(two more labels to show the numbers)*

Make the event actionPerformed method for the button, and create the program in the source window.

First, add the following import statement between the package statement and the public class statement so that we can use the Java programs for decimal number output formatting.

```
import java.text.*;
```

Next copy this method for computing the square of a number right before the button actionPerformed method.

```
/*
 * method to compute the square of a number
 */
private double computeSquare(int num)
{
   double square;
   square = num * num;
   return square;
}
```

Make sure that the curly brackets match!

Now copy this code to the body of the actionPerformed method, right after the TODO comment, and again make sure that the curly brackets match. Also, you may need to retype the quotes or rename the labels.

```
// arrays for two sets of numbers
//     and for two strings to display them as multiline labels
int[] numbers = new int[23];
double[] results = new double[23];
String numbertext = "<html>", resulttext = "<html>";

// create a number formatter that will display only 2 digits after the decimal point
DecimalFormat df = new DecimalFormat("####.00");

// set the numbers array to have numbers between 2 and 220,
//    at intervals of 10
// and add each number as a line on a string to be displayed
for (int i = 0; i < numbers.length; i++)
{
   numbers[i] = (i * 10) + 2;
   numbertext = numbertext + "<p>" + String.valueOf(numbers[i]);
}

// set the results array to have the square of each number
// and add each result as a line on a string to be displayed
```

```
for (int i = 0; i < results.length; i++)
{
   results[i] = computeSquare(numbers[i]);
   resulttext = resulttext + "<p>" + df.format(results[i]);
}

// display the strings in labels
jLabel4.setText(numbertext);
jLabel5.setText(resulttext);
```

Test your program

Run the program. Do the results look reasonable? Write the square of 212 here:

3. Change this GUI application to display Fahrenheit and Celsius numbers.

Now we are going to change this application to have different results in the table.

First change the GUI form so that
the label Numbers now says Fahrenheit, and
the label Squares now says Celsius.

Now write another method called computeCelsius. This method should also take an integer parameter which will be a temperature in Fahrenheit. The method should return a double value which is the equivalent temperature in Celsius. If there is an integer variable named fahrenheit and a double variable named celsius, the formula for converting a Fahrenheit number to Celsius an be written in Java as:

```
celsius = (5.0 / 9.0) * (fahrenheit - 32);
```

Now change the actionPerformed method to display a table of temperature in Fahrenheit with the equivalent temperatures in Celsius. Keep the numbers array defined as it is, but change the results array so that each element is obtained by calling the computeCelsius method with an element from the numbers array.

Test your program

Run the program. Do the results look reasonable? Write some values from the table here:

Fahrenheit number	Equivalent Celsius number
32 (freezing)	
72	
212 (boiling pt. of water)	

Hand in your lab sheet at the end of class.