IST 256
Lab
Week 7 – Wednesday, October 12, 2011

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 an upper 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 ArrayAverage.

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 static main method header**.

```
/*
 * This method creates an array of random integers.
 *    The random numbers are created by a Java random number generator:
 * Given an upper bound 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. (Note that the average is computed by adding up all the values in the array, using a for loop, and then dividing the total by the number in the array to get the average. The average should be of type double.)

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:	

C. Add code to compute the maximum value of the array

Use the solution developed in class to add code to compute the maximum of all the values in the array. Add this code to the main method after your other code. (Note that the maximum is computed by starting the maximum so far to be the first value in the array, and then using a for loop to compare with the remaining values of the array and setting the maximum so far to be any larger value.)

Test this program at least three times and write three different maximum 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

a. Create the GUI

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)*

Drag the last two labels to be long and narrow to display a result with 23 lines.

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

b. Writing the program

1. 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.*;
```

2. 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!

3. Write the code for the button actionPerformed method.

Write **two array declarations** for arrays of length 23. Make the name of the first array be *numbers* and its values be of type *int* and the second array be named *results* and its values be of type *double*.

Now **declare two Strings** to show the values in the arrays. Since we want this to be in a table with the next number on a new line, make the two strings have the html tags for multi-line label displays.

```
// strings to display the numbers and results array values
String numbertext = "<html>", resulttext = "<html>";
```

Next **create a number formatter** to display only 2 significant digits.

```
// number formatter displaying only 2 digits after the decimal point
DecimalFormat df = new DecimalFormat("####.00");
```

Initialize the numbers array to have the numbers 0, 10, 20, 30, etc, by **writing a for loop** with an index variable *i* that goes over every element of the array and gives it the value of 10 * i. Also inside the loop, add this statement to put the every value of the array numbers appended to the String numbertext:

```
numbertext = numbertext + "<p>" + String.valueOf(numbers[i]);
```

Now **write another for loop** where the index variable *i* goes over every element of the results array and assigns the value of the array by calling the method computeSquare(numbers[i]). Also, inside this loop, add this statement to every value of the array results appended to the String resulttext.

```
resulttext = resulttext + "<p>" + df.format(results[i]);
```

Finally, display the two strings in the two labels on the GUI.
Test your program

4. Add another label to the GUI to show the average of the squares. Add the code into the same button that will **compute the average of the results array** and display it in this label.

Test your program and write the average of the squares here:

Hand in this lab sheet with the code that you wrote for the TableDisplayGUI by Monday, October 10.