IST 256
Lab
Week 7 – Tuesday, February 28, 2012

1. Java application programs using arrays

Create a project called TestArrays1 and leave it as a Java application (do not create a Form). We are going to create and run an application similar to the one shown as TestArrays1 on the class web page.

Note that there is a method called "main" that looks like this:

```
public static void main(String[] args) {
// TODO code application logic here
}
```

Inside this method (right after the comment that begins //TODO), first write a declaration (and allocation) of two integer arrays of size 15:

```
// declares (and allocates) integer arrays of size 15
int [ ] numbers = new int[15],
int [ ] squares = new int[15];
```

Now assign all the elements of the numbers array to be the numbers from 1 to 15. Each number can be computed by adding one to the index value of the for loop:

```
// set the elements of the numbers array to be 1 to 15
for (int i = 0; i < numbers.length; i++)
{
    numbers[i] = i + 1;
}
```

Now we can use the elements of the numbers array to set the elements of the squares array. Each element of the squares array will be the square of a number in the numbers array.

```
// set the elements of the squares array to be the numbers 1 to 15 squared
for (int i = 0; i < squares.length; i++)
{
    squares[i] = numbers[i] * numbers[i];
}
```

Print out the corresponding elements of the two arrays:

```
// print out the all the elements of the two arrays
 for (int i = 0; i < numbers.length; i++)
{
```

```
            System.out.println(numbers[i] + " squared is " + squares[i]);
    }
```

Run this program and observe the values.

2. Modify the Java application program using arrays

Now we are going to modify our application to be similar to the one shown as TestArrays2 on the web page.

Change the declaration of the numbers array to initialize the array to a set of 15 numbers:

```
// declares, allocates and initializes the numbers array;
int[] numbers = {7, 5, 3, -4, 17, 23, 16, 2, 0, -12, 2, -1, 0, 7, 9 };
```

Remove the previous for loop that puts values into the numbers array.

Now run the program.

We'll add a part to show how to use array indexing to get a single element of the array:

```
// select an element of the array and display it
int selectindex = 5;
int selectedelement = numbers[selectindex];
System.out.println();    // prints a blank line
System.out.println("The element selected at index "+
            selectindex +  " is "+ selectedelement );
```

Run this program. Set the value of selectindex to be different numbers. What happens if the value is not between 0 and 14?

For example, set selectindex = 15; and write what happens here:

3. Writing a GUI application to use arrays and radiobuttons

Create a new Project named TestArrays3 and make a GUI form named TestArrays3Form. (As usual, change the run property to have the Form as the main class.)

In this project, we will use some predefined arrays for color names. The user will be able to select between regular colors and designer colors. Then they will enter a number between 0 and 9, which is used as a color code, and view the color name given for that code.

Here is a sketch of the user form:

Select Types of Colors

o Regular o Designer (radiobuttons in a buttongroup)

Enter Color Code between 0 and 9 |_____| (textfield)

|__ Select Color __| (button)

Result: (label)

Create the Event -> Action -> actionPerformed method for the button.

Create two arrays with color names

Go to the source tab and in the program, at the top of the actionPerformed method, we will two arrays with color names. Here is the code for the two arrays: (You can use these color names, or you are welcome to create your own.)

```
// regular and designer color arrays
    String[] regularColors = {"white", "beige", "yellow", "orange",
        "red", "purple", "blue", "turquoise", "green", "black"};
    String[] designerColors = {"eggshell", "ecru", "sunshine", "jackolantern",
        "scarlet", "plum", "royal", "teal", "grass", "midnight"};
```

Write the actionPerformed method to first just show regular colors:

1. Declare a variable to hold a string for the result and one to hold an integer for the color code

```
    // strings for the name of the color and the result message
    String colorname, message = "";
    // integer for color code, used as an index into the arrays
    int colorcode;
```

2. Add code to get the number that the user typed in.

```
colorcode = Integer.parseInt(jTextField1.getText());
```

Then use it as an array index to select the name of the color.

```
colorname = regularColors[colorcode];
```

Create a message showing the name of the color and display it

```
// put the message string in the results label
message = "Your color is " + colorname;
 jLabel3.setText(message);
```

3. Test this part of the program with color numbers between 0 and 9. Now type in a number that is not between 0 and 9. What happens?

Extend the actionPerformed method to check that the color number is between 0 and 9:

4. Design and write an if statement that will check that the value of colornumber is between 0 and 9. If the number is in the correct range, assign the message variable as before, but if it is not, assign the message variable to give an alert to the user instead. For example,

```
message = "Please enter a number between 0 and 9";
```

Test your program.

Extend the actionPerformed method to display related colors:

5. Now suppose that we want to display not only the color of the number that the user selected, but also the colors on either side of it, to the left and right. (The colors are in order according to the color wheel, so this will show similar colors.)

Given that the variable colorcode is used as an index into the regular colors array, what we want is to also display the colors at

```
regularColors[colorcode – 1]
```

and

```
regularColors[colorcode + 1]
```

But we must be careful not to use the index colorcode -1 if it will be less than 0, or to use the index colorcode + 1 if it will be greater than 14.

Write appropriate if statements so that after displaying the selected color, we also show

- if colorcode – 1 is less than 0, also display the color at regularColors[colorcode + 1]
- otherwise if colorcode + 1 is greater than 9, also display the color at regularColors[colorcode – 1]
- and otherwise also display both regularColors[colorcode – 1] and regularColors[colorcode + 1]

Test your program.

Extend the actionPerformed method to choose selected colors:

6. Now design and add code to the actionPerformed method so that you test to see which RadioButton is selected.

If RadioButton1 is selected, use the regular colors array to get the color name.

If RadioButton2 is selected, do the same computation as the regular colors, but use the designer colors array to get the color names.

Test your program and write some results here:

Regular or Designer	Color Code	Resulting Color Name(s)

Your lab sheet with attached code must be handed in by Tuesday, March 6.