Labs 3, 4, & 5 In-Review

COMP 110
Summer II 2012

Peter Lincoln
7/9/2012
Announcements

- Labs 3, 4, and 5 have been graded
  - Comments on Sakai

- Grading In-Progress
  - Program 2
  - Midterm
Questions?

Yesterday:
- Midterm
Today in COMP 110

- Review Labs 3, 4, & 5

- Lecture
  - Information Hiding & Encapsulation

- In-Class Exercise
public static void main(String[] args)
{
    int[] array = new int[args.length];
    // Set integer values for initial program setting
    for (int i = 0; i < args.length; i++)
    {
        array[i] = Integer.parseInt(args[i]);
    }
    // Set inputs for user to see, once program is effectively run
    System.out.println("Original array:");
    displayArray(array);
    // Set up integer for Bubble Sort and give the user the required
    // array in the correct order
    int swaps = bubbleSort(array);
    System.out.println("Sorted Array:" );
    displayArray(array);
    // Tell user the number of swaps that are to be executed once program is run
    System.out.println("Number of Swaps: " + swaps);
}

/* TODO: Implement this program here */
* 
* Create an integer array (int[]) such that
* its contents are the numerical values of the
* contents of args. Display that integer array.
* Execute bubblesort and print that integer array. 

Poor Indenting

```java
public class ArrayUtils {

    public static void main(String[] args) {
        // convert args array to integers
        int[] array = new int[args.length];
        for (int x = 0; x < args.length; x++) {
            array[x] = Integer.parseInt(args[x]);
        }

        // Display array
        ArrayUtils.displayArray(array);

        // sort array
        int numberSwaps = ArrayUtils.bubbleSort(array);

        // After sorting output the array
        System.out.println("was sorted to");
        ArrayUtils.displayArray(array);
        System.out.println("Total number of swaps: "+ numberSwaps);
    }

    // use array display method for each value to go through loop
    public static void displayArray(int[] array) {
        for (int x : array) {
            System.out.print(x + " ");
        }
    }

    public static int bubbleSort(int[] array) {
        return 0; // Placeholder for bubble sort implementation
    }
}
```
Poor Indenting

- Hard to read your code

- Can lead to bugs
  - Java does not use indenting for code blocks
  - Java uses `{}`’s for code blocks
    - Use `{}`’s for all conditional blocks and loop bodies!
Fixing Indenting (The Easy Way)

- Step 1: Download and import the COMP 110 Formatter Profile
  - https://sites.google.com/a/cs.unc.edu/2012-summer2-comp110/resources/eclipse-setup#TOC-Eclipse-Settings

- Step 2: Use the menu command: Source | Format
Source | Format

[Image of Eclipse IDE menu with 'Format' selected]

Array of values from the command line (args)

Input arguments to the program

integer array (int[]) such that inputs are the numerical values of the args. Display that integer array. BubbleSort upon that integer array: e.g., bubbleSort(intArray);. Display the sorted array.

Display the array with a call to array(intArray);
Source | Format

- Interprets your code to identify blocks
  - Indents contents of blocks to same level

- Can indicate possible bugs
  - If Source | Format puts lines of code that you wanted to be in the same block at different indent levels, then they weren’t in the same block
The only exception to avoid single-letter variable names is for the normal for loop

```c
for (int i = 0; i < 5; i++)
{
    // ...
}
```
The only exception to avoid single-letter variable names is for the normal for loop:

```c
for (int i = 0; i < 5; i++)
{
    for (int j = 0; j < 10; j++)
    {
        for (int k = 0; k < 20; k++)
        {
            // ...
        }
    }
}
```

Ok Loop Variable Names: i, j, k

- Not l, looks too much like 1

If a better name exists for a loop variable, then use it instead.
One Branch to Execute

```java
if (...) {
} else if (...) {
} else if (...) {
} else {
}
```

Many Branches to Execute

```java
if (...) {
} else if (...) {
} else if (...) {
} else {
} else if (...) {
} else if (...) {
} else if (...) {
} else {
}
```
Lab 3

- To examine the user’s text input
  - Use a large if/else if/... statement
    - if (inputString.equals("MOUTH"))
    - if (inputString.equalsIgnoreCase("MOUTH"))
  - Use a large switch statement
    - switch(inputString)
      { case "MOUTH":
        // ...
      }
  - Could also use inputString.toUpperCase()
Lab 3

- To examine the user’s numerical input
  - String sizeStr = JOptionPane.showInputDialog(...);
    int sizeInt = Integer.parseInt(sizeStr);

- Compute new values for xNose and noseDiameter
  - noseDiameter = noseDiameter * sizeInt
  - Need to change xNose to keep the nose centered
Lab 3: xNose

- Keep nose center in the same location

\[ \text{noseCenter} = xNose + \frac{\text{noseDiameter}}{2} \]

- Solve for new value of xNose: \( xNose' \)

\[
\begin{align*}
xNose' + \frac{\text{noseDiameter}'}{2} &= xNose + \frac{\text{noseDiameter}}{2} \\
xNose' + \frac{n \times \text{noseDiameter}}{2} &= xNose + \frac{\text{noseDiameter}}{2} \\
xNose' &= xNose + \frac{\text{noseDiameter}}{2} - \frac{n \times \text{noseDiameter}}{2} \\
xNose' &= xNose + \frac{\text{noseDiameter} \times (1 - n)}{2} \\
xNose' &= xNose - \frac{\text{noseDiameter} \times (n - 1)}{2}
\end{align*}
\]
Lab 4

- Variable Declaration
  - Basketball scores are integers (int)
  - A count of something is also an integer
  - An average or percentage is a floating-point value (double) (float ok)
Lab 4: Division

- Variable Declaration
  ```
  int gameCount, totalScore;
  ```

- Computing the Average
  ```
  double avgScore = totalScore / gameCount;
  ```
  Incorrect! Performs integer division

  ```
  double avgScore =
      totalScore / (double)gameCount;
  ```
  Correct! Forces floating-point division
Lowest/Highest Scores

- Close Example in Book
  - Section 4.2, Subsection “Controlling the Number of Loop Iterations”

- Initialization Options
  - ```java
  int min = Integer.MAX_VALUE, max = Integer.MIN_VALUE;
  ```
  ```java
  int next = kb.nextInt();
  int min = next;
  int max = next;
  while(next > 0)
  {
    // ...
  }
  ```
Lab 5

- Array Initialization from Command Line Arguments
  - `int[] values = new int[args.length];`
  - `for (int i = 0; i < args.length; i++)`
    - `values[i] = Integer.parseInt(args[i]);`

- Array Display (no extra spaces)
  - `for (int i = 0; i < integerArray.length; i++)`
    - `if (i > 0)`
      - `{ System.out.print(" "); }
    - `System.out.print(integerArray[i]);`
  - `System.out.println();`
Lab 5: Arrays

- Declaration
  - Java Conventional: `int[] integerArray;`
  - Non-Conventional: `int integerArray[];`

- Method Header
  - No need to include size of array as 2\textsuperscript{nd} argument
  - `integerArray.length` provides the number of elements
Lab 5: Bubble Sort

There are many possible solutions. This is just one of them.

**Outer Loop**

```java
public static int bubbleSort(int[] integerArray) {
    // Set number of swaps to zero
    int numSwaps = 0;

    // Assume that the array is not sorted
    boolean isSorted = false;

    // While the array is not sorted
    while (!isSorted) {
        // Guess that the array is sorted
        isSorted = true;

        // For each index in the array
        for (int i = 0; i < integerArray.length - 1; i++) {
            // Inner Loop
            }
        }
    }
    
    // Return the number of swapping operations performed
    return numSwaps;
}
```

**Inner Loop**

```java
// For each index in the array
for (int i = 0; i < integerArray.length - 1; i++) {
    // If elements are out of order
    if (integerArray[i] > integerArray[i + 1]) {
        // Then swap them
        int temp = integerArray[i];
        integerArray[i] = integerArray[i + 1];
        integerArray[i + 1] = temp;

        // Indicate that the array is not sorted
        isSorted = false;

        // Increment the number of swaps by 1
        numSwaps++;
    }
}
```
Questions?

- General Grading Questions?

- If you have specific questions about your assignment, then see me during a lab, after class, or at office hours.
Logistics

Next:
- Information Hiding & Encapsulation

Later:
- In-Class Exercise