Laboratory Assignment № 3

Dr. Duncan, CSC 1350, Louisiana State University 09/24/2019

Using Decision Statements

Learning Objectives

1. Using Nested-If Statements and Keeping Common Code out of Them
2. Using a Counter Variable for the Number of Comparisons
3. Building a Complex String Incrementally, Using Concatenation

In this week's lab, you will write an interactive program that uses decision statements. A decision statement is a control structure that allows a program to condition the execution of a code segment on a logical expression. When using decision statements, avoid redundant test cases. A good programming habit, especially when writing large programs or programs with challenging control structures, is to program incrementally.

Definition 1. The **statistical median** is an order statistic that gives the “middle" value $\tilde{x}$ of a sample. More specifically, it is the value $\tilde{x}$ such that an equal number of samples are less than and greater than the value (for an odd sample size), or the average of the two central values (for an even sample size).

The *MedianOfThreeCalculator* Program

Write a program *MedianOfThreeCalculator* that computes the median of three integers using only the less than (<) operator when comparing them, counts the number of comparisons used to determine the median and prints a strings detailing all the comparisons made along with the logical values, as shown in the sample program interaction

- **Preliminary Version** : Compute and the median of the numbers, using at most three comparisons: “first < second?", “second < third?" and "first < third?".
- **Intermediate Version**: Modify the program so that it also counts and displays the total number of comparisons, *numComparisons*. The program will initialize this counter variable to 1 prior to the first comparison and update it by 1 prior to each other comparison that the program evaluates.
- **Final Version**: Modify the program so that it also displays the relational expressions, comparisons, being evaluated.

Additional Requirements

When generating the *comparisons string* (eg: “11 < 19? (T), 19 < 8? (F); 11 < 8? (F)") for the output part given the input **11 19 8**, follow these steps:

1. “11 < 19?" before the actual comparison “first < second".
2. “(T), 19 < 8?" after the comparison “first < second" succeeded and before the comparison “second < third".
3. “(F), 11 < 8?" after the comparison “second < third" failed and before the comparison “first < third".
4. “(F)" after the comparison “first < third" failed.

Use a String variable *comparisons* for this purpose. It will be initialized as *comparisons* = *first* + *“ < ”* + *second* + *“?”* before the first comparison and then updated using concatenation, as described above.

Compile and run your program six times to ensure that it works correctly for all the following permutations of the inputs: (8, 11, 19), (8, 19, 11), (11, 8, 19), (11, 19, 8), (19, 8, 11) and (19, 11, 9). What are the least and most number of comparisons used in determining the median of three distinct integers? Remove all Netbeans auto-generated comments. Write header comments using the following Javadoc documentation template:

```
/**
 * Explain the purpose of this class; what it does <br>
 * CSC 1350 Lab # 3
 * @author YOUR NAME
 * @since DATE THE CLASS WAS WRITTEN
 */
```

Here is a sample program interaction:

Listing 1: Sample Run: Preliminary Version.

```
Enter three distinct integers to compute their median -> 11 19 8
median(11, 19, 8) = 11.
```

Listing 2: Sample Run: Intermediate Version.

```
Enter three distinct integers to compute their median -> 11 19 8
median(11, 19, 8) = 11.
#Comparisons: 3.
```

Listing 3: Sample Run: Final Version.

```
Enter three distinct integers to compute their median -> 11 19 8
median(11, 19, 8) = 11.
Comparisons: 11 < 19? (T), 19 < 8? (F), 11 < 8? (F); #Comparisons = 3.
```

Listing 4: Sample Run: Final Version (Another Input).

```
Enter three distinct integers to compute their median -> 8 11 19
median(8, 11, 19) = 11.
Comparisons: 8 < 11? (T), 11 < 19? (T); #Comparisons = 2.
```

Submitting Your Work for Grading

Using windows explorer, navigate your way through your netbeansprojects folder and find *MedianOfThreeCalculator.java*, your source code for the program. Right-click the file and create a compressed (zipped) folder containing a copy of the file. Drag the zip file to the desktop and reame the zip file *PAWSID_lab03.zip*, where *PAWSID* is the prefix of your LSU/Tiger email address - the characters left of the @ sign. Double-click the zip file to verify that your program file is in the zip file. If the zip file does not contain your source file, repeat the steps. Upload the zip file to the digital dropbox on Moodle.