Activity 4.2 - Compare and Switch

Overview

In this activity, you will explore various data type comparisons and use switch statements. We will also introduce the VSCode debugger tool.

Instructions

Getting Started

1. Create an "`Activity4.2`" folder in VScode.
2. Import DoubleComparison.java, StringComparison.java, CharacterComparison.java, and Verses.java.
3. Create a text file `Activity4-2.txt` in VS code to keep track of your answers asked in this activity.

Part 1: Data Comparisons

Write your answers to the questions asked below in a document named Activity4-2.txt.

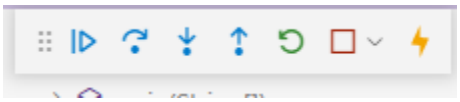
1.1 DoubleComparison.java

Calculates the value of `1.0 - 0.9` and stores the result in a double variable. Uses an `if` statement to compare the calculated result to the expected result (`0.1`). Prints out the result and whether or not they are equal.

1. Change your VSCode settings so you can see more precision in Debug mode:
    a. Open the user settings in VSCode (Gear and Settings)
    b. Search for this key: java.debug.settings.numericPrecision
    c. Set it to 17.
    d. Restart the debugger
2. Set a breakpoint on line 15 by left clicking in the left margin next to the line number OR hover over the left margin at the line, right click, and select "`Add Breakpoint`" OR move your cursor to the appropriate line and push F9 OR (if you are severely masochistic) select "`Run → New Breakpoint → Inline Breakpoint`" - you will

see the breakpoint as a dot marker in the margin.

3. Run the debugger using "`Run → Start Debugging`". The menu below will appear. Hover over each item to see the name of the function it performs.



4. Look at the "Variables" section in the debugger panel. What is the value of the result variable? What is the value of the expected variable? Why aren't they the same?

5. Click "`Run → Step Into`" (F11, or click on the blue arrow pointing down) once to execute the next line of code. Does it go into the if-statement, or the else-statement? Why?

6. Click "Step into" again. This will actually step into the System.out.println code! We don't want to be here, because we didn't write that code. Click "Run → Step Out" (Shift-F11 or blue arrow pointing up) to return back to our code.

7. Click the "Continue" icon  (F5) to finish executing the program.

8. Modify the `if` statement to compare the result using Math.abs and the TOLERANCE as discussed in the textbook and class notes. Run the debugger again to observe how your code executes now.

9. Click the "debugger" icon on the left  to toggle off the debugger view.

1.2 StringComparison.java

Compares the value of a CODE_WORD to another word variable. Then, prompts the user for a code word and uses a `Scanner` to read the input value. Compares the input value to the CODE_WORD.

1. Set a breakpoint on line 17. Run the debugger.
2. Look at the "Variables" panel. What is the value (labeled "value") of CODE_WORD? What is the id (number after @)? What is the value of myWord? What is the id?

3. Use the “Step over” icon to execute the next line of code. Does it go into the if-statement block, or else block? Why?
4. Click the “Step over” icon (F10) to skip over the System.out.println statement and continue to the next line.
5. Set a breakpoint on line 31. Click the “Resume” icon to run the program up until your new breakpoint. When prompted, enter “peanut” as a code word.
6. Look at the “Variables” tab in Eclipse. What is the value of CODE_WORD? What is the id? What is the value of input? What is the id?
7. Use the “Step over” icon to execute the next line of code. Does it go into the if-statement block, or else block? Why?
8. Click the “Continue” icon to finish executing the program.
9. Fix the conditions in the if-statements to correctly compare the String values using the equals method. Run the debugger again to observe how your code executes now.
10. Click the “debugger” icon on the left  to toggle off the debugger view.

1.3 CharacterComparison.java

Stores the first letter of a word in a `char` variable and prints its ascii decimal value. Uses an `if` statement to determine if the character is an upper-case letter, lower-case letter, or a number.

1. Set a breakpoint on line 15. Run the debugger.
2. Look at the “Variables” panel. What is the value of firstLetter? What is the value of asciiValue?
3. Use the “Step over” icon to trace through the steps of the program. Change the word to different values and see how the program execution changes.
4. Click the “debugger” icon on the left  to toggle off the debugger view.

Part 2: 12 Days of Christmas (`switch` statements)

PP 4.14

Modify Verses.java to print the verses of the song “The Twelve Days of Christmas”, in which each verse adds one line. For example, the first two verses of the song are

On the 1st day of Christmas my true love gave to me

A partridge in a pear tree.

On the 2nd day of Christmas my true love gave to me
Two turtle doves, and
A partridge in a pear tree.

1. Use a `switch` statement to put the appropriate suffix on the day number (e.g. 1st, 2nd, 3rd, 4th, 5th, etc.).
2. Use a separate `switch` statement to control which lines get printed for the given day.

* You can choose a different holiday and/or your own gifts if you would like, but the song pattern must remain the same.

Terminology Identification

In your reflection give a brief example of each of the following: when it would be appropriate to use equals(), compareTo(), and ==; when it would be appropriate to define a tolerance; and when it would be appropriate to use a switch statement as opposed to if-else statements.

Code Review

When you are finished with this activity, pair up with a classmate and review each other's code to make sure it meets all the requirements.

Submission

After completing the assignment, use the assignment link in Canvas and follow the submission instructions there. You will upload your `Activity4-2.txt` and `Verses.java` files and submit your reflection in the "Comments" box.

Reflection Requirements

Write a one paragraph reflection describing your experience with this activity. The reflection should also include the name of your code review partner AND something interesting you found in their code. Please review the activity rubric for more details.