CSCI 1301: Introduction to Computing and Programming

Lab 05 – Testing and Debugging Java Code

Introduction

Errors can be categorized into:

- Syntax errors
- Logical errors
- Runtime errors

A syntax error occurs when your program does not comply with the syntactical rules of the Java language. Usually, these errors are caused by misspelled words and missing punctuation, among others. Syntax errors are relatively easy to find and correct as the compiler points out the line where the syntax error has occurred. Runtime errors occur during the execution of a program when the program attempts to perform an invalid operation, like dividing by zero, causing the immediate termination of program's execution.

Logical errors are errors in your program due to the design of a faulty algorithm to solve the problem at hand or in the implementation of such an algorithm. When your application has a logical error, it might compile and run without the computer generating any error messages, but it will not work as expected. Logical errors are generally more difficult to locate and fix than syntax errors. The process of finding a logical or runtime error is called ***debugging***. An easy way to find a logical error is by tracing the value of the variables in a program while running using **`System.out.println(...);`** statements or by tracing the execution of the program step by step using a debugger.

In this lab you will learn how to monitor the value of the variables in a Java program and how to use the Eclipse debugger to locate its logical errors.

What to Submit

You should submit an error free copy of the program **`TemperatureConverter.java`**.

Instructions

Download the file **`TemperatureConverter.java`**, which contains several bugs, and the text document **`DebuggingWorksheet.txt`**.

Note: The worksheet has been provided for your convenience. Completing it might help you debug the program, but you do not need to submit it with your completed source file.

For the lab, you will test and debug the program to make it a correct, error free program. **Please resist the urge to try and look at the program to fix the errors. Even if you can do that, it is not going to help you moving forward in this class. Debugging skills will be critical as we move to more complex programming constructs.**

The program should allow the user to enter a temperature in Fahrenheit and display the equivalent temperature in a temperature scale of the user's choice: Kelvin, Rankine, Reaumur, or Celsius. The program *should* do this. However, because of errors, it does not function properly in its current form.

The conversion from Fahrenheit to the other temperature scales is performed according to the following table:

From Fahrenheit to	Conversion Formula
Kelvin Degrees	[°K] = ([°F] + 459.67) × 5/9
Rankine Degrees	[°R] = [°F] + 459.67
Reaumur Degrees	[°Re] = ([°F] – 32.0) × 4/9
Celsius Degrees	[°C] = ([°F] – 32.0) × 5/9

The program should do the following:

I. Prompt the user for a valid temperature in degrees Fahrenheit.
II. If the user enters a valid temperature in degrees Fahrenheit (i.e. **temperature greater than or equal to -459.67**), then the program asks the user the temperature scale to perform the conversion and displays such a conversion.

The program **`TemperatureConverter.java`** has several logical errors that you need to identify and correct using **`System.out.println`** and the **Eclipse** debugger.

1. After you have downloaded the program **`TemperatureConverter.java`** to your computer, create a new **Java Project** called **`TemperatureConverter`** in Eclipse, then create a new class called **`TemperatureConverter`** in the **TemperatureConverter Java Project**.

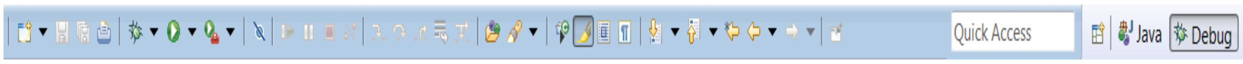
Using **Notepad**, open the file **`TemperatureConverter.java`** you have just downloaded on the desktop and copy all of its content into the class **`TemperatureConverter`** in your **Eclipse** project.

2. Add **`System.out.println`** statements as needed to help you understand and debug the program. After debugging, you should remove or comment out these **`println`** statements so they don't show up in the correct version of your program.
3. Compile and run the program.
4. Execute the program entering the value **56** at the prompt and enter 3 when prompted for the temperature scale.

   - **[Worksheet Item 1] What do you observe when you run the program?**

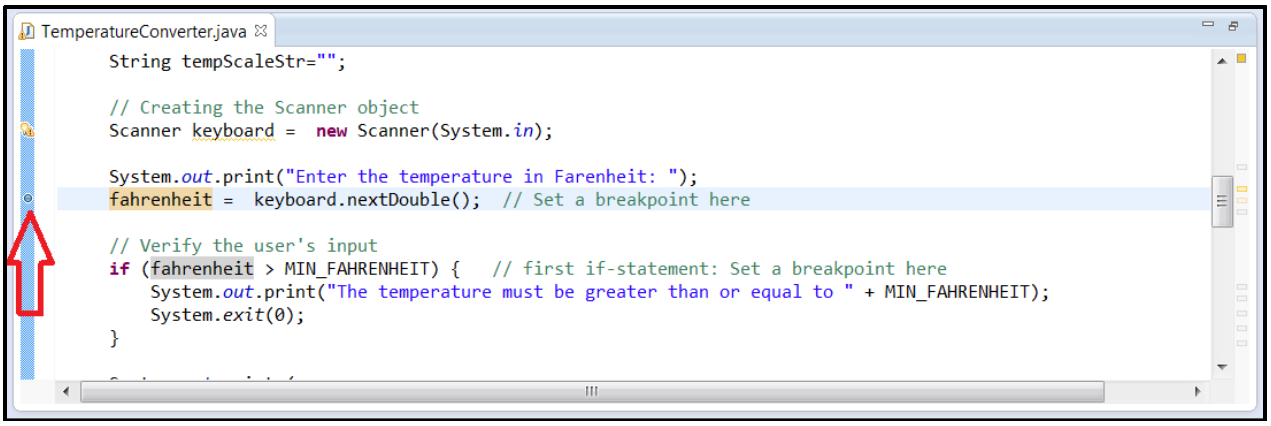
5. One easy way to debug your program is to print to the console the values of the variables and other messages that will help you trace the execution of the program in order to find out the logical errors. However, this technique can be very time consuming if you are dealing with larger programs. Instead, a more convenient way to locate logical errors in a program is by tracing the execution of the program and monitoring the values of the variables using a debugger. For the rest of this lab exercise, you will learn the basics of the **Eclipse** debugger that will help you debug and fix logical errors in a Java program.

6. Click on the **Open Perspective** , and then the **Debug Perspective** Debug in **Eclipse**:



Afterwards, **Eclipse** will open several windows to help you to trace the execution of this project.

7. Scroll through the Java source code and set break points where it is indicated by a comment (look for "**`Set a breakpoint here`**"). To set a breakpoint, click on the left margin (in blue-grey) on the line where you want to place the breakpoint. You will see a small dot in the margin to the left of the line.

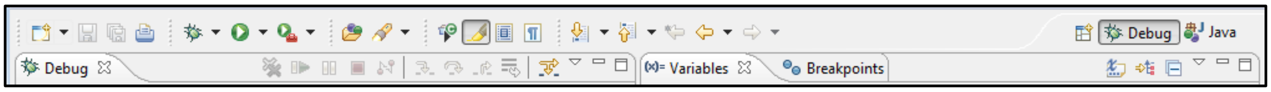


```
        String tempScaleStr="";

        // Creating the Scanner object
        Scanner keyboard =  new Scanner(System.in);

        System.out.print("Enter the temperature in Farenheit: ");
        fahrenheit =  keyboard.nextDouble();  // Set a breakpoint here

        // Verify the user's input
        if (fahrenheit > MIN_FAHRENHEIT) {   // first if-statement: Set a breakpoint here
            System.out.print("The temperature must be greater than or equal to " + MIN_FAHRENHEIT);
            System.exit(0);
        }
```

8. After you have set the indicated breakpoints in the program, click on the tab **Breakpoints**.

   ○ **[Worksheet Item 2] Where (what line numbers) have you set the breakpoints?**

9. Run the program in **Debug** mode by clicking the little bug in the toolbar , or **Run/Debug** in the menu bar. Click **OK** on the window **<u>Save and Launch</u>**.



10. Click on the **Variables** window tab, (x)= Variables , it should be located in the top right window.

   ○ **[Worksheet Item 3] What is the value of the named constant `MIN_FAHRENHEIT` and the variable `tempScaleStr` displayed in the Variables window?**

11. The debugger will stop on the first breakpoint you set. Observe that the debugger highlights the line of code that will execute next.

Now, click on the **Step Over** icon, , in the toolbar to execute the statement

```
fahrenheit = keyboard.nextDouble();
```

Click on the **Console** window, then enter **56** and hit enter.
Afterwards, do the following:

a) Click on the **Step Over** icon, , several times to see what happens and answer the following questions in the Debugging worksheet:

- **[Worksheet Item 4a] Does the first `if` statement display the error message correctly?**
- **[Worksheet Item 4b] What is the problem?**
- **[Worksheet Item 4c] How would you fix the program so it terminates when the user enters a Fahrenheit temperature less than the possible minimum temperature?**

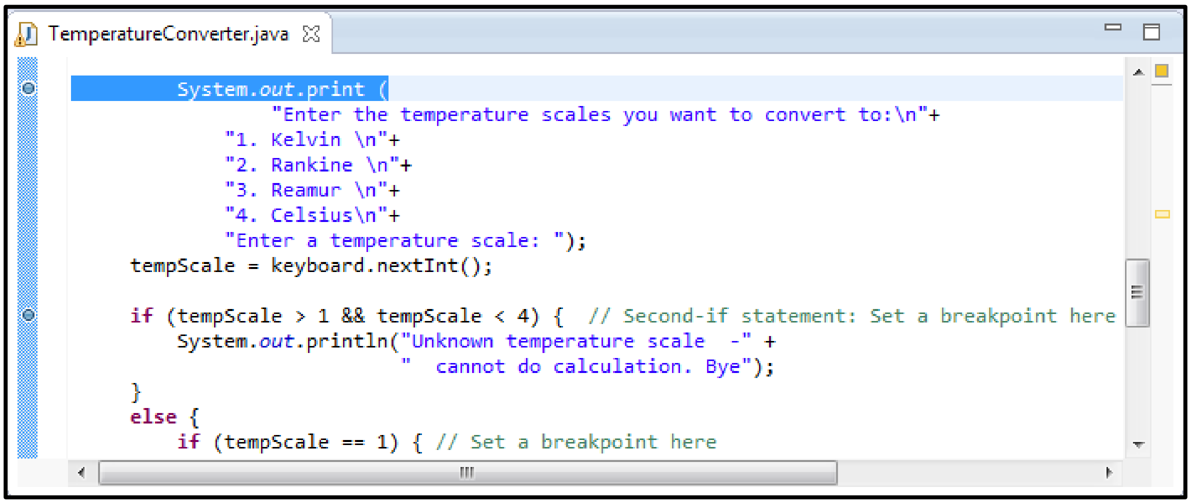
b) Terminate the current execution of the program by clicking on the

Terminate icon, , in the toolbar or to the right of the **Console** Window.

12. Modify the first if statement so it works correctly.

13. Run the program in **Debug** mode again by clicking the little bug in the toolbar, or **Run/Debug** in the menu bar. When you reach the **first and second breakpoints**, use

Step Over icon, , to be sure that the first if statement works properly. If you have modified this if statement correctly you will notice that the next statement to be executed is the prompt that asks the user to enter a temperature scale:



```
    System.out.print (
            "Enter the temperature scales you want to convert to:\n"+
        "1. Kelvin \n"+
        "2. Rankine \n"+
        "3. Reamur \n"+
        "4. Celsius\n"+
        "Enter a temperature scale: ");
tempScale = keyboard.nextInt();

if (tempScale > 1 && tempScale < 4) {  // Second-if statement: Set a breakpoint here
    System.out.println("Unknown temperature scale  -" +
                       "  cannot do calculation. Bye");
}
else {
    if (tempScale == 1) { // Set a breakpoint here
```

14. Click on the **Resume** button, , in the toolbar, enter **3** when asked about the temperature scale (and hit **Enter** afterwards).

15. Now, click on **Step Over,** , in the toolbar.

   - **[Worksheet Item 5a] Is the condition of the second `if` statement correct?**
   - **[Worksheet Item 5b] How will you change it?**

16. Terminate the current execution of the program (click on the **Terminate** icon, , in the toolbar or to the right of the **Console** window). Then, do the following:

   c) Remove the first breakpoint you set by clicking on the dot in the margin on the left.

   d) Remove the breakpoint you set on the first if statement.

   e) Modify the condition of the second if statement properly.

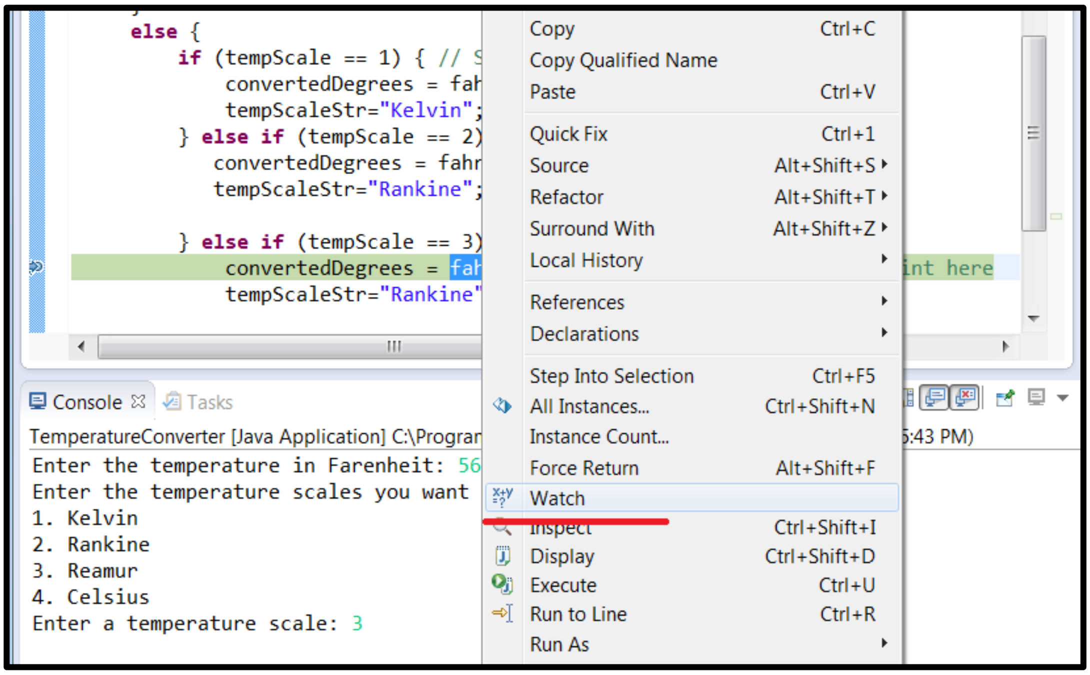
   f) Run the program again in Debug mode (click on the little bug in the toolbar), or click **Run/Debug** in the menu bar. Then click on **Step Over,** , or the **Resume** button, , in the toolbar to trace the execution of your program until it reaches the statement:

```
convertedDegrees = fahrenheit - 32* 4/9 ;
```

17. We are going to monitor the value of the following two expressions:

```
1) fahrenheit - 32
2) fahrenheit - 32* 4/9
```

To do so, select the expression, then right click and click on **Watch.** A window called **Expressions** will open up.



```
else {
    if (tempScale == 1) { // S
        convertedDegrees = fah
        tempScaleStr="Kelvin";
    } else if (tempScale == 2)
       convertedDegrees = fahr
       tempScaleStr="Rankine";

    } else if (tempScale == 3)
        convertedDegrees = fah                                    int here
        tempScaleStr="Rankine"
```

```
TemperatureConverter [Java Application] C:\Progran                5:43 PM)
Enter the temperature in Farenheit: 56
Enter the temperature scales you want
1. Kelvin
2. Rankine
3. Reamur
4. Celsius
Enter a temperature scale: 3
```

- **[Worksheet Item 6] What is the value of the expression just selected in the Watch window?**

18. Click on the **Resume** button, ▶, in the toolbar.

- **[Worksheet Item 7a] Does the program compute the correct temperature in Reaumur degrees? (Check if multiplying the 1st expression by 4/9 gives you the value of the 2nd expression.**
- **[Worksheet Item 7b] Does the program assign the correct value for`tempScaleStr`?**
- **[Worksheet Item 7c] How can you modify the program to compute the value properly?**

19. Fix the body of the if statement that computes the Reaumur temperature properly.

20. There are other logical errors, use the debugger to fix them. Then, compile, run, and test the program until you are sure the program is error free and works correctly to convert any valid temperature in Fahrenheit to an equivalent temperature when the scale chosen by the user is valid.

Additional Requirements

These are things that make the graders lives easier, and ultimately, you will see in the real world as well. Remember the teaching staff does not want to touch your code after they gave you requirements; they want to see the perfect results they asked for! Here is a checklist of things you can **lose points** for:

- (10 points) If the source file(s)/class(es) are named incorrectly (case matters!)
- (10 points) If your source file(s) have a package declaration at the top
- (10 points) If any source file you submit is missing your Statement of Academic Honesty at the top of the source file. All submitted source code files must contain your Statement of Academic Honesty at the top of each file.
- (10 points) If you have more than one instance of Scanner in your program. Your program should only have one instance of Scanner.
- (15 points) Inconsistent I/O (input/output) that does not match our instructions or examples exactly (unless otherwise stated in an assignment's instructions). Your program's I/O (order, wording, formatting, etc.) must match our examples and instructions.
- (100 points) If the source file(s) are not submitted before the specified deadline's late period ends (48 hours after the deadline) or if they do not compile.
- (25 points) Late penalties will be deducted as per the course syllabus.
- If your (10 points) comments or (10 points) variables are "lacking"
  - Here, "lacking" means that you or a TA can find **any** lines of code or variables that take more than 10 seconds to understand, and there is no comment, or the variable name does not make sense (variable names like **b**, **bb**, **bbb**, etc. **will almost never be acceptable)**
- (10 points) Indentation is not consistent throughout your source code
  - Refresh your memory of indentation patterns in chapter 2 in the course textbook
  - Be careful of a combination of tabs and spaces in your files (use one or the other)!

If any of the above do not make sense to you, talk to a TA, or ask on Piazza!

eLC Submission and Grading

After you have completed and thoroughly tested **`TemperatureConverter.java`**, upload and submit it to ***eLC.*** You do not need to submit the debugging worksheet. Always double check that your submission was successful on ***eLC***!

The lab will be graded according to the following guidelines.

- A score between 0 and 100 will be assigned.
- All instructions and requirements must be followed; otherwise, points maybe deducted.
- The program will be evaluated against several test cases to determine whether each error has been corrected. **Case(s) taken from the examples below as well as other test cases will be used**. **In other words, you should test all temperature scales to make sure they are working.**

Examples - *Remember, these are not exhaustive- try your own!*

```
Enter the temperature in Fahrenheit: 56
Enter the temperature scales you want to convert to:
1. Kelvin
2. Rankine
3. Reaumur
4. Celsius
Enter a temperature scale: 1
56.0 degrees Fahrenheit is 286.48333333333335 degrees Kelvin.
```

***

```
Enter the temperature in Fahrenheit: 56
Enter the temperature scales you want to convert to:
1. Kelvin
2. Rankine
3. Reaumur
4. Celsius
Enter a temperature scale: 2
56.0 degrees Fahrenheit is 515.6700000000001 degrees Rankine.
```

***

```
Enter the temperature in Fahrenheit: 113
Enter the temperature scales you want to convert to:
1. Kelvin
2. Rankine
3. Reaumur
4. Celsius
Enter a temperature scale: 3
113.0 degrees Fahrenheit is 36.0 degrees Reaumur.
```

***

```
Enter the temperature in Fahrenheit: 56
Enter the temperature scales you want to convert to:
1. Kelvin
```

```
2. Rankine
3. Reaumur
4. Celsius
Enter a temperature scale: 4
56.0 degrees Fahrenheit is 13.333333333333334 degrees Celsius.
```