ALGONQUIN COLLEGE

OBJECT ORIENTED PROGRAMMING (JAVA) (CST8284)

Course Overview & Introduction

LAB 2

Arrays, and Use of the Debugger

<u>Final due date:</u> Week of 3 (your lab day that week)

LAB 2: Preparations

The essence of this lab, is to learn to work with 2-dimensional arrays, as well as to demonstrate the use of the Eclipse debugger.

Preparations:

- Review the information on **Debugging** provided to you in Week 1 **Hybrid** tasks on Brightspace course page

- Review **one and two dimensional arrays** using the course textbook and/or other resources such as:

http://docs.oracle.com/javase/tutorial/java/nutsandbolts/arrays.html

PART 1

REVIEW AND UPDATE YOUR JAVA CODE

The Outcomes to Demonstrate

- Show your professor your **updated version** of the code
- Run and show the **Javadoc** for the comments in this code
- Demonstrate to your professor (**using** the code you **updated**) <u>why</u> and <u>how</u> to correctly use the Eclipse debugger focusing on:
    - Breakpoints
    - Single stepping (*step into* and *step over*)
    - Inspecting variables

Part 1: Description

- In this part, you will work with 2D arrays. You will **update** the Java code provided to you so that it performs additional tasks. **This java program:**

- **Shows** the number of people who have recovered from COVID-19:
  - Across **7 provinces** in Canada
  - Over a period of **8 months** (Feb. to Sept., 2020).

- **Sums** up the numbers of the recovered persons across Canada for each month.

- **Prints** a formatted output using `printf`

Pause…

- Before you continue, have you reviewed the hybrid materials on how to use a debugger?
- Have you reviewed the hybrid material on 2D-arrays?
- If not, **pause now** and review these materials **before** you **continue**!
- You may **not** be able to successfully complete this lab without these reviews.

Part 1 - You are Required to:

- **Create** a new project in eclipse – **Lab 2**
- **Load** and review the code file provided into your Eclipse editor (**CovidStatistics.java**).
- DO **NOT** load the code file that has the sample output portion (**CovidSample**). It's just for your review.
- **Open** and **review** the **sample** code file (CovidSample) on Notepad to understand the entire program and expected **output**.

Part 1 - You are Required to: (2)

- **Update** the commented section in the code provided to you using a **nested** for loop to print the sum of the elements in the array.
  - **Compute** the **sum** of the recovered persons in Canada for each **month** specified.
  - **Use** `printf` to format and **print** the column **sum**.

The program output is shown in **CovidSample** file for your guidance **only**. See comments inserted.

PART 2

DEMONSTRATING THE USE OF A DEBUGGER

You are Required to:

- Show **how** to run your code in different perspectives (e.g. <u>debug</u> and <u>Java</u> mode)
- **Identify** different <u>view panels</u> in debug mode
- **Select** a **breakpoint** in the code you have updated to demonstrate your work
- **Explore** and **analyze** the **variables** in your code

Important…

Some important concepts you need to know in order to make effective use of a debugger:

- Breakpoints
- Single Stepping
- Inspecting Variables

- Review the **Hybrid resources** and course materials to learn how the Eclipse debugger works to complete this part.

Starting the Debug Mode

- On the top bar of your Eclipse screen:
  - Go to **Window** and then **Open Perspective** to select the **debug**
  - Or you can: Go to **Run** and then select **Debug**
- If it is your first time running **Debug**, agree to the debug mode pop up window seen on your screen in the debug perspective.
-

Starting the Debug Mode (2)

- Your screen changes. Explore the additional view panels included.

- Notice the **Debug** and **Java** icons at the far <u>top right </u>corner of your screen, which you can use to go back and forth on Java or Debug **modes**

- Example of the new view panels that appear are the **Variables**, **Breakpoint** and **Expressions** (at the top right corner of your debug mode screen). Explore them.

Insert a Breakpoint in your code

- Ensure that the eclipse editor shows line numbers for your code
- Insert a **breakpoint** in your **updated** code:
  - Double click on the blue line margin corresponding to the desired code line number
  - **OR you can**: do a **right click** on the margin of the desired line number and then select **Toggle Breakpoint.**
- Explore how the **toggle breakpoint and disable breakpoint** work

Explore the following Step Commands using your code

- **Step into** - You can use this button to step into a method you wish to debug one step after another
- **Step over** – You can use this button to skip a method you do not wish to debug when invoked
- **Step Return -** You can use this button at the end of your debugging and having the debugger back to an initial point of start.

Explore the following Step Commands (2)

- **Terminate** – You can use this button to stop the running of a program if there are no further analysis required, or if errors are encountered. The terminated program could e in the debug or normal mode.
- **Resume** – You can use this button to restart execution of the program again from any suspended state. This continues till another breakpoint.

Demonstrating Your Work

- **To obtain your mark, show your Professor**:
  - The code you updated
  - Run the program and show that the output of your code is correct.
  - Run Javadoc to document the comments in your code and show theJavadoc output

Demonstrating Your Work (2)

- **Show your Professor how you**:
  - Select a **breakpoint** in your updated code
  - Run your code in debug mode (as a Java application):
    - Go to **Run** and select **Debug As** then select **Java Application**
    - Click **Yes** in the pop up window to agree to the debug perspective (if it pops up)
    - The breakpoint will be **highlighted** in your code.

Exploring the Step into and Step over

- Click on the **step into** icon at the top bar of your screen.
  - What did you observe?

- Click on the **step over** icon at the top bar of your screen.
  - What did you observe?

Demonstrating your Work (3)

Explore the variables

- The Variables, Breakpoint and Expression view panels are at the right-hand top corner of your screen.
- **Focus** on the Variables panel and carefully inspect the two columns (Name and Value).
- Observe the names of the **variables,** the **values** stored and **how**.

Demonstrating your Work (4)

- What **types** (of values) do you see?
- Click on **>** beside **recovered in the Name**
- Click on **>** beside **[1]** in the Name column
- Observe changed in the **Value** Column?
- What does this change mean?

❖ Discuss your observations with your professor while you obtain your marks.

Test your Knowledge further…

- When do you need to use a 2D array?
- Why should you consider switching to the **debug mode** from the **run** mode?
- Why should you ***step into*** a method?
- When should you ***step over*** a method?
- Why and where do you examine the code **variables** using the debugger? What do you expect to see?

References – Course textbooks

- Java How to Program, Early Objects Plus MyProgrammingLab with Pearson eText -- Access Card Package, 11/E. Author: Deitel ISBN: 9780134800271
- Big Java Early Objects, 7/E. Author: Horstmann, C. Wiley. ISBN: eText: 978-1-119-49909-1 or loose-leaf paper: 978-1-119-74020-9.

Finalizing…

- Remember to **demonstrate your work** to your lab professor to receive your marks

- **Marks** are all or **nothing**. Show your work fully

- Remember to **review your hybrid tasks** as specified for each week

- Remember to **keep ahead** by checking if there are any more assessments due this week