CS112 – Spring 2024

Lab04

Instructor: Paul Haskell

INTRODUCTION

This week you will install the Eclipse Integrated Development Environment (IDE) and you will use it to complete two simple programs and to push them to GitHub. Here's the challenge: the programs are due by the end of class!

Installing Eclipse

Point a browser to https://www.eclipse.org/downloads/ and download the "Eclipse IDE 2023-06". Mac users, be sure you download the proper version for your CPU type (x86 or ARM). Save the downloaded file somewhere you can find it. Run the installer, and if asked, install the "Eclipse IDE for Java Developers". Pick a folder where Eclipse should be installed on your computer. On Windows, I have **not** had good look installing it in the usual "Program Files" directories. I put it into a personal directory or Temp directory.

To use Eclipse to push your Java files to GitHub, and to build and run your Java files either in Eclipse or using **javac** and **java**, you need to set up Eclipse in a particular way. Eclipse will ask you to set up a Workspace where it saves its projects. Please pick your **CS112/MyWork** directory. (NOT the CS112/MyWork/eclipse-workspace directory that Eclipse may offer to you!)

The Eclipse Welcome Window will ask if you want an overview, tutorial, set preferences, etc. Feel free to do those on your own as you want to learn more about Eclipse. For now, you can just click the "Hide" button in the upper right.

Using Eclipse

When you start Eclipse, you can create a new Java Project by using the File->New menu to select the "Java Project" option. Do not create a module—select the "Don't Create" option if asked, and clear the "module metadata" checkbox near the bottom of the "Create Project" window. For this week, please create a project called **Lab04**.

When you make the **Lab04** project, Eclipse will make a **Lab04** directory inside your **MyWork** directory. It also makes a "**src**" directory (for "source code") inside the **La04** directory, and also a "**bin**" for ("binary code" i.e. *.class files) directory inside your **Lab04** directory.

Ok, now let's make a Java file! Find your new project in the list of projects on the left side of the Eclipse window. Open the little pulldown menu to see the "src" directory. Right-click that, go to the "New" menu, and select "Class". This pops up a window: enter a Name for your new Java source file (use the "default package"), and click OK. Now the source code window should open; this lets you edit and save the source code right in the window.

Go ahead and write a basic "HelloWorld" program. You can compile and run it in one step by clicking the green circle with the white triangle inside. Or you can go to the "Run" menu and select the "Run" option.

You might have noticed that Eclipse automatically highlights Java errors—and sometimes even suggests fixes—in the source code window. Very convenient.

GitHub and Eclipse

One of the useful features of Eclipse is that it makes working with GitHub really easy—much easier than with the command line. But before we talk about GitHub, let's talk about "Eclipse Views". A "View" is just an arrangement of subwindows inside Eclipse that are useful for doing a particular activity. Right now, you're probably using Eclipse's "Java View". Do you see a little icon that looks like this in the upper right-hand corner of the Eclipse window?

If you click that, you get the "Java View". You can customize this view how you want, but by default, it should include a "Package Explorer" that shows your various projects, a Text Window for editing Java files, and probably a Console window to show the output of your programs when they run.

Next you will enable the "Git View". On the top-level "Window" menu, select "Show View" and then "Other…". Select the "Git" entry and push the "Open" button. You now should have a new "View button" at the top-right of your Eclipse window, showing the Git View.

When you select the Git View, again you can customize things how you like them. However, the really useful controls are in the "Git Staging" tab. Please select that.

- The "Unstaged Changes" pane shows any Java files that you have modified but not yet "staged" i.e. added. You can drag files to the "Staged Changes" area—that's the same as typing "git add" on the command line.
- The "Commit Message" window lets you type in a commit message, just as you would do with the "git commit -m ThisIsMyComment" command
- The "Commit and Push" button does just what it says: commits your changed files and pushes them to GitHub. Easy!
- The "History" tab shows a history of all the commits you made, along with your comments. You can click on a particular commit to get more details about it.
- One more neat trick: if you double-click on a filename in the "Unstaged Changes" or "Staged Changes" pane, Eclipse will open up a window that shows the difference between the previous checked-in version of the file and your current version. This makes it easy for you to see if you are happy with your changes.

Once I started using Git from Eclipse, I never used the command line again. It's so easy that I make Git commits multiple times per hour. And that's good—it means my work won't get lost through some accident or computer crash.

Cubes

Ok, now for the real assignment! Using Eclipse, you will write a program called **Cubes.java** in the **Lab04** directory that will print the cubes of the integers starting with 1 and increasing. The program should

- only print the values of the cubes (not the original numbers)
- should not print any output values greater than or equal to 2000. (For example, the cube of 20 is 8000, which is too big!)

Your output should look like:

1
8
27
etc

Collatz Conjecture[1]

On to some serious math. There is a mathematical conjecture (something that people think is true but cannot prove) that says "Pick any positive integer. If the number is even, replace it with x/2. If it is odd, replace it with 3x + 1. Keep repeating that process until your number equals 1." The conjecture says all positive integers eventually converge to 1.

In Eclipse, in the **Lab04** project, create a new program called **Collatz.java**. Now use a text editor to open the file **CollatzStart.java** in **CourseInfo/Lab04**, and copy that text into your own program, in Eclipse.

You must finish the Java code for **Collatz.java** . Your program should run the recipe above for the integers from 1 through 200, keeping track of the number of "steps" required (either dividing by 2 or replacing with 3x + 1) until the result equals 1. Print out a table that lists each number and its required number of steps.

- 1 needs 0 steps
- 2 needs 1 step

So your output should look like:

1 0
2 1
3 7
etc

[1] Taken from *Fundamentals of Java Programming,* Mitsunori Ogihara, Springer Press.

Reminder

Put useful comments into your code. Design and organize your code so that it is easy for others to understand.

Use Eclipse to push **Cubes.java** and **Collatz.java** to GitHub before the end of class.

Conclusion

In this lab you used Eclipse to create, edit, run, test two programs and to upload them to GitHub.

Rubric

Cubes.java is worth 5 points:

- One point if it has the correct name, is located in the correct directory, and compiles
- Four points if the output is correct

Collatz.java is worth 10 points:

- Two points for each of 5 cases that we will verify from your program output