Programming Exercise 0 – Hello World

Authors: Vince, Jack, Andrew

Purpose

Welcome to your first CS 1331 programming assignment! In this assignment, you will verify that you have properly installed Java on your machine and that you are able to run a Java program.

Solution Description

Installing Java

In this class, we will be using Java 11 (openjdk). Make sure you have Java 11 installed and are using Java 11 in this course, even if you already have a previous version of Java. We have detailed instructions for this on Canvas (Modules → General Information) in a document titled "Guide to Installing Java.pdf".

Open the command line and run the `javac -version` command. You should see output similar to the following:

```
javac 11.0.20
```

Take and save a screenshot of the command line showing the result of `javac -version`

Next, run the `java -version` command. You should see similar output to the following:

```
openjdk version "11.0.20" 2023-07-18
OpenJDK Runtime Environment Temurin-11.0.20+8 (build 11.0.20+8)
OpenJDK 64-Bit Server VM Temurin-11.0.20+8 (build 11.0.20+8, mixed mode)
```

Take and save a screenshot of the command line showing the result of `java -version`

NOTE: It is okay if the minor version reported is different than what is written above. As long as the number reported is "11.0.X", you should be fine!

Writing and Compiling a Java Class

In order to create a Java class, you must first create a file with the .java extension. In our example here, we will use `Test.java`. Within the class, you will need to write a class header. For now, just memorize that you need to write `public class` before the class name. We will go more in-depth into what exactly this means later. As for the class name, it must exactly match (case-sensitive) the file name.

Therefore, our `Test.java` file should have the following (note the capitalization and curly braces):

```
public class Test {

}
```

Now that you have written a Java class, you should be able to compile your code! On the command line, navigate to the directory where you have saved this file.

After navigating to the proper directory, you will be able to compile the Java class you have just created! We do this by using the `javac FileName.java` command. Since we named our example file `Test.java`, we will compile it using `javac Test.java`.

This should create a new file in your directory called `Test.class`! Created by the Java compiler, this file contains Java bytecode, which will not make much sense to you if you open it up. However, it makes a lot of sense to the Java Virtual Machine, also known as the JVM (a computing machine that allows our program to run).

If a new file was not created, your program did not compile. Read the compilation error produced on the command line and fix it before proceeding!

The Main Method

The method header should look like this:

```
public static void main(String[] args) {
}
```

This method is critical for a Java class to be "runnable." Now, you can write the contents of your first Java program within your main method. For this assignment, simply do the following:

- Print out "`Hello World`"

Now, compile your code and correct any error messages that may appear. In the future, as programs get more complicated, it is good practice to compile often as you make incremental changes. This will help you correct any errors you may make as you go instead of dealing with them all at the end.

To run your program, use the command `java FileName`. Since we named our file `Test.java`, we will use the command `java Test`. Note that we didn't include the `.class` extension after `Test`; it will not run if you write `java Test.class`. If you ever want to run an updated version of your program after making alterations, know that you must recompile before doing so.

Good luck, and happy hacking!

Turn-In Procedure

Submission

To submit, upload the files listed below to the corresponding assignment on Gradescope:

- `Test.java`
- Screenshot of the command line showing the result of `javac -version`
- Screenshot of the command line showing the result of `java -version`

Make sure you see the message stating the assignment was submitted successfully. From this point, Gradescope will run a basic autograder on your submission as discussed in the next section. **Any autograder tests are provided as a courtesy to help "sanity check" your work and you may not see all the test cases used to grade your work.** You are responsible for thoroughly testing your submission on your own to ensure you have fulfilled the requirements of this assignment. If you have questions about the requirements given, reach out to a TA or Professor via the class forum for clarification.

You can submit as many times as you want before the deadline, so feel free to resubmit as you make substantial progress on the assignment. We will only grade your latest submission. **Be sure to submit every file each time you resubmit**.

Gradescope Autograder

If an autograder is enabled for this assignment, you may be able to see the results of a few basic test cases on your code. Typically, tests will correspond to a rubric item, and the score returned represents the performance of your code on those rubric items only. If you fail a test, you can look at the output to determine what went wrong and resubmit once you have fixed the issue.

The Gradescope tests serve two main purposes:

- Prevent upload mistakes (e.g., non-compiling code)
- Provide basic formatting and usage validation

In other words, the test cases on Gradescope are by no means comprehensive. Be sure to thoroughly test your code by considering edge cases and writing your own test files. You also should avoid using Gradescope to compile, run, or Checkstyle your code; you can do that locally on your machine.

Other portions of your assignment can also be graded by a TA once the submission deadline has passed, so the output on Gradescope may not necessarily reflect your grade for the assignment.

Allowed Imports

To prevent trivialization of the assignment, you may not import any classes for this assignment.

Feature Restrictions

There are a few features and methods in Java that overly simplify the concepts we are trying to teach or break our autograder. For that reason, do not use any of the following in your final submission:

- `var` (the reserved keyword)
- `System.exit`
- `System.arraycopy`

Collaboration

No collaboration is allowed on this assignment. Please see the syllabus for more details.

In addition, it is not allowed to upload your code to any sort of public repository. This could be considered an Honor Code violation, even if it is after the assignment is due.

You **MAY NOT** use code generation tools to complete this assignment. This includes generative AI tools like ChatGPT.

Important Notes (Don't Skip)

- Non-compiling files will receive a 0 for all associated rubric items
- Do not submit .class files
- Test your code in addition to the basic checks on Gradescope
- Submit every file each time you resubmit
- Read the "Allowed Imports" and "Restricted Features" to avoid losing points
- **Check on Ed Discussion for a note containing all official clarifications and sample outputs**

It is expected that everyone will follow the Student-Faculty Expectations document, and the Student Code of Conduct. The professor expects a **positive, respectful, and engaged academic environment** inside the classroom, outside the classroom, in all electronic communications, on all file submissions, and on any document submitted throughout the duration of the course. No inappropriate language is to be used, and any assignment, deemed by the professor, to contain inappropriate, offensive language or threats will get a zero. You are to use professionalism in your work. Violations of this conduct policy will be turned over to the Office of Student Integrity for misconduct.