Module 1 Study Questions

Re-submit Assignment

Due No Due Date **Points** 0 **Submitting** a file upload

Submit brief answers to the following questions. This assignment will not be graded, though I will review your answers to check your progress. Complete this assignment and save it in a .txt format. Please be sure to include your last name and the course number in the file name, like so: [your name], X436.2, Module 1 Study. Upload your .txt file by clicking submit on the right side of your screen.

1. What kinds of diagnostics does your Java compiler produce when you feed it erroneous input? To find out, start with the SumOneToTen program and make small changes until you see errors. How helpful are the compiler messages? Are there any you think could be improved?
2. What happens if you assign a real-number value (such as 2.3) to an integer variable in Java? Try it and see. What happens in C or in Visual Basic, or any other language you know? Why is this a good thing? Try it with a quoted string of characters: int x = "What happens?"
3. How large is the compiled version of SumOneToTen (i.e., the .class file)? How large would an executable compiled from your favorite language be? Why is Java different (or not)?
4. What features make a program hard to understand? easy to understand? hard to modify? What was the weirdest bug you ever found in a program? What feature, added to the language the program was written in, might have prevented the bug from ever happening?
5. It is obvious that what a programming language is capable of expressing limits what kind of programs one can write with it: Java has no "address of" operator, so it would be very hard to use it to write UNIX device drivers! More often, though, limitations arise because a program, when expressed in some language, is very convoluted or difficult to follow. What kinds of programs are hard to understand when written in the language(s) you already know? (*Example:* FORTRAN 77, a very good language for numerical computation, is quite horrible at handling structured data and has no recursion, so writing a recursive-descent language parser would be impossibly complex.)