Homework 3 (due 4/2/2009)

Write two *equivalent* programs (i.e., they should do exactly the same thing, step-by-step) in C++ and Java to showcase the strengths of Garbage Collection vs. explicit memory management. Your programs should take one or more input parameters. These should be such that you can pick values for the parameters to make any one version faster than the other (for the exact same parameter values).

High-level hint: The easiest way to make Garbage Collectors perform badly is to allocate live data that are comparable in size to your RAM and to keep forcing garbage collection to occur. Conversely, the easiest way to make an explicit memory allocator do badly (relative to the GC version) is to allocate a lot of data, kill almost all of them, keep operating on the live data, which now have very bad locality.

Low-level hints: Try to have as much of an apples-to-apples comparison as possible. For instance, Java may have more of a space overhead per object. C++ objects won't even have a v-table pointer if you don't have virtual functions. Try to make your in-memory object representations as similar as possible in both languages. Java will have a heavy startup-overhead, since it will take a few calls for each method to be dynamically compiled and the dynamic compilation will take some time. You can start your measurements after the program has operated for a little while, or you can pick parameters so that your program runs for a long time and the constant startup overhead matters little. Use the -Xmx<size> option to set the maximum heap size of your Java Virtual Machine.