Java Reflection

Compile and execute Countdown. It will take a while to finish. Evidently, it spits out something useful once the countdown gets to zero. But you won't wait for it to finish. You're just not that patient.

We can use Java's reflection library to inspect the members (state and behavior) of classes. Compile and run Inspect to see this. It basically gets all of the Countdown class' declared fields (in this case, the static variable *cycle*.

We can even view the values of any member variables (including the ones found via Inspect above). Compile and run View to see this. Note that we can even modify a class member's accessibility (i.e., make it public even if it's private)!

Believe it or not, we can even change the values of member variables. Compile and run Edit to see this. Once we've done this, we can use reflection to invoke (execute) the main method with the new value(s) of member variable(s).

The end game: we've managed to change the countdown from 2 million to 10. So we'll get the output at the end in 10 seconds as opposed to a little over 23 days!

Someone actually did something like this for a popular online game (Runescape) to cheat and stock up on a lot of “free” gold. They then sold the gold on eBay for real cash! They made a lot of money (millions) before they were sued (and lost). What happened? The game designers wrote their game in Java and programmed it badly.