Speed in Matlab, C and Java

Here, you will find the code for the demonstrations from the class of (2/5/04). I created 4 files:

- speedTest.m -- a matlab loop that adds random numbers,
- speedTest2.m -- a faster version of the matlab code. The reason that it is faster is that loops are very slow in matlab, but vector-based operations are very fast. In my tests on my workstation, this one was almost as good as C.
- mex_speedTest.c -- C code to do the same thing. See the file for compilation instructions. It also provides at least one example of how C and matlab can communicate.
- SpeedTest.java -- the Java code. You compile this with javac to create SpeedTest.class, just like you would any other piece of Java code.

Here is a transcript of my session with Matlab in which I tested these out on my workstation:

```
>> n = 10^6;
>> tic; x = speedTest(n); toc

elapsed_time =

    4.0123

>> tic; x = speedTest2(n); toc

elapsed_time =

    0.1522

>> tic; x = mex_speedTest(n); toc

elapsed_time =

    0.1335

>> s = SpeedTest

s =

SpeedTest@5c8569

>> methods(s)


Methods for class SpeedTest:


SpeedTest  getClass   notify     sumRand    wait
equals     hashCode   notifyAll  toString

>> tic; s.sumRand(n); toc

elapsed_time =

    0.2651
```