C SC 397a, Spring 2010
Assignment 1
Due: Monday, January 25 at 22:00:00 MST

Problem 1. (15 points) `ctest.java`

Write a Java program `ctest` that uses the `Counter` class on slide 14 and performs one of several CPU intensive computations based on a command line argument. Later in the course you'll transliterate `ctest.java` into a C++ program and compare the performance of the two.

If `ctest` is run with the argument "1" (`java ctest 1`), it creates one hundred million instances of `Counter` and then terminates. Use the name "x" for each instance (i.e., `new Counter("x")`). No output is produced.

If run with the argument "2", `ctest` fills an array with 50,000 instances of `Counter` and then bumps each counter N+1 times, where N is the index of the counter in the array. (The first counter is bumped once, and the last counter is bumped 50,000 times.) Use the name "x" for each counter. No output is produced.

If run with the argument "3", `ctest` behaves like the "2" case but when done with all the bumping, it prints all the counters. The output should look like this:

```
x's count is 1
x's count is 2
x's count is 3
...
x's count is 49999
x's count is 50000
```

Don't bother with any error checking; assume that `ctest` is always run with 1, 2, or 3 as the command line argument.

Using `/usr/bin/time`, time three executions of "`java ctest 1`". Example:

```
$ /usr/bin/time java ctest 1
1.15user 0.13system 0:01.35elapsed 94%CPU ...
0inputs+64outputs (1major+68361minor)pagefaults 0swaps
```

Compute the average `user` time (underlined above) for the three runs to two decimal places. Repeat with "`ctest 2`" and "`ctest 3`". (Use the shell to direct the output of "`ctest 3`" into a file.)

Create a text file named `answers.txt` that has the average time for each of the three computations. In addition to noting the averages for Java, using whatever you've heard about C++, estimate the running times for the C++ version that you'll write later in the course.

Use this format for `answers.txt`:

```
Java results:
ctest 1: 1.03
ctest 2: 2.97
ctest 3: 3.32

C++ estimates:
ctest 1: 0.50
ctest 2: 1.50
ctest 3: 1.70
```

Note that you won't be graded on the accuracy of your C++ estimates—they're just for fun! The only way to lose any points on the estimates is to omit them.

IMPORTANT: `answers.txt` must be a plain text file, created with an editor like Emacs, `vi`/`vim`, or `pico`. Do not submit a PDF or a Word document, for example. If you create it on a system other than `lectura`, be sure it looks ok with "`cat answers.txt`" on `lectura`.

Miscellaneous

A reference version of `ctest` is available in the form of a jar file, `ctest.jar`. It can be found in `/cs/www/classes/cs397a/spring10/files/a1` on `lectura`, which can also be reached via the Files of various sorts link on the class website.

The jar can be run like this:

```
$ java -jar ctest.jar 1
```

Note that Java 6 is installed on `lectura`, and `ctest.jar` contains Java 6 class files. If you try to run the jar with an older JVM you'll get a "Bad version number in .class file" error.

No comments are needed in `ctest.java`.

Deliverables

Use `turnin` with the tag `397a_1` to submit your solutions for grading. The deliverables for this assignment are `ctest.java` and `answers.txt`. To keep things simple, have `ctest.java` contain both the `Counter` and `ctest` classes; `ctest` is public, `Counter` is not.

Here's an example of a `turnin` submission:

```
$ turnin 397a_1 ctest.java answers.txt
Turning in:
 ctest.java -- ok
 answers.txt -- ok
All done.
```