Exercise #3: Arithmetic Operations

WARNING: CURRENTLY THE FORTRAN VERSION IS FAILING. A SOLUTION HASN'T YET BEEN FOUND. USE THE C VERSION ONLY, EVEN IF YOU PREFER FORTRAN.

In this exercise, we'll use the same conventions and commands as in previous exercises. You should refer back to the previous exercise descriptions for details on various Unix commands. You **MUST** complete the previous exercises **BEFORE** starting this exercise. For this exercise, **YOU ARE EXPECTED TO KNOW HOW TO ACCOMPLISH BASIC TASKS**, based on your experiences with previous exercises.

In the exercise, you'll benchmark various arithmetic operations, using various compilers and levels of compiler optimization.

Specifically, you'll benchmark using the following compilers:

- the GNU C compiler, `gcc`, or Fortran compiler, `gfortran`, for various optimization levels;
- the Intel C compiler, `icc`, or Fortran compiler, `ifort`, for various optimization levels;
- the Portland Group C compiler, `pgcc`, or Fortran compiler, `pgf90`, for various optimization levels.

Here are the steps for this exercise:

1. Log in to the Linux cluster supercomputer (`boomer.oscer.ou.edu`).
2. Copy the `ArithmeticOperations` directory:
   ```
   cp  -r  ~hneeman/SIPE/ArithmeticOperations/  ~/SIPE/
   ```
3. Choose which language you want to use (C or Fortran90), and `cd` into the appropriate directory:
   ```
   cd  ~/SIPE/ArithmeticOperations/C/
   ```
   OR:
   ```
   cd  ~/SIPE/ArithmeticOperations/Fortran90/
   ```
4. Edit the batch script `arithmetic_operations.bsub` so that it contains your username and your e-mail address.
5. Compile, using the ***shell script*** named `make_cmd` (a shell script is a file containing a sequence of Unix commands), which in turn invokes the `make` command:
   ```
   make_cmd
   ```
   If that doesn't work, try this:
   ```
   ./make_cmd
   ```
6. Submit the batch job:
   ```
   bsub  <  arithmetic_operations.bsub
   ```
7. Once the batch job completes, examine the several output files to see the timings for your runs with executables created by the various compilers under the various levels of optimization.
8. Use your favorite graphing program (for example, Microsoft Excel) to create graphs of your various runs, so that you can compare the various methods visually.