**Blue Waters Petascale Semester Curriculum v1.0**

**Unit 4: OpenMP**

**Lesson 12: OpenMP Tasks**

**Exercise Instructions for Students**

*Developed by Cameron Foss for the Shodor Education Foundation, Inc.*



*Except where otherwise noted, this work by The Shodor Education Foundation, Inc. is licensed under CC BY-SA 4.0. To view a copy of this license, visit*[*https://creativecommons.org/licenses/by-sa/4.0*](https://creativecommons.org/licenses/by-sa/4.0)

*Browse and search the full curriculum at*[*http://shodor.org/petascale/materials/semester-curriculum*](http://shodor.org/petascale/materials/semester-curriculum)

*We welcome your improvements! You can submit your proposed changes to this material and the rest of the curriculum in our GitHub repository at*[*https://github.com/shodor-education/petascale-semester-curriculum*](https://github.com/shodor-education/petascale-semester-curriculum)

*We want to hear from you! Please let us know your experiences using this material by sending email to* [*petascale@shodor.org*](mailto:petascale@shodor.org)

Assignment:

1. Add OpenMP directives to the Fibonacci (or Compute Pi) codes to parallelize them.
2. Use OpenMP tasks to parallelize the recursive part of the code.
3. Once the code is parallelized, generate the following plots:
   1. Runtime as a function of the number of threads.
   2. Runtime as a function of problem size (increase N in either code) for 2, 4 and 8 threads.

(the above is preliminary, and needs testing)

Compiling and Running:

See the ‘compile’ script for examples on how to compile the serial and parallelized codes.

Once the parallelized code is compiled, make sure to run:

$ export OMP\_NUM\_THREADS=N

To set the number of OpenMP threads to N (some integer). Then the code can be run with

$ ./foo.exe

where ‘foo’ is the filename of the executable.