5)

\*\*\* Real is the wall clock time, from start to finish of the call

\*\*\* User is the amount of CPU time spent outside the kernel within the process

\*\*\* Sys is the amount of CPU time spent in the kernel within the progress

\*\*\* User+Sys is the total CPU time the process used. If multi-thread, might exceed t he wall clock time.

Times for each implementation:

time -p ./dd\_serial : 122.49

time -p ./dd\_omp 1: Real: 0.02, User: 0.01

time -p ./dd\_threads 1: Real: 0.02, User: 0.01

time -p ./dd\_omp 2: Real: 0.02, User: 0.02, Sys: 0.02

time -p ./dd\_threads 2: Real: 0.02, User: 0.01, Sys: 0.01

time -p ./dd\_omp 3: Real: 0.04, User: 0.13, Sys: 0.00

time -p ./dd\_threads 3: Real: 0.04, User: 0.08, Sys: 0.03

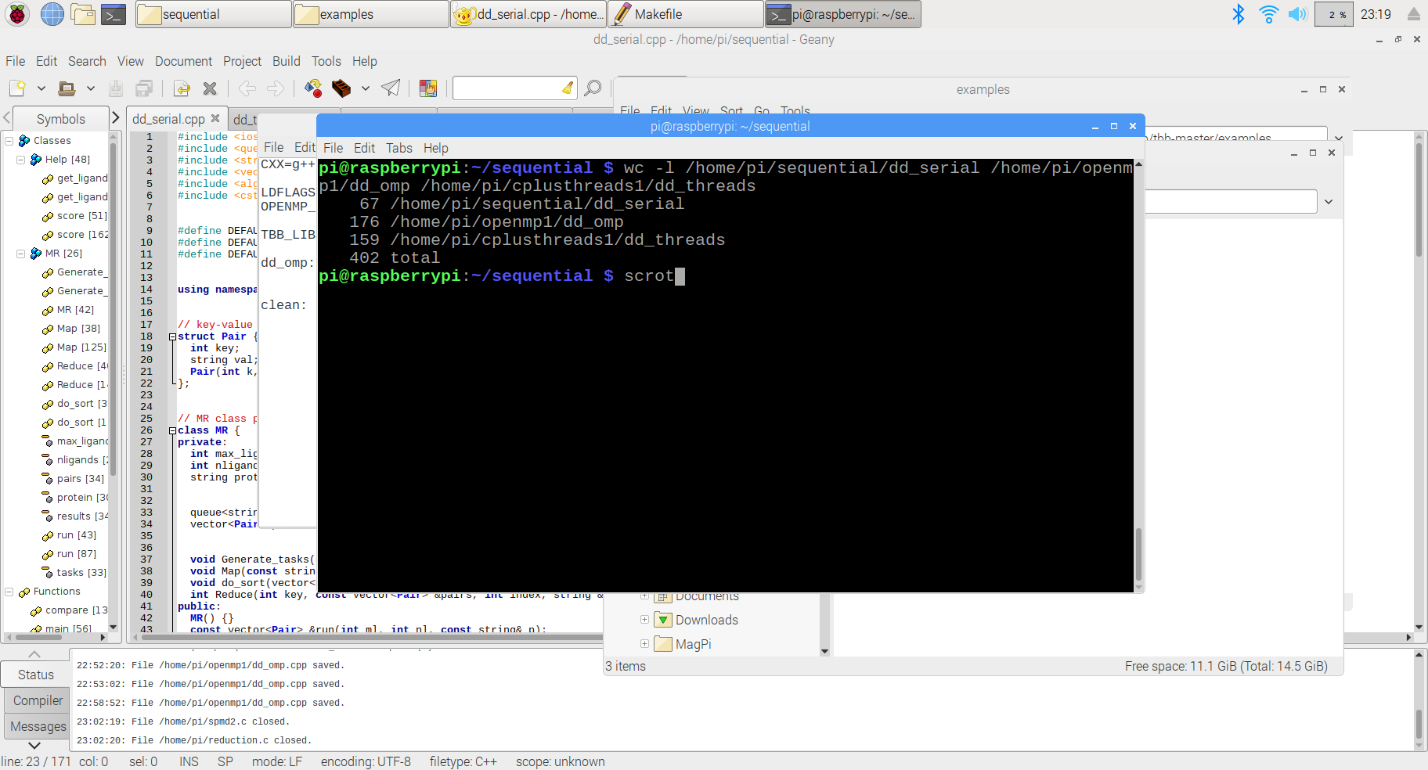
time -p ./dd\_omp 4: Real: 0.21, User: 0.56, Sys: 0.01

time -p ./dd\_threads 4: Real: 0.16, User: 0.56, Sys: 0.01

2.3 Discussion Questions  
 1) Which approach is the fastest?

Using the total of User and Sys time, dd\_threads is the fastest. The user time increases less than the dd\_omp user time, even with a longer sys time.

2) Determine the number of lines in each file using wc-1. How does the C++11 implementation compare to the OpenMP implementations?



3) Increase the number of threads to 5. What is the run time for each?

dd\_omp 5: Real: 0.95, User: 2.6, Sys: 0.02

dd\_threads 5: Real: 0.68, User: 2.58, Sys: 0.01

The run times using User+Sys for dd\_omp is 2.62, and for dd\_threads is 2.59.

Using real, the times are 0.95 for dd\_omp, and 0.68 for dd\_threads.

4) Increase the maximum ligand length to 7, and rerun each program. What is the run time for each?

The default maximum ligand length is already 7 in each of the programs, so there are no reported changes.