**Test task**

1 – A 200Gb file filled with text lines divided by “\n”.

2 – Lexicographically sort lines of this file.

3 – Accept input/output file name (source\_file.txt, sorted\_file.txt)

+ work with small files, empty files, files much bigger than 200Gb

+ work well with non-ascii symbols

+ assume that zero-byte is not present in the file

+ assume that memory limit (8Gb of RAM) is much bigger than the size of the longest line

4 – Simple main.cpp solution

5 – Note

+ Use External Sort

+ Care about bottle necks from IO

+ Use multiple threads (mutex lock)

+ Use efficient Sorting Algorithms (both time and memory consumption)

+ Use Sorting Algorithms from standard libraries

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Problem 1. Lexicographically sort text lines

+ compare 2 text lines:

std::lexicographical\_compare

<http://www.cplusplus.com/reference/algorithm/lexicographical_compare/>

work well with non-ascii symbols? => Need to test

+ compare non-ascii symbols:

Write a function my\_comp() for std::lexicographical\_compare

<https://stackoverflow.com/questions/39971585/sorting-a-vector-of-strings-by-the-first-letter-in-non-ascii-order-in-c>

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Problem 2: External sort