Oblig 1 IN3030 – Find the k largest numbers in a large array

The CPU used for this oblig is AMD Ryzen 7 4700U with Radeon Graphics, 2 GHz, 8 kjerne(r), 8 logiske prosessor(er)

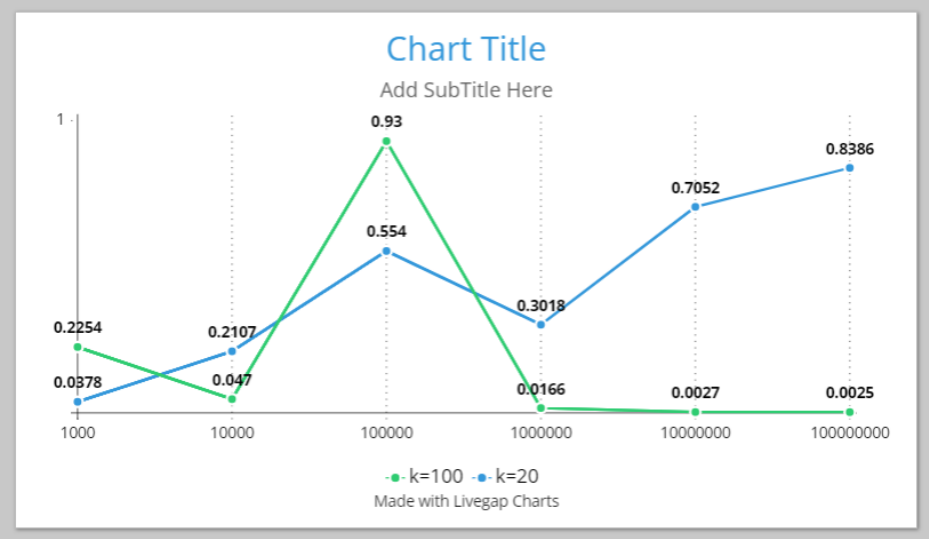
Table for k = 20

	Arrays.sort()	Sequential insertion	Parallel insertion
n = 1000	0.2736ms	0.0366ms	0.9681ms
n = 10 000	0.6453ms	0.1824ms	0.8653ms
n = 100 000	22.7007ms	0.6018ms	1.0862ms
n = 1 000 000	59.4162ms	0.7863ms	2.605ms
n = 10 000 000	755.0349ms	6.0165ms	8.5308ms
n = 100 000 000	8993.4068ms	57.1636ms	68.1578ms

Table for k = 100

	Arrays.sort()	Sequential insertion	Parallel insertion
n = 1000	0.2675ms	0.2728ms	1.2101ms
n = 10 000	0.6162ms	0.2203ms	4.6848ms
n = 100 000	20.5456ms	2.0381ms	2.1913ms
n = 1 000 000	57.3026ms	0.6163ms	37.1251ms
n = 10 000 000	757.3357ms	6.0037ms	2204.3365ms
n = 100 000 000	8768.4719ms	58.0377ms	NaN

Graph:



Observations:

My parallel code is still not correct as shown in the tables above. I don't know how to solve it properly. I think it has something to do with how I sort for each thread and then how I find the k largest numbers from all for the chunks.

If we compare the two tables, we see that for table where k = 100 the algorithms use more time than when k = 20. I would have thought it would be a larger difference, but that might just be because I have not done it correctly.