Lab 2 Report – Paul Schmid

Question 1:

Question 1 was figuring out the performance increase of parallelization for the SOR benchmark. In order to test this, I tested the running time for no parallelization, parallelization with no schedule, a static schedule, a dynamic schedule, a dynamic schedule with a chuck size of 1, and a dynamic schedule with a chuck size of 4. This showed that for some reason, for this code no parallelization was faster than anything and dynamic scheduling is faster than static and has a closer standard deviation.

	Trail 1	Trail 2	Trail 3	Trail 4	Trail 5	Average	Standard Deviation
No Parallel	0.003522	0.003512	0.003492	0.003492	0.003488	0.003501	1.33626E-05
No Schedule	0.94779	0.858958	1.087083	1.232965	0.792045	0.983768	0.158998613
Static Schedule	1.024934	1.048415	0.671435	0.68162	1.184688	0.922218	0.20791646
Dynamic Schedule	1.382234	0.270077	0.850301	1.037799	0.805361	0.869154	0.361942111
Dynamic,1 Schedule	0.904987	0.148707	0.844831	0.871379	0.264619	0.606905	0.32939741
Dynamic, 4 Schedule	0.838456	0.912085	0.245922	0.931476	1.011686	0.787925	0.276554602

Question 2:

Question 2 has a code output 1 – 20, via thread, in order with multiples of 5 being run in a separate thread. I started by making two sections, one that output 5, 10, 15, and 20 and one that output the rest. I then added 2 locks to make sure that the code executed at the desired time.