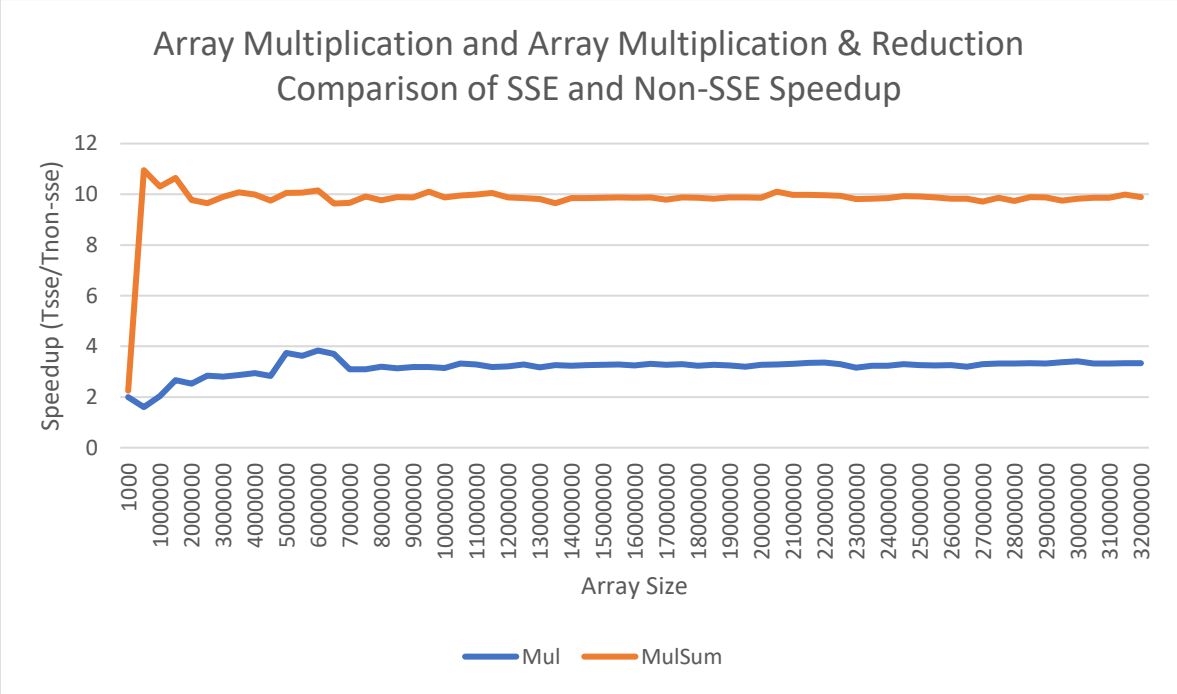
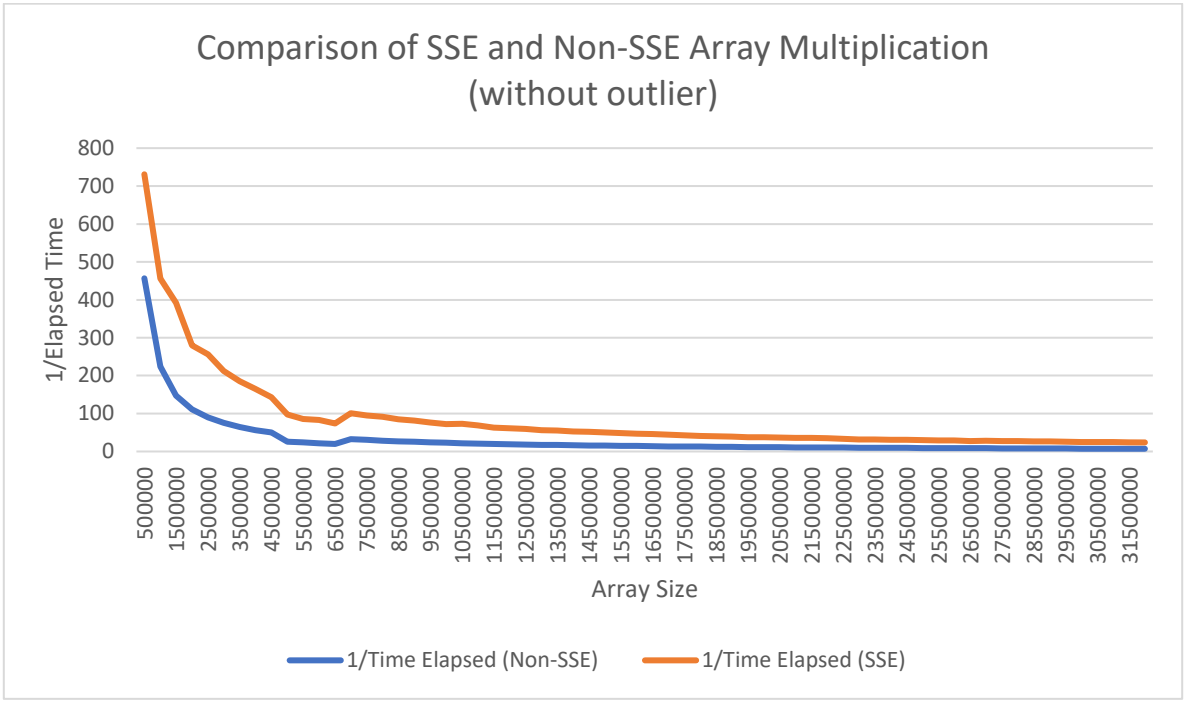
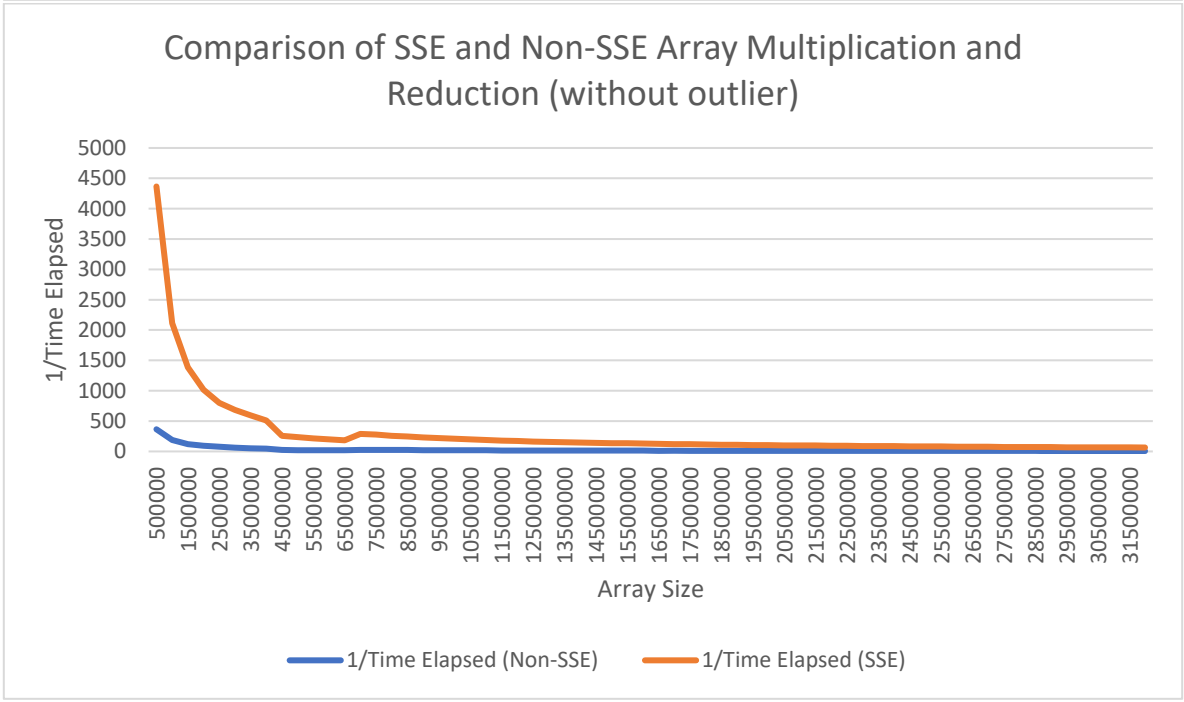
Taylor Kirkpatrick

Project 5

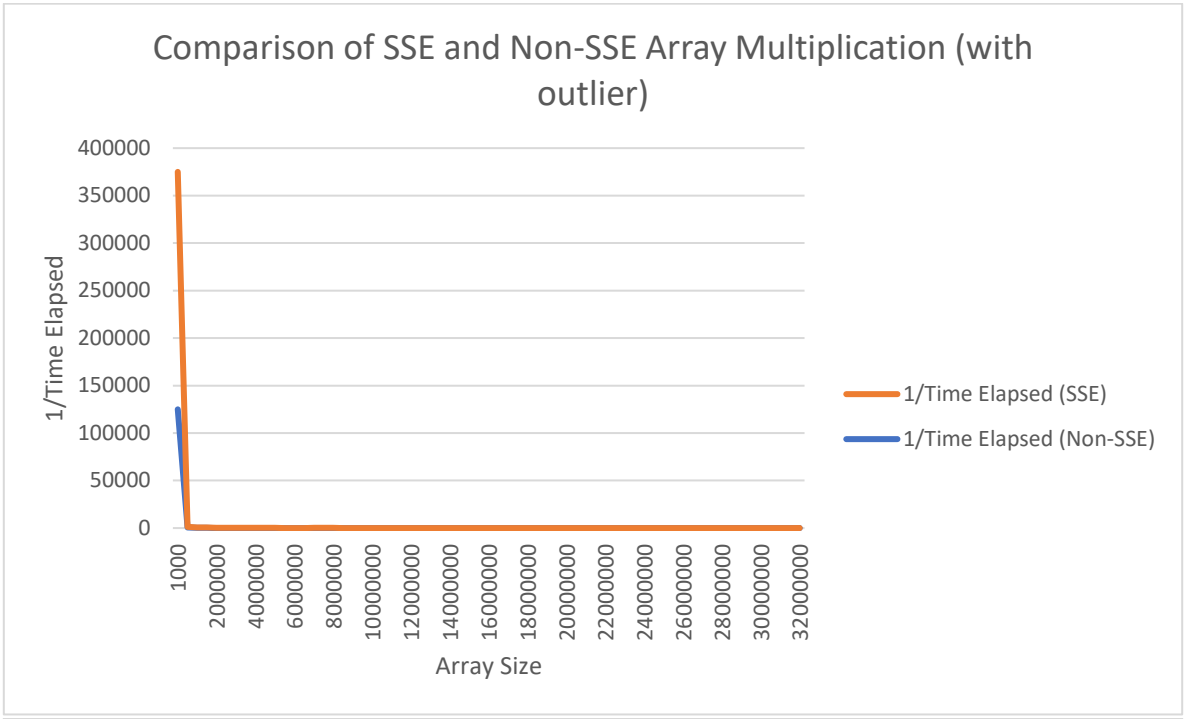
1. This was run on OSU's flip server. Average system load during the testing of this program was 1.35 (found using the uptime command occasionally during testing). Flip has 24 processors with 6 cores per processor for a total of 144 cores. The size of the caches are as follows: L1: 32K, L2: 256K, L3: 12288K (cache sizes found using the lscpu command).
2. Note that the instructions were unclear on what data was to be plotted on the graph. Below are three graphs that could show relevant trends and assist in understanding the concepts at work here.

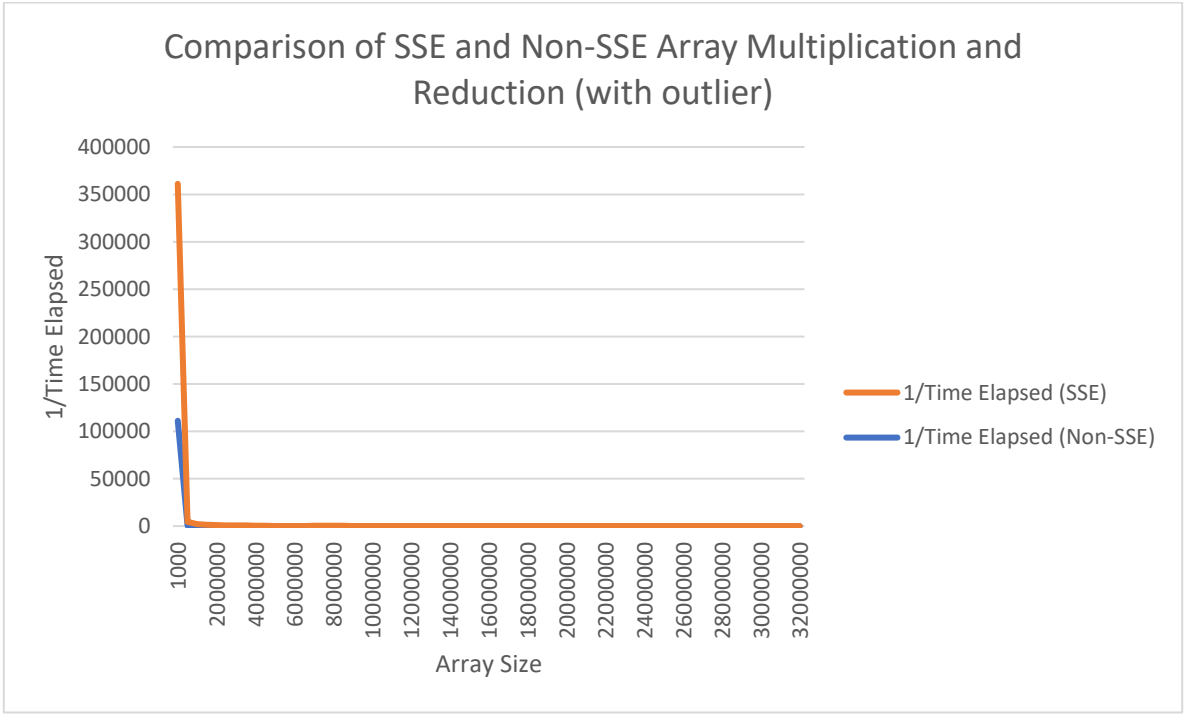






It is important to note that the last two graphs have “(without outlier)” in their titles. Below are the same graphs plotted with the outlier entry found at an array size of 1000.





As these versions are significantly less helpful at visualizing the data as the first two, both versions were plotted and added to this report.

Table:

Array Size	Time Elapsed (Non-SSE)	Time Elapsed (SSE)	1/Time Elapsed (Non-SSE)	1/Time Elapsed (SSE)	Speedup	Function
1000	0.000008	0.000004	125000	250000	2	Mul
500000	0.00219	0.001368	456.6210046	730.994152	1.600877193	Mul
1000000	0.00446	0.002192	224.2152466	456.2043796	2.034671533	Mul
1500000	0.006788	0.00255	147.3187979	392.1568627	2.661960784	Mul
2000000	0.009032	0.003574	110.7174491	279.798545	2.527140459	Mul
2500000	0.011098	0.003898	90.10632546	256.5418163	2.847101077	Mul
3000000	0.013231	0.00472	75.58007709	211.8644068	2.803177966	Mul
3500000	0.015466	0.005395	64.65795939	185.3568119	2.866728452	Mul
4000000	0.017783	0.006053	56.23348141	165.2073352	2.937882042	Mul
4500000	0.019767	0.006996	50.58936612	142.9388222	2.825471698	Mul
5000000	0.038365	0.010245	26.06542421	97.60858956	3.744753538	Mul
5500000	0.042309	0.011681	23.63563308	85.60910881	3.622035785	Mul
6000000	0.046059	0.012009	21.71128335	83.27088017	3.83537347	Mul
6500000	0.049863	0.013474	20.05495056	74.21701054	3.700682796	Mul
7000000	0.030603	0.009892	32.67653498	101.0917913	3.093712091	Mul
7500000	0.032725	0.010562	30.55767762	94.67903806	3.098371521	Mul
8000000	0.034902	0.010905	28.6516532	91.70105456	3.200550206	Mul
8500000	0.0371	0.011842	26.9541779	84.44519507	3.132916737	Mul
9000000	0.039276	0.012315	25.46084123	81.20178644	3.189281364	Mul
9500000	0.041742	0.013131	23.95668631	76.15566217	3.17888965	Mul
10000000	0.043765	0.013912	22.84930881	71.88039103	3.145845313	Mul
10500000	0.045781	0.013781	21.84312269	72.56367462	3.322037588	Mul
11000000	0.048002	0.014604	20.83246531	68.47439058	3.286907697	Mul
11500000	0.050458	0.015824	19.81846288	63.19514661	3.188700708	Mul
12000000	0.052384	0.016358	19.08979841	61.13216775	3.202347475	Mul
12500000	0.054939	0.016739	18.20200586	59.74072525	3.282095705	Mul
13000000	0.056712	0.017909	17.63295246	55.83784689	3.166675973	Mul
13500000	0.05906	0.018154	16.93193363	55.08427895	3.253277515	Mul
14000000	0.061131	0.018895	16.35831248	52.92405398	3.235300344	Mul
14500000	0.06326	0.019383	15.80777743	51.59160089	3.263684672	Mul
15000000	0.065587	0.020077	15.24692393	49.80823828	3.266772924	Mul
15500000	0.067633	0.020577	14.78568155	48.59794917	3.286825096	Mul
16000000	0.069794	0.021489	14.32787919	46.53543674	3.247894271	Mul
16500000	0.072022	0.021788	13.88464636	45.89682394	3.305581054	Mul
17000000	0.074248	0.022655	13.46837625	44.14036637	3.277333922	Mul
17500000	0.076573	0.023258	13.05943348	42.99595838	3.292329521	Mul

18000000	0.078555	0.024267	12.72993444	41.20822516	3.237112128	Mul
18500000	0.08149	0.024885	12.27144435	40.18485031	3.274663452	Mul
19000000	0.083037	0.025623	12.04282428	39.02743629	3.240721227	Mul
19500000	0.085102	0.026634	11.75060516	37.54599384	3.195239168	Mul
20000000	0.087285	0.026696	11.45672223	37.45879533	3.26959095	Mul
20500000	0.0896	0.027317	11.16071429	36.60724091	3.280008786	Mul
21000000	0.093043	0.028094	10.7477188	35.59478892	3.311845946	Mul
21500000	0.093854	0.028079	10.65484689	35.61380391	3.342497952	Mul
22000000	0.096207	0.028684	10.39425406	34.86264119	3.354030121	Mul
22500000	0.098253	0.029857	10.17780628	33.49298322	3.29078608	Mul
23000000	0.100511	0.031814	9.949159793	31.43270258	3.159332369	Mul
23500000	0.102533	0.031719	9.752957584	31.52684511	3.23254201	Mul
24000000	0.104805	0.032429	9.541529507	30.83659687	3.231829535	Mul
24500000	0.106924	0.032477	9.352437245	30.79102134	3.292299166	Mul
25000000	0.109136	0.033473	9.162879343	29.87482449	3.260418845	Mul
25500000	0.111285	0.034346	8.985937009	29.11547196	3.240115297	Mul
26000000	0.113605	0.034811	8.802429471	28.72655195	3.263479935	Mul
26500000	0.115937	0.036341	8.625374126	27.51712941	3.190253433	Mul
27000000	0.118102	0.03579	8.467257117	27.94076558	3.299860296	Mul
27500000	0.120069	0.036127	8.32854442	27.68012844	3.323525341	Mul
28000000	0.122771	0.037007	8.145246027	27.02191477	3.317507499	Mul
28500000	0.125764	0.037748	7.951401037	26.49146975	3.331673201	Mul
29000000	0.126544	0.038029	7.902389683	26.29572169	3.327565805	Mul
29500000	0.131596	0.038986	7.599015168	25.65023342	3.375468117	Mul
30000000	0.135061	0.039602	7.404061868	25.25124994	3.410459068	Mul
30500000	0.133124	0.040132	7.511793516	24.91777135	3.317153394	Mul
31000000	0.135538	0.040865	7.378004692	24.47081855	3.316725804	Mul
31500000	0.137559	0.041262	7.26960795	24.23537395	3.333793805	Mul
32000000	0.139911	0.041937	7.147400848	23.84529175	3.336218614	Mul
1000	0.000009	0.000004	111111.1111	250000	2.25	MulSum
500000	0.002738	0.00025	365.230095	4000	10.952	MulSum
1000000	0.005345	0.000519	187.090739	1926.782274	10.29865125	MulSum
1500000	0.008412	0.00079	118.8777936	1265.822785	10.64810127	MulSum
2000000	0.010607	0.001085	94.277364	921.6589862	9.776036866	MulSum
2500000	0.013312	0.001379	75.12019231	725.1631617	9.653372009	MulSum
3000000	0.015991	0.001615	62.53517604	619.1950464	9.901547988	MulSum
3500000	0.018654	0.001852	53.6078053	539.9568035	10.07235421	MulSum
4000000	0.021568	0.00216	46.36498516	462.962963	9.985185185	MulSum
4500000	0.041806	0.004291	23.92001148	233.04591	9.742717315	MulSum
5000000	0.04655	0.004631	21.48227712	215.9360829	10.05182466	MulSum

5500000	0.051118	0.005081	19.5625807	196.8116512	10.06061799	MulSum
6000000	0.056337	0.005552	17.75032394	180.1152738	10.14715418	MulSum
6500000	0.057871	0.006008	17.279812	166.4447403	9.632323569	MulSum
7000000	0.036741	0.003801	27.21754988	263.0886609	9.666140489	MulSum
7500000	0.039504	0.003986	25.31389226	250.8780733	9.910687406	MulSum
8000000	0.042003	0.004304	23.80782325	232.3420074	9.759061338	MulSum
8500000	0.044617	0.004511	22.4129816	221.680337	9.890711594	MulSum
9000000	0.04731	0.004789	21.1371803	208.8118605	9.878889121	MulSum
9500000	0.050958	0.005044	19.62400408	198.2553529	10.10269627	MulSum
10000000	0.052458	0.005309	19.06286934	188.3593897	9.880956866	MulSum
10500000	0.055093	0.005535	18.15112628	180.6684734	9.953568202	MulSum
11000000	0.057728	0.005777	17.32261641	173.100225	9.992729791	MulSum
11500000	0.0613	0.006101	16.3132137	163.9075561	10.04753319	MulSum
12000000	0.06295	0.006375	15.88562351	156.8627451	9.874509804	MulSum
12500000	0.066335	0.006733	15.07499812	148.5222041	9.852220407	MulSum
13000000	0.069108	0.007039	14.47010476	142.0656343	9.817871857	MulSum
13500000	0.070868	0.007342	14.1107411	136.2026696	9.652410787	MulSum
14000000	0.073492	0.007458	13.6069232	134.0842049	9.854116385	MulSum
14500000	0.076176	0.007734	13.12749422	129.2991983	9.849495733	MulSum
15000000	0.078794	0.007986	12.69132167	125.2191335	9.866516404	MulSum
15500000	0.081396	0.008245	12.285616	121.2856277	9.872164948	MulSum
16000000	0.084137	0.008535	11.88537742	117.1646163	9.85787932	MulSum
16500000	0.086597	0.008772	11.54774415	113.999088	9.871979024	MulSum
17000000	0.089218	0.009116	11.20850053	109.6972356	9.786967968	MulSum
17500000	0.091909	0.009304	10.88032728	107.4806535	9.878439381	MulSum
18000000	0.094516	0.009578	10.58021922	104.4059303	9.868030904	MulSum
18500000	0.097271	0.009903	10.28055638	100.9795012	9.822377057	MulSum
19000000	0.099753	0.010106	10.02476116	98.95111815	9.870670889	MulSum
19500000	0.102371	0.010372	9.768391439	96.41342075	9.869938295	MulSum
20000000	0.104976	0.010642	9.525986892	93.96729938	9.86431122	MulSum
20500000	0.110067	0.010895	9.085375271	91.78522258	10.10252409	MulSum
21000000	0.110223	0.011051	9.072516625	90.48954846	9.9740295	MulSum
21500000	0.112923	0.011325	8.855591863	88.30022075	9.971125828	MulSum
22000000	0.115502	0.011593	8.657858738	86.25894937	9.96308117	MulSum
22500000	0.118893	0.011964	8.410924108	83.58408559	9.937562688	MulSum
23000000	0.120697	0.012303	8.285210072	81.28098838	9.810371454	MulSum
23500000	0.123281	0.012542	8.111550036	79.73210014	9.829453038	MulSum
24000000	0.125988	0.012784	7.937263866	78.22277847	9.855131414	MulSum
24500000	0.130078	0.0131	7.687695075	76.33587786	9.929618321	MulSum
25000000	0.131942	0.013311	7.579087781	75.12583577	9.912253024	MulSum

25500000	0.134	0.013576	7.462686567	73.65939894	9.870359458	MulSum
26000000	0.136644	0.013916	7.318286935	71.85972981	9.81920092	MulSum
26500000	0.13951	0.014203	7.16794495	70.40766035	9.822572696	MulSum
27000000	0.141935	0.014623	7.045478564	68.38542023	9.70628462	MulSum
27500000	0.144795	0.014676	6.906315826	68.13845735	9.866107931	MulSum
28000000	0.14717	0.015123	6.794863084	66.12444621	9.731534748	MulSum
28500000	0.149973	0.015168	6.667866883	65.92827004	9.887460443	MulSum
29000000	0.152493	0.015445	6.557678057	64.74587245	9.873292328	MulSum
29500000	0.154862	0.015894	6.457362038	62.91682396	9.743425192	MulSum
30000000	0.157728	0.016057	6.340028403	62.27813415	9.823005543	MulSum
30500000	0.160145	0.016239	6.244341066	61.58014656	9.861752571	MulSum
31000000	0.162928	0.016513	6.137680448	60.55834797	9.866650518	MulSum
31500000	0.167573	0.016783	5.967548471	59.58410296	9.984686886	MulSum
32000000	0.168465	0.017044	5.935951088	58.67167332	9.884123445	MulSum

*Note that “Mul” stands for “Multiplication” and “MulSum” stands for “Multiplication & Reduction”

3. In the speedups for both Multiplication and Multiplication + Reduction, both speedups tend towards a horizontal asymptote, with multiplication averaging around 3.3 and multiplication + reduction averaging around 10. While the speedup is not changing, it is still important to note that this is a speedup, not a flat time value. This means that the multiplication + reduction SIMD SSE function runs consistently around 10 times faster no matter the array size. The same is true for the multiplication function, but only 3.3 times as fast. An interesting bump is also found in the graph (pictured to the right), but this is presumed to be a drop in performance due to high server usege. The dataset was not redone because this bump still shows an interesting behavior of the SIMD SSE functions and the normal ones. Even with a busier server, the more efficient functions run faster and are less impacted by the slowdown than the inefficient functions. Had the efficient functions less efficient but still better than the normal functions, they would continue to show the regular speedup here. Instead, while the literal time elapsed is of course longer, the speedup actually increases when everything is supposed to get slower! This is a very interesting behavior.

4. As can be seen by the horizontal asymptote, the speedup is consistent across all tested array sizes. Note that it is may be possible for the asymptote to change at even larger array sizes should the program encounter a new issue independent of the SIMD SSE/regular comparison, but such a change is not present in this data.
5. Before mentioning the SIMD elements to this, I think that the overhead provided by the non-SSE functions being entirely in C++ also likely plays into this consistent speedup. While the functions provided for this assignment are still in C files, they skip much of the overhead needed to translate C operations to more machine-readable language. The main

reason for this consistent speedup is most likely the cause of the SIMD nature of the more efficient functions. While the regular functions instead worked as SISD, inefficiently multiplying each individual array value to a different array value with a separate operation per array value, the SIMD functions are capable of multiplying many (if not all) elements of one array to the other array in a single operation. Assuming an operation in both function versions take a similar amount of time, the SIMD version is shockingly more efficient, effectively doing more operations per cycle than the SISD version.

6. The SSE SIMD multiplication function hovers around a speedup of 3.3. While this is, of course, good, it doesn't seem to reach the speed seen by the other function. This is likely because the multiplication function is running into cache issues by violating temporal coherence. The size of the L1 cache on the computer I tested this on is 32Kb, or 32000 bits, or 1000 floating point numbers. In this case, it is likely that the function's speed was throttled by how fast the L1 cache could be filled. So while the process was waiting for the L1 cache to be restocked with new data, it was doing just that: waiting. Thus, a limit on the speed.
7. This issue is clearly not visible in the multiplication & reduction function, however. While the speedup of 10 it averages at may also be caused by a hardware limitation somewhere (perhaps the limit of the L2 cache?) it lacks the lower throttling present on the multiplication function. I believe this to be because this function is, in fact, slightly slower. While it does the same multiplication operation as the multiplication function, the multiplication & reduction function also must add values. This is another operation. This operation also takes advantage of the lower C overhead and the SIMD efficiency, so it is still much faster than the normal function, it is slower than the multiplication function. This however works to its advantage. Where the multiplication function runs too fast and exhausts the L1 cache causing a period of waiting for it to be restocked, the multiplication & reduction function is busy doing a summation operation while the L1 cache is filled with the data that will actually be used by the function. Thus the multiplication & reduction function avoids the waiting period that the multiplication function has to experience.