Debug Mode

Run # (2^20)	C kernel Clock Ticks/Second	ASM Kernel Clock Ticks/Second	Run # (2^24)	C kernel Clock Ticks/Second	ASM Kernel Clock Ticks/Second	Run # (2^26)	C kernel Clock Ticks/Second	ASM Kernel Clock Ticks/Second
1	3	2	1	53	16	1	908	110
2	3	2	2	61	26	2	208	120
3	3	2	3	38	21	3	220	149
4	3	1	4	37	19	4	210	98
5	3	2	5	37	25	5	201	116
6	2	2	6	39	22	6	220	117
7	2	1	7	37	22	7	207	110
8	2	1	8	39	16	8	215	84
9	2	1	9	41	17	9	195	122
10	3	1	10	38	14	10	210	123
11	2	2	11	44	23	11	213	116
12	2	2	12	40	16	12	214	115
13	3	1	13	39	20	13	208	103
14	2	2	14	43	29	14	216	120
15	2	1	15	42	21	15	207	114
16	3	2	16	43	33	16	217	115
17	2	2	17	48	17	17	211	112
18	3	1	18	40	14	18	218	116
19	3	1	19	40	21	19	216	115
20	2	2	20	39	15	20	210	116
21	3	2	21	39	19	21	211	115
22	3	2	22	44	18	22	216	107
23	3	1	23	39	20	23	218	108
24	3	3	24	43	20	24	204	111
25	3	1	25	55	22	25	220	114
26	3	1	26	41	23	26	210	117
27	3	3	27	41	27	27	223	114
28	3	2	28	41	25	28	211	113
29	2	1	29	40	23	29	208	112

30	3	1	30	40	20	30	202	118
Average Runtime (CPU cycles)	**2.6**	**1.6**	**Average Runtime (CPU cycles)**	**42.0**	**20.8**	**Average Runtime (CPU cycles)**	**234.9**	**114.0**

Release Mode								
Run # (2^20)	**C kernel Clock Ticks/Second**	**ASM Kernel Clock Ticks/Second**	**Run # (2^24)**	**C kernel Clock Ticks/Second**	**ASM Kernel Clock Ticks/Second**	**Run # (2^26)**	**C kernel Clock Ticks/Second**	**ASM Kernel Clock Ticks/Second**
1	2	2	1	30	32	1	1897	1683
2	1	1	2	28	29	2	120	124
3	3	1	3	22	28	3	116	63
4	5	2	4	20	19	4	617	117
5	2	1	5	21	21	5	1362	115
6	2	1	6	30	16	6	163	117
7	3	1	7	23	15	7	141	78
8	2	2	8	34	22	8	101	70
9	2	1	9	16	16	9	140	65
10	1	1	10	25	20	10	144	91
11	1	1	11	24	22	11	144	80
12	1	1	12	31	20	12	145	119
13	2	2	13	25	17	13	130	125
14	1	2	14	21	20	14	107	87
15	1	1	15	18	29	15	87	74
16	3	2	16	33	22	16	105	96
17	1	2	17	23	28	17	147	79
18	2	1	18	21	22	18	147	87
19	1	1	19	28	24	19	122	101
20	2	1	20	24	20	20	89	69
21	2	2	21	21	21	21	138	119
22	2	1	22	27	39	22	143	84
23	2	2	23	27	16	23	89	74
24	2	1	24	18	14	24	91	129

25	1	1
26	4	1
27	2	1
28	2	2
29	1	1
30	2	1
Average Runtime (CPU cycles)	**1.9**	**1.3**

25	24	24
26	22	11
27	24	30
28	25	12
29	30	16
30	28	29
Average Runtime (CPU cycles)	**24.8**	**21.8**

25	140	117
26	143	100
27	136	86
28	138	90
29	81	70
30	135	74
Average Runtime (CPU cycles)	**241.9**	**146.1**

Analysis

For the analysis, the pair opted to use the processor time (CPU Cycles) which can be read as ticks per seconds for the running time of the two kernels. The highest vector size that the machines for both members to handle was 2^26.

Upon running the two kernels in Debug and Release mode, it is evident that the x86-64 kernel performs faster across different data sizes compared to the C kernel. In Debug mode, the x86-64 kernel has average runtimes of 1.6 ticks per second for 2^20, 20.8 ticks per second for 2^24 and 114.0 ticks per second for 2^26 data size. Meanwhile in Release mode, the average runtimes of the x86-64 kernel are 1.3 ticks per second for 2^20, 21.8 ticks per second for 2^24 and 146.1 ticks per second for 2^26 data size.

It can also be observed that as the data size increases, the performance gap between the x86-64 and the C kernel becomes more significant. Therefore, it is safe to say that the x86-64 kernel is capable and more effective when it comes to handling larger datasets.

Overall, the performance of the Assembly Kernel is more efficient and faster in terms of performance time than the C kernel. The effectiveness of the Assembly Kernel significantly increases as the vector size increases.