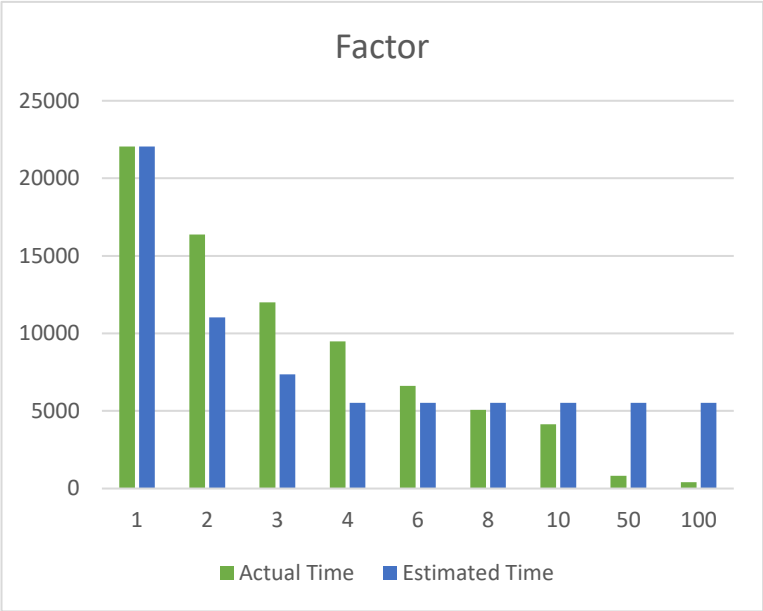
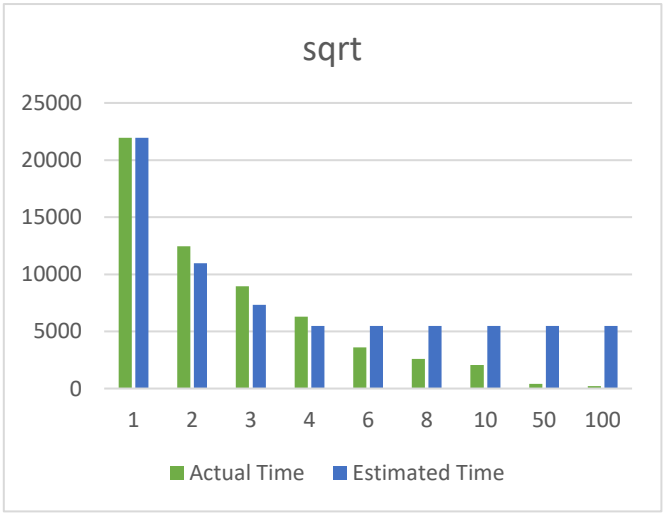
Factor	1	2	3	4	6	8	10	50	100
Estimated Time	22060	11030	7353	5515	5515	5515	5515	5515	5515
Actual Time	22060	16380	12010	9480	6620	5080	4140	820	410



sqrt	1	2	3	4	6	8	10	50	100
Actual Time	21950	12450	8950	6290	3590	2600	2070	410	200
Estimated Time	21590	10975	7320	5490	5490	5490	5490	5490	5490



Question 1: Is the hypothesis correct that equation 1 will work: Total_Time[X children] = Total_Time[1 child] / minimum(#processors, #processes)?

While it appears to be a good indicator from a general sense of how runtime scales in relation to the # of processors, it isn't capable of estimating run time past 4 processors given the constraints of the hypothesis and equation. Using the data we can see that even though there are only 4 physical processors, the "actual time" data shows that the program is still gaining efficiceny even as the number of processes surpasses the number of cores on the CPU.

Question 2: Why is this correct or incorrect?

This is incorrect because the hypothesis is assuming that with 4 cores efficiency in total runtime should plateau at anything greater than 4 processes. While we can clearly see that the total actual runtime when the number of processes is greater than 4 is still becoming more and more efficient. This is has to do with how the number of processes through each iteration appears to be increasing, but what is really happening is the program is just dividing up the size of the workload in to smaller chunks. Making it more manageable for the work to be divided up amongst

Question 3: What have you learned from this experiment? How will you use your gained knowledge in the future? Summarize.

I learned that a cpu can gain efficicency in runtime/execution past the number of physical cores the code is running on. This has to do with underlying intricacies in how the cpu breaks the workload into more manageable pieces. I will use this gained knowledge in the future by understanding that physical limitations aren't always the bottleneck, and while at face value it can appear that way, efficiencies can be made in other ways that can make up for this perceived block.