A1 Q4 Name: Leila Erbay Student Id: 260672158

IterativeAddition

Number of Digits	Average Runtime	Scientific Notation of Ave. RunTime
2	39030	3.9E+04
4	1128642	1.12E+06
5	10108509	1.01E+07
6	110852982	1.12E+08
7	1148761511	1.15E+09
8	8606655845	8.61E+09
16	Too slow	--
32	Too slow	--
64	Too slow	--
128	Too slow	--
256	Too slow	--
512	Too slow	--
1024	Too slow	--
2048	Too slow	--
4096	Too slow	--

Standard Multiplication

Number of Digits	Rounded Average Runtime	Scientific Notation of Rounded Ave. RunTime
2	9239	9.24E+03
4	13233	1.32E+04
8	21797	2.18E+04
16	46992	4.70E+04
32	86945	8.69E+04
64	183099	1.83E+04
128	354977	3.55E+05
256	992432	9.92E+05
512	3562104	3.56E+06
1024	13793264	1.38E+07
2048	55776706	5.58E+07

4096	206100840	2.06E+08

RecursiveMultiplication

Number of Digits	Rounded Average Runtime	Scientific Notation of Rounded Ave. RunTime
2	16326	1.63E+04
4	25607	2.56E+04
8	65835	6.58E+04
16	136405	1.36E+05
32	385287	3.85E+05
64	1387560	1.39E+06
128	4262892	4.26E+06
256	15517394	1.55E+07
512	65900383	6.59E+07
1024	246315052	2.46E+08
2048	1027812940	1.03E+09
4096	4336164619	4.34E+09

RecursiveFastMultiplication

Number of Digits	Rounded Average Runtime	Scientific Notation of Rounded Ave. RunTime
2	17774	1.78E+04
4	40538	4.05E+04
8	79790	7.98E+04
16	165432	1.65E+05
32	378756	3.79E+05
64	960587	9.61E+05
128	2436025	2.43E+06
256	6762818	6.76E+06
512	19825932	1.98E+07
1024	58862835	5.89E+07
2048	198962684	1.99E+08
4096	554153896	5.54E+08

b) Prediction of runtime for each method for multiplying two-8192 digits

	Iterative Addition	*Standard Multiplication*	*Recursive Multiplication*	*Recursive Fast Multiplication*
Nanoseconds	2.92E+16	2.74E+14	5.71E+15	7.44E+14
Seconds	29,180,313.6	274,307.3178	5,707,433.574	744,438.1696

c) Formulas for each method:

Iterative Addition:
[Nanoseconds] T(n) = (4.35E+8)n^2 -(1.47E9)n
[Seconds] T(n) =.435n^2 -1.47n

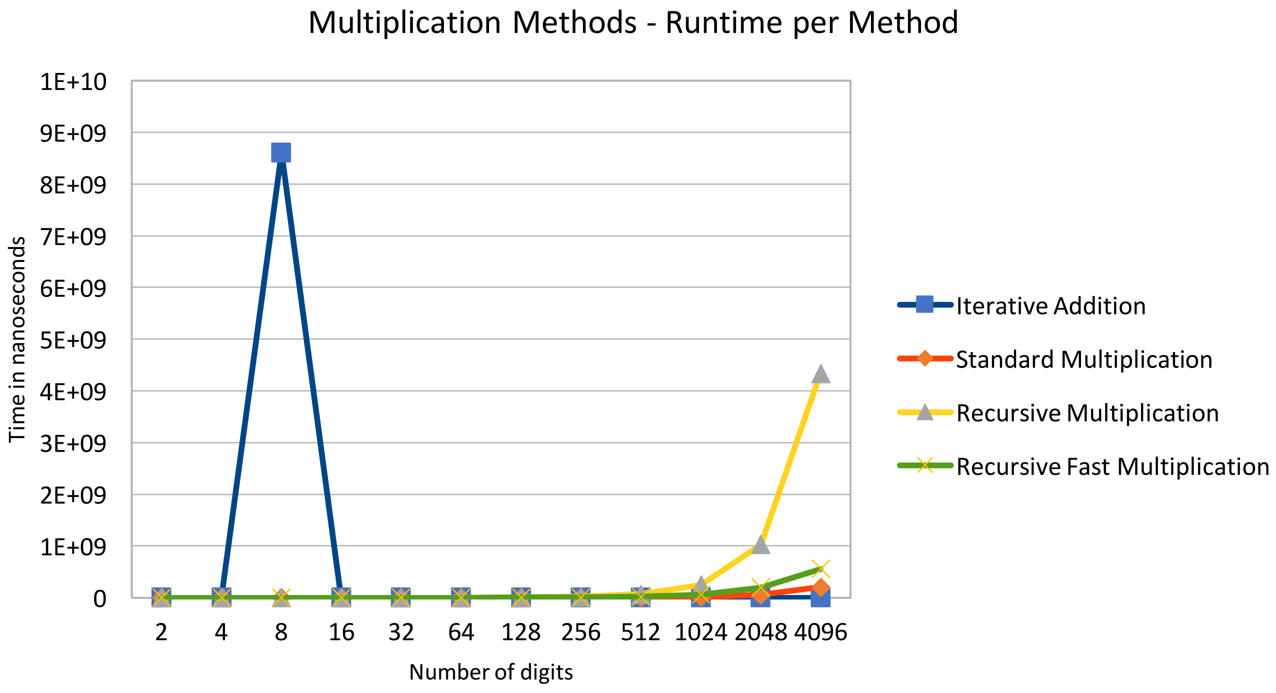
Standard Multiplication
[Nanoseconds] T(n)= (4.09E6)n^2 (2.05E7)n
[Seconds] T(n) = 0.00409n^2 0.0205n

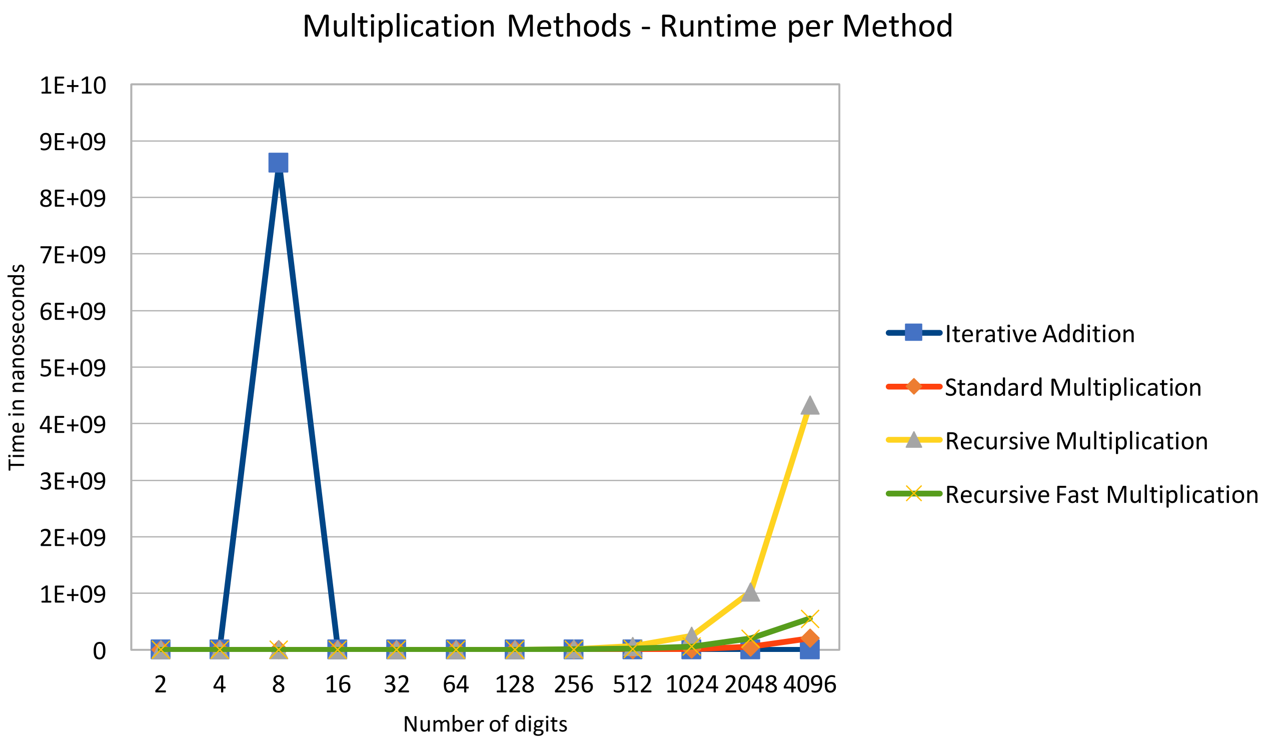
Recursive Multiplication
[Nanoseconds] T(n) = (8.51E7)n^2 (4.31E8)n
[Seconds] T(n) = 0.0851n^2 -0.431n

Recursive Fast Multiplication:
[Nanoseconds] T(n) = (1.11E7)n^2 5.45E7n
[Seconds] T(n) = 0.0111n^2 -0.0545n

d) When does Recursive Fast Multiplication become faster than the other three methods?
From the data I have collected, my recursiveFastMultiplication is always slower than my standardMultiplication; recursiveFastMultiplication is always faster than iterativeAddition; recursiveFastMultiplication becomes faster than recursiveMultiplication between 30-31 digits. Although it would seem intuitive for my recursiveFastMultiplication to become faster than my standardMultiplication, there are many factors within the code that will affect the speed of each method and having searched for explanations, I’ve learned that loops can often be faster than recursive methods. This shows that recursive methods are not necessarily faster than simpler methods.

Tables and charts of runtime for each method





Multiplication Methods - Runtime per Method