Name: Jorge Aranda

(Use this page as the cover sheet when you turn in your project report)

Project #2
CS3310 Design and Analysis of Algorithms

1. (40 points)	Date Sets, Test Strategies & Results	
2. (10 points)	Theoretical Complexity Comparisons	
3. (10 points)	Select 2 Versus Select 3	
4. (10 points)	Select 4 Versus Select 1	
5. (15 points)	Strength and Constraints of Your Work (at least 150 words)	
6. (15 points)	Program Correctness	
(100 points)	**Total**	

1: 40 points
2: 10 points
3: 10 points
4: 10 points
5: 15 points
6: 15 points

Total: 100 points

100

Test Strategies

My program works by first asking the user to input a number 'n' for the size of the array. My program then creates an array with random numbers from 1 – 1000. The numbers are always random and thus each test case should be different. Once the array is created, my program then runs it through a method that would conduct each algorithm. Each call creates a copy of the given array so that it does not alter it in any way for the other algorithms. Right before and after each method is called, I set two timers to record the time taken for the code to be executed. One is recorded in nanoseconds and the other is recorded in milliseconds. However, since I recorded so much data with each k value, I only recorded nanoseconds. Each algorithm is executed with the same exact array so that it ensures each algorithm is being tested with the same results.

My program can continue running until the user enters -1. Once the user enters this sentinel value, the program finishes running. To ensure that each test case runs with the maximum amount of processing power, I closed every other program except excel. I kept excel open in order to input my values after each test case. Each test case in the following data tables was conducted as follows: First, I would run the program. After the program starts to run, I enter 10 so that my program creates an array of size 10 and then executes each algorithm on them. It records each time in nanoseconds and milliseconds for k = 1, n/4, n/2, 3n/4, and n. Once it finishes executing, I enter 10, 50, 100, 250, 1000, and so on until I got to 10000000. My computer was not able to execute the code after 10000000, it would take long to execute as well as I would receive a stack overflow error for some of my algorithms. After I reached 10000000, I stopped the program and recorded my data. I would then restart the program and repeat the process for the other test cases. I conducted a total of 10 test cases and recorded my data in excel. I have attached the excel document along with this report. The excel document contains all the test cases as well as a graph for all the average times of each n case. To get this average, I added all the k's for that given n and then divided it by 5 (for all the values for k).

Unfortunately, the algorithm I used to get my Quick Sort using the MM rule working was only able to work for k = 1, n/4, and n/2. Once it attempted to do the algorithm on 3n/4 or n, it would cause a stack overflow error. It would sometime work for 3n/4, but it was not reliable. So, I was only able to conduct test cases up to k = n/2 for my Quick Sort using the MM rule algorithm.

The following contains my data sets for each algorithm, a graph for the averages of the n test cases, as well as averages for nanoseconds on each test case.

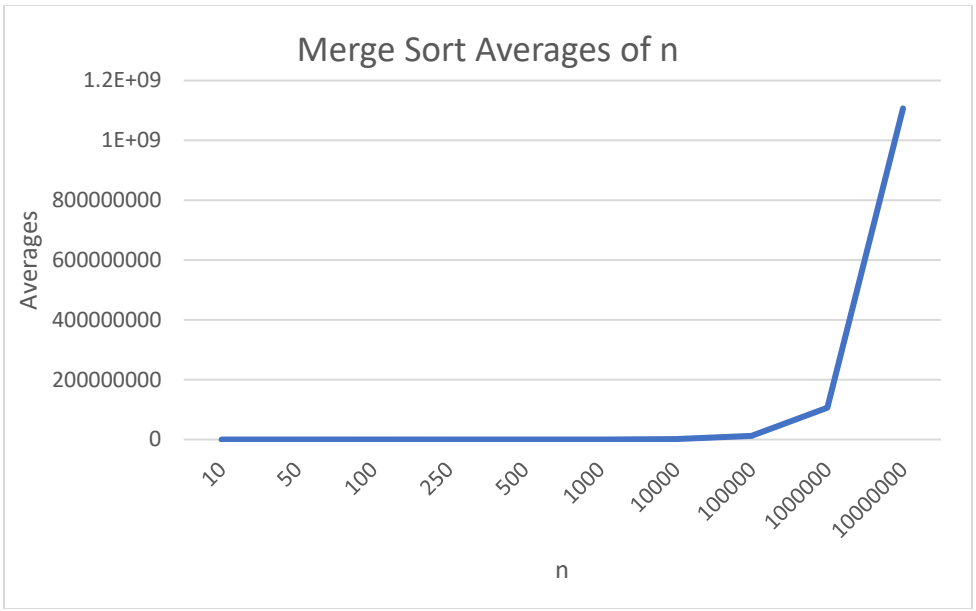
Merge Sort Algorithm Data Set

n	10					50					100					250					500				
Test Case																									
	k=1	k=n/4	k=n/2	k=3n/4	k=n	k=1	k=n/4	k=n/2	k=3n/4	k=n	k=1	k=n/4	k=n/2	k=3n/4	k=n	k=1	k=n/4	k=n/2	k=3n/4	k=n	k=1	k=n/4	k=n/2	k=3n/4	k=n
1	47900	17500	14900	16200	14300	35300	35601	30800	28201	32701	29901	17799	15799	15200	15600	45100	34701	38500	28901	25199	64700	59700	51500	51099	50600
2	50800	17600	14500	15700	14100	35600	49300	36300	32400	34100	31900	18200	16500	16500	15500	42400	44000	31900	31100	33600	64200	60700	49600	51800	48200
3	50300	26500	15900	16400	14200	37600	37000	30800	31400	33100	30200	17900	16000	16700	14300	43800	42500	29600	28700	32200	66200	60400	52700	51700	50600
4	46800	24000	16000	16300	14200	37400	35810	32800	26401	32801	28900	17899	18700	18000	15900	46900	34701	37500	29800	26200	65800	58000	56500	52300	50600
5	50200	17400	14800	15200	15000	35400	48300	36900	31800	38900	32600	18700	17500	17500	14800	45000	48900	32900	32100	34000	63200	60900	49000	58900	49200
6	51200	17800	14500	15320	13900	37600	37000	30800	31400	32000	30400	17900	15000	16600	14300	43800	43500	28700	31900	33500	66200	62400	53000	56000	56600
7	68540	17500	14850	15700	14300	35600	35600	30800	29300	32600	29901	16700	15800	15200	15200	46100	39401	36500	29800	25309	68000	58700	52500	52600	52600
8	50120	23000	15000	14900	14500	32500	47100	35300	32600	34100	30970	18700	16900	18500	14500	42900	46000	37900	32300	33600	65200	64700	50600	51800	42200
9	48600	21850	16500	14800	15000	34200	39000	32800	35400	33200	32400	18100	15400	17500	13100	44800	44800	29780	29400	35000	69200	62500	49700	58700	52600
10	54120	23000	14800	15600	13500	33000	36050	30900	33000	31000	33500	17800	17900	16800	14800	43900	48000	32000	24800	28000	62800	59800	50600	54000	49500
Total	518580	206150	151750	156120	143000	354200	400761	328200	311902	334502	310672	179698	165499	168500	148000	444700	426503	335280	298801	306608	655500	607800	515700	538899	502700
Average	51858	20615	15175	15612	14300	35420	40076.1	32820	31190.2	33450.2	31067.2	17969.8	16549.9	16850	14800	44470	42650.3	33528	29880.1	30660.8	65550	60780	51570	53889.9	50270

n	10	50	100	250	500	1000	10000	100000	1000000	10000000
Total Average of n	117560	172956.5	97236.9	181189.2	282059.9	521255.8	7779579.4	61355840	531189564	5534325536

1000					10000					100000					1000000					10000000				
k=1	k=n/4	k=n/2	k=3n/4	k=n	k=1	k=n/4	k=n/2	k=3n/4	k=n	k=1	k=n/4	k=n/2	k=3n/4	k=n	k=1	k=n/4	k=n/2	k=3n/4	k=n	k=1	k=n/4	k=n/2	k=3n/4	k=n
115899	102500	98700	98500	99001	1151299	1122199	1153000	1131600	1224100	12594100	12297200	12492100	12509799	12281300	103730601	101337500	1.15E+08	100372000	106490000	1124532101	1119540400	1.231E+09	1039825001	1053161499
117000	100500	104400	99000	101400	1.1E+07	1118300	1137400	1197400	1192300	12495200	12307600	12535100	12381200	12646800	104114500	104925200	1.1E+08	101010700	109443500	1121476700	1094280100	1.197E+09	1015947200	1030596600
115600	100100	98400	98400	104200	1139800	1122000	1120900	1125500	1138300	12749800	12382500	12640300	12367200	12575900	102081300	110111300	1.09E+08	101810300	114081900	1128471200	1137481200	1.217E+09	1035663500	1048679710
117899	103500	97600	99500	99200	1151299	1123199	1152000	1131600	1234100	12694100	12296200	12392100	12549799	12381300	103830601	101437500	1.15E+08	100472000	105490000	1123532101	1119543400	1.231E+09	1034825001	1052161499
118000	102500	94600	98500	101600	1.1E+07	1138300	1138400	1147400	1194300	12498200	12407600	12555100	12371200	12446800	104614500	104945200	1.1E+08	101210700	109653500	1126476700	1094580100	1.197E+09	1015647200	1036596600
116200	101100	97400	98400	120200	1139800	1132000	1160900	1123500	1128300	12849800	12682500	12630300	12467200	12475900	102381300	110121300	1.05E+08	101410300	114881900	1122471200	1137681200	1.217E+09	1034663500	1048879710
118899	102300	97900	98300	99000	1151299	1128199	1153400	1121600	1234100	12593100	12397200	12472100	12409799	12381300	103830601	102337500	1.17E+08	100772000	106990000	1124632101	1119640400	1.231E+09	1036825001	1053361499
119000	108500	104600	99200	101400	1145800	1118300	1137900	1187400	1182300	12445200	12407600	12535100	12371200	12446800	104214500	103925200	1.09E+08	101210700	109843500	1123476700	1094980100	1.197E+09	1015947200	1030296600
115600	103100	98600	98400	103200	1139800	1132000	1140900	1125500	1148300	12759800	12482500	12440300	12347200	12585900	103081300	110111300	1.06E+08	101810300	113081900	1125471200	1137381200	1.217E+09	1035663500	1047679710
113560	105000	99800	98400	108000	1139600	1185600	1138000	1128500	1186500	1278900	12387000	12865300	12465400	12532500	103032140	110211300	1.05E+08	103565100	113251200	1132651020	1136580010	1.216E+09	1035626200	1045665200
1167657	1029100	992000	986600	1037201	3.2E+07	1.1E+07	1.1E+07	1.1E+07	1.2E+07	114958200	1.24E+08	125557800	124239997	1.25E+08	1.035E+09	1.059E+09	1.1E+09	1013644100	1103207400	1.1253E+10	1.1192E+10	1.215E+10	10300633303	1.0447E+10
116766	102910	99200	98660	103720	3176030	1132010	1143280	1142000	1186260	11495820	12404790	12555780	12424000	12475450	103491134	105946330	1.1E+08	101364410	110320740	1125319102	1119168811	1.215E+09	1030063330	1044707863

n	10	50	100	250	500	1000	10000	100000	1000000	10000000
Total Average of n	117560	172956.5	97236.9	181189.2	282059.9	521255.8	7779579.4	61355840	531189564	5534325536
Average of n	23512	34591.3	19447.38	36237.84	56411.98	104251.16	1555915.88	12271168	106237913	1106865107

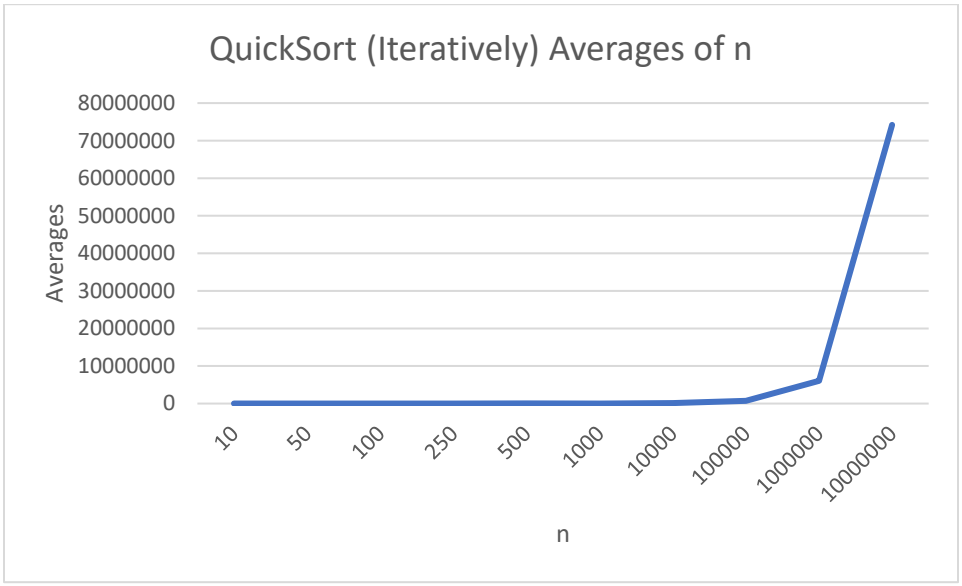


Quick sort (Iteratively) Algorithm Data Set

n	10					50					100					250					500					1000		
Test Case																											
	k=1	k=n/4	k=n/2	k=3n/4	k=n	k=1	k=n/4	k=n/2	k=3n/4	k=n	k=1	k=n/4	k=n/2	k=3n/4	k=n	k=1	k=n/4	k=n/2	k=3n/4	k=n	k=1	k=n/4	k=n/2	k=3n/4	k=n	k=1	k=n/4
1	18800	10899	9600	10400	10401	12301	12101	13001	11700	1800	18500	26300	26301	16000	13600	20500	27701	27500	20500	22400	19100	31600	26800	32801	36300	11100	17400
2	19600	11000	12600	10500	9800	11000	11500	15600	14600	20000	16400	15700	14900	16600	12400	11900	28200	27500	27600	22200	17500	39200	33600	38600	23000	11300	20300
3	20900	11600	11100	10100	10100	12400	13800	13100	12300	17700	11100	19700	17800	16300	16100	27000	25500	25000	26200	16200	31600	42100	44100	34700	26700	16200	20500
4	19800	11799	9900	10300	10401	13301	13201	13201	11800	19000	18500	25300	23301	16200	14600	20500	26701	29600	20400	23500	19300	38200	26900	32601	36300	18200	17400
5	19700	11000	11600	10500	10800	12000	12500	15650	13600	21000	15400	16700	16900	16400	13400	12900	23200	26500	26500	23200	27500	38900	33600	37600	23000	11200	20300
6	21000	11400	11100	10200	10100	12300	12800	12100	13300	17600	11320	13700	16800	16300	14100	27600	25400	24000	20500	18200	31800	31600	45100	34700	26700	11300	20600
7	19800	10899	9400	10400	10401	12401	11101	14001	12700	17000	17500	22300	24201	16200	13600	22500	27801	26500	27600	23400	19300	39200	26800	34600	32500	16200	23500
8	19600	11000	12800	10600	9900	12000	12500	15600	13600	20200	16200	19700	16900	15600	12400	12900	22200	27400	26200	22200	17400	42100	33800	32801	36300	13100	17400
9	20600	11300	12100	10100	10100	12300	12800	14100	12300	16700	11100	16700	18800	16300	16200	26500	24500	25600	25300	20200	31600	32400	43100	38400	22000	11300	20400
10	20400	10900	12000	10200	9600	13000	13650	15500	14560	2000	12300	24030	19800	16450	12600	26000	25630	25400	25700	19600	28600	34600	35620	34700	26700	16200	20500
Total	200200	111797	112200	103300	101603	123003	125953	141853	130460	153000	148320	200130	195703	162350	139000	208300	256833	265000	246500	211100	243700	369900	349420	351503	289500	136100	198300
Average	20020	11179.7	11220	10330	10160.3	12300.3	12595.3	14185.3	13046	15300	14832	20013	19570.3	16235	13900	20830	25683.3	26500	24650	21110	24370	36990	34942	35150.3	28950	13610	19830

1000					10000					100000					1000000					10000000				
k=1	k=n/4	k=n/2	k=3n/4	k=n	k=1	k=n/4	k=n/2	k=3n/4	k=n	k=1	k=n/4	k=n/2	k=3n/4	k=n	k=1	k=n/4	k=n/2	k=3n/4	k=n	k=1	k=n/4	k=n/2	k=3n/4	k=n
11100	17400	22101	16201	14900	55800	238100	156700	171700	64100	993600	832100	921601	466900	308901	2477300	9092300	8675400	8906500	1490300	95983100	124387700	57644701	67159599	23717800
11300	20300	26400	24400	16400	106600	133900	108400	116000	117000	511700	476200	844200	766600	598500	6363900	6320200	8160200	4239700	1472500	96654700	113775400	95109800	77313300	29650000
16200	20500	18400	21200	15000	155500	165700	144000	134500	55900	692400	1096800	773600	654700	480600	4609100	5216500	7599000	7818200	4736600	100925600	82814200	69309500	36220600	23522500
18200	17400	22101	16201	14900	105800	228100	159600	161700	98600	762300	1032100	921601	466900	308901	2477300	9093300	8595400	6567000	3490300	100983100	124387700	57644701	57313300	23717800
11200	20300	26400	24300	16400	107600	233900	156700	117000	65100	983600	676200	843200	776600	588500	6463900	6322200	8465200	8906500	1472500	99654700	123775400	95109800	67159599	29650000
11300	20600	17400	21200	16000	155500	167700	108400	134500	107000	512700	1096800	773600	664700	489600	4609100	7216500	8675400	4249700	4736600	100925600	92814200	69309500	67313300	23522500
16200	23500	26000	23800	14900	165550	210030	144000	171700	65900	793400	832100	796000	466900	308901	2677300	9092300	8160200	7818200	2490300	98983100	124387700	57644701	36220600	29450000
13100	17400	22101	16201	15400	85800	238100	108400	126000	64100	993600	776200	921601	768600	588500	6363900	8320200	7599000	8966500	1472500	94654700	123775400	95109800	57313300	22717800
11300	20400	26300	22400	15000	106600	134900	144000	134500	107000	521700	1096800	844200	644700	480600	4709100	6216500	8260200	4239700	4736600	100925600	82814200	69309500	67313300	29450000
16200	20500	18700	21200	16000	156500	165700	146500	135800	106900	692400	963200	773600	726300	308901	6467500	7816500	7909000	7828200	2350400	102924600	93414200	95109800	77313300	23422500
136100	198300	225903	207103	154900	1201250	1916130	1376700	1403400	851600	7457400	8878500	8413203	6402900	4461904	47218400	74706500	82099000	69540200	28448600	992614800	1086346100	7.61E+08	6.11E+08	2.59E+08
13610	19830	22590.3	20710.3	15490	120125	191613	137670	140340	85160	745740	887850	841320.3	640290	446190.4	4721840	7470650	8209900	6954020	2844860	99261480	108634610	76130180	61064020	25882090

n	10	50	100	250	500	1000	10000	100000	1000000	10000000
Total of n	62910	67426.9	84550.3	118773.3	160402.3	92230.6	674908	3561391	30201270	370972380.1
Average of n	12582	13485.38	16910.06	23754.66	32080.46	18446.12	134981.6	712278.1	6040254	74194476.02

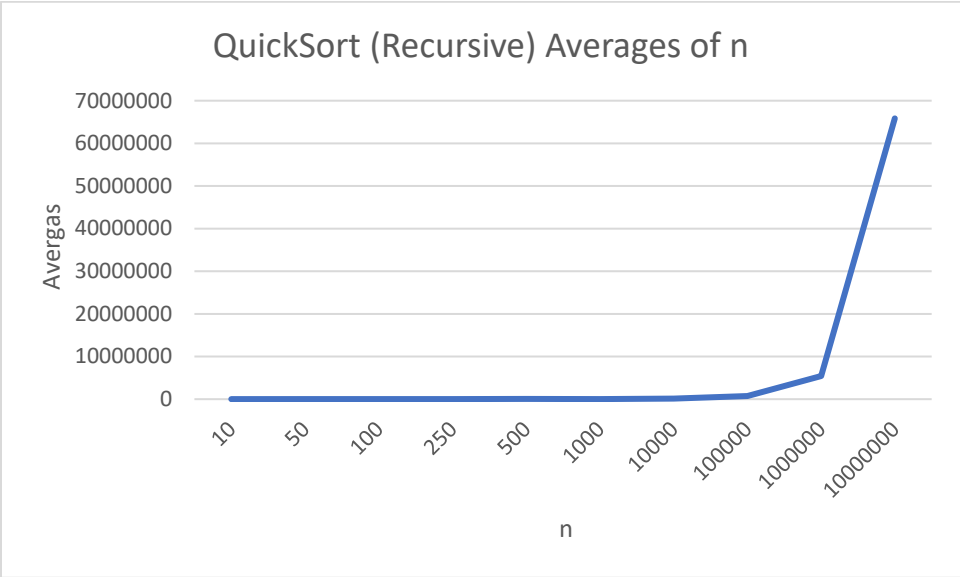


Quick Sort (Recursive) Data Set

n	10					50					100					250					500					1000		
Test Case																											
	k = 1	k = n/4	k = n/2	k = 3n/4	k = n	k = 1	k = n/4	k = n/2	k = 3n/4	k = n	k = 1	k = n/4	k = n/2	k = 3n/4	k = n	k = 1	k = n/4	k = n/2	k = 3n/4	k = n	k = 1	k = n/4	k = n/2	k = 3n/4	k = n	k = 1	k = n/4
1	12700	9800	9600	9799	9901	12801	12300	12600	11500	10899	17499	18800	17200	12300	11300	16501	21600	23300	22400	13899	20601	27200	38400	46800	34301	11801	16599
2	13000	12000	11600	10300	9300	10800	11200	14200	14100	12600	14700	15300	15200	15900	22800	11100	26900	25700	23000	21700	14200	36300	35800	43900	32600	10500	13100
3	13600	12100	11500	10000	10200	13000	13300	11900	11300	10400	12000	16100	16200	12400	12100	25300	23600	25200	22800	14300	48100	47100	98300	74100	9200	14200	21900
4	13700	11300	12000	11000	9801	11801	13300	12700	11500	11350	17390	18800	17400	12400	12300	14501	21400	22300	22700	1899	20501	28200	76500	56800	34301	11601	17500
5	13000	12800	11600	9899	10300	12800	12200	14200	13100	11899	14200	15400	15200	15600	21800	13100	26800	23700	23200	20700	14300	34300	38600	43900	32600	11500	16600
6	12600	12000	11500	10300	10200	12000	12300	11800	12300	12600	13000	16400	16200	13400	12100	24300	23600	24200	23800	15300	48100	42100	34800	65100	12200	12200	14100
7	13700	12100	9900	10200	10901	12801	12300	12600	12500	11400	16499	17650	16200	13300	18650	16501	21400	23300	23400	18899	20601	37200	98300	46800	23200	12801	18900
8	13000	9900	11600	9990	9300	11800	11200	13200	14100	12899	14700	17800	16200	14900	11300	18100	24900	26700	23000	21700	34200	36300	37400	53900	34301	12500	16599
9	12800	12000	11400	10300	10200	13000	12300	12900	13300	12600	13000	16300	17200	13500	22800	23300	23600	25200	22600	17300	48100	37100	32800	74100	32600	13200	17100
10	12900	12100	11800	10000	9600	12900	12600	13400	13200	12400	13500	16800	16200	12800	13100	24600	22600	23600	23000	19600	38600	29000	88300	67900	10200	12650	21800
Total	131000	116100	112500	101788	99703	123703	123000	129500	126900	119047	146488	169350	163200	136500	158250	187303	236400	243200	229900	165297	307303	354800	579200	573300	255503	122953	174198
Average	13100	11610	11250	10178.8	9970.3	12370.3	12300	12950	12690	11904.7	14648.8	16935	16320	13650	15825	18730.3	23640	24320	22990	16529.7	30730.3	35480	57920	57330	25550.3	12295.3	17419.8

1000					10000					100000					1000000					10000000				
k = 1	k = n/4	k = n/2	k = 3n/4	k = n	k = 1	k = n/4	k = n/2	k = 3n/4	k = n	k = 1	k = n/4	k = n/2	k = 3n/4	k = n	k = 1	k = n/4	k = n/2	k = 3n/4	k = n	k = 1	k = n/4	k = n/2	k = 3n/4	k = n
11801	16599	24900	21600	17200	5300	219399	161700	173000	100400	1306900	734400	765800	392901	340100	1602001	7658001	8632201	7394600	1143699	56760600	81889501	74730100	73575700	45387099
10500	13100	23300	20200	14100	97700	140900	105100	106800	82200	511700	624000	910400	727000	626000	5842100	5731400	8288400	3720800	1462000	66330800	85744300	70040400	54574300	34841200
14200	21900	25900	18200	10200	123600	151600	127900	122200	52600	618800	733600	722300	769600	420900	4025500	4906300	6638900	7746000	3110300	68719700	94686400	85229900	51148900	41618400
11601	17500	24300	19800	17200	9300	229399	171700	163000	100400	1206900	754400	745800	592901	440100	6602001	7458001	7632201	7394600	2143699	55760600	81889501	74730100	73575700	45347099
11500	16600	22300	21600	16100	117700	130900	125100	126800	102200	1111700	634000	890400	727000	526000	5842100	5631400	7288400	4720800	2462000	66330800	85744300	70240400	54674300	34841200
12200	14100	26900	20200	11200	123600	141600	147900	142200	112600	1218800	733600	822300	669606	520900	4025500	5006300	6638900	7646000	3110300	67719700	94686400	85229900	51148900	41618400
12801	18900	24900	19200	17200	123300	219399	151700	133000	100400	1306900	724400	765800	592901	440100	3602001	7658001	8632201	7394600	3143699	56760600	81889501	74730100	73575700	45387099
12500	16599	23300	21600	13100	99700	190900	115100	106800	92200	1211700	624000	910400	727000	626000	5842100	5631400	6888400	5720800	2462000	65330800	85744300	70040400	54574300	34841200
13200	17100	25900	21200	11200	123600	151600	137900	122200	102600	619800	713600	722300	769600	620900	4025500	4906300	6638900	7746000	3110300	64719700	94682400	85229900	51148900	41718400
12650	21800	24500	18200	12500	131560	203500	146500	165400	76500	865000	689500	765200	656500	596500	4659000	5213500	7986500	6759220	1462000	67330800	81889501	74730100	73775700	45367099
122953	174198	246200	201800	140000	955360	1779197	1390600	1361400	922100	9978200	6965500	8020700	6625009	5157500	46067803	59800603	75265003	66243420	23609997	6.36E+08	868846104	7.65E+08	6.12E+08	4.11E+08
12295.3	17419.8	24620	20180	14000	95536	177919.7	139060	136140	92210	997820	696550	802070	662500.9	515750	4606780	5980060	7526500	6624342	2361000	63576410	86884610	76493130	61177240	41096720

n	10	50	100	250	500	1000	10000	100000	1000000	10000000
Total of n	56109.1	62215	77378.8	106210	207010.6	88515.1	640865.7	3674691	27098683	329228110
Average of n	11221.82	12443	15475.76	21242	41402.12	17703.02	128173.1	734938.2	5419737	65845622

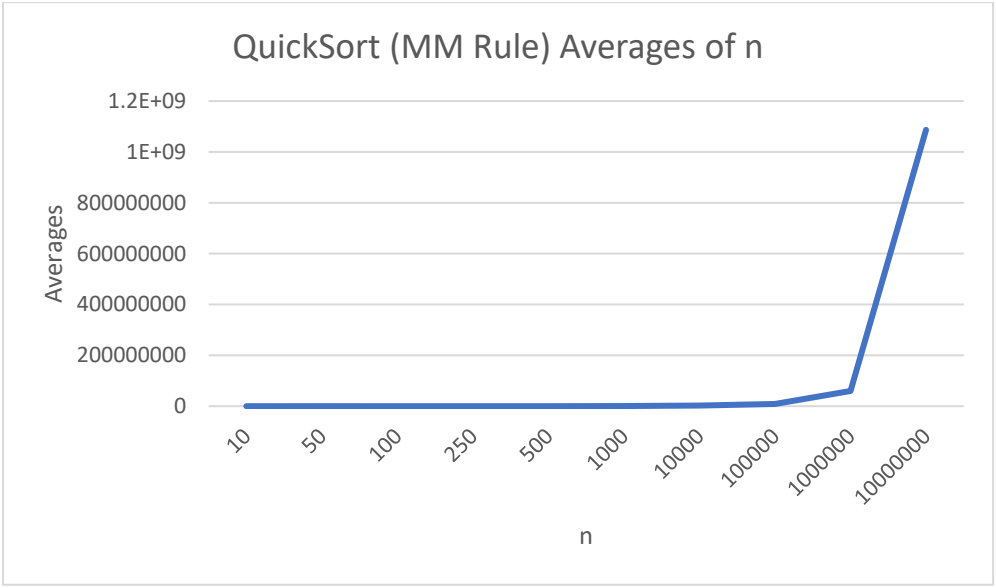


Quick Sort (MM Rule) Data Set

n	10			50			100			250			500			1000			10000		
Test Case																					
	k=1	k=n/4	k=n/2	k=1	k=n/4	k=n/2	k=1	k=n/4	k=n/2	k=1	k=n/4	k=n/2	k=1	k=n/4	k=n/2	k=1	k=n/4	k=n/2	k=1	k=n/4	k=n/2
1	307999	17601	16400	52100	43799	36200	85500	133900	54800	91300	104600	67299	144400	111400	215999	298600	135600	2220601	2110200	550800	569800
2	311400	20500	18100	64500	44000	32300	81400	137500	58900	94600	113500	79400	119300	180700	140300	298700	2115000	115200	3273600	2259300	1745300
3	361800	18000	16400	55300	42200	33800	73900	125900	59200	90400	99600	61800	200800	96700	175100	317900	337900	2092900	6042200	2705300	2776500
4	317999	19601	17400	54100	44799	35200	91500	123900	56800	92300	114600	77299	134400	121400	203999	308600	235600	2220601	2310200	1050800	569800
5	321400	20500	17100	62500	42000	33300	71400	127500	57900	93600	113500	79400	129300	140700	180300	308700	2015000	125200	3173600	2259300	1745300
6	341800	19000	18400	61300	43200	33800	83900	125900	58200	92400	109600	61800	190800	106700	195100	307900	337900	2092900	5042200	2605300	569300
7	327999	18601	15400	52100	43799	32200	85500	133800	55800	92300	104600	67299	144400	111400	205999	318600	435600	2100601	2110200	1040800	579800
8	321400	20500	18100	64500	43000	32300	71400	137500	57900	93600	113500	79400	169300	180700	150300	308700	1115000	135200	3273600	2259300	1745300
9	361800	19000	16400	54300	41200	34800	73900	135900	59200	91400	99600	61800	200800	96700	165100	317900	337900	2092900	6032200	2405300	2776500
10	341800	20350	17600	64500	45000	33600	91000	137000	58700	92400	108500	72000	155600	107800	178600	298900	332790	145600	3073000	550900	569800
Total	3315397	193653	171300	585200	432997	337500	809400	1318800	577400	924300	1081600	707497	1589100	1254200	1810797	3084500	7398290	13341703	36441000	17687100	13647400
Average	331539.7	19365.3	17130	58520	43299.7	33750	80940	131880	57740	92430	108160	70749.7	158910	125420	181079.7	308450	739829	1334170.3	3644100	1768710	1364740

10000			100000			1000000			1000000		
k = 1	k = n/4	k = n/2	k = 1	k = n/4	k = n/2	k = 1	k = n/4	k = n/2	k = 1	k = n/4	k = n/2
2110200	550800	569800	23281099	4112300	4124101	105705100	43039499	32018900	1250786800	687288599	328306299
3273600	2259300	1745300	25564400	4049700	7944800	108087900	36617100	38846900	2500008000	701023000	1724383700
6042200	2705300	2776500	8997900	3182700	3082500	104465600	37626300	32317500	1328048800	399878400	365993100
2310200	1050800	569800	21281099	4122300	7124101	105705100	42039499	34018900	1240786800	697288599	328306299
3173600	2259300	1745300	24564400	4029700	7644800	107087900	37617100	37846900	2400008000	711023000	1724383700
5042200	2605300	569300	9997900	3162700	3182500	103465600	37626300	32317500	1428048800	499878400	365993100
2110200	1040800	579800	21281099	4212300	3124101	105705100	43039499	33018900	1250786800	587288599	323406299
3273600	2259300	1745300	24564400	4048700	7944800	106087900	38617100	38846900	2600008000	601023000	1724383700
6032200	2405300	2776500	10997900	3202700	3082500	105465600	37626300	34317500	1228048800	499878400	335993100
3073000	550900	569800	11670600	4129700	4135020	107082030	39624500	32018900	2309980040	712023000	1746383700
36441000	17687100	13647400	182200797	38252800	51389223	1058857830	393473197	345568800	17536510840	6096592997	8967532997
3644100	1768710	1364740	18220079.7	3825280	5138922.3	105885783	39347319.7	34556880	1753651084	609659300	896753300

n	10	50	100	250	500	1000	10000	100000	1000000	10000000
Total Average	368035	135569.7	270560	271339.7	465409.7	2382449	6777550	27184282	179789983	3260063683
Average of n	122678.33	45189.9	90186.67	90446.57	155136.6	794149.8	2259183	9061427.3	59929994.2	1086687894



Merge Sort Data Set Explanation

Merge Sort is said to have a time complexity of $O(nlogn)$. Through the data I collected, Merge Sort seemed to have performed the worst out of all the algorithms. I believe this to be because it sorts the whole array and then returns k after it sorts the array. While a simple and straightforward approach, it is not the best algorithm to return k. Surprisingly, according to the data, it seemed to have done better with k = n/4, n/2, and 3n/4 as opposed to k = 1 and n. Not sure why that was the case, since it should have just returned an index after it sorted the data.

Quick Sort (Iteratively) Algorithm Data Set Explanation

Quick Sort is said to have a best time complexity of $O(n)$ and worse time complexity of $O(n^2)$. You can somewhat notice this best and worse time complexity in the data tables. For each k in the different n sized arrays, it was random as to when it would perform the quickest. Sometimes it would perform really quickly, while at other times it seemed to have doubled the time of other k values in the same n sized array. It seemed to have performed similar to Quick Sort Recursively.

Quick Sort (Recursive) Algorithm Data Set Explanation

Quick Sort Recursively is said to have a time complexity similar to Quick Sort Iteratively. $O(n)$ best case complexity and $O(n^2)$ as worse case complexity. This, much like the Iterative case, can be noticed through the data tables. For each k in the different sized arrays, it was random as to when it would perform the quickest. It is worth noting that this algorithm seemed to have the lowest average for the biggest array of 10000000. Surprisingly, this algorithm seemed to have outperformed all other algorithms. However, it only outperformed Quick Sort Iteratively by a small amount.

Quick Sort (MM Rule) Algorithm Data Set Explanation

Quick Sort using the Median of Medians rule is said to have a worse case time complexity of $O(n)$. This can somewhat be seen in the data set. It did seem to stay consistent the most with having little increase in average time after n = 100. However, it surprisingly did not perform better than both Quick Sort Recursively or Iteratively. Quick Sort with the Median of Medians rule did eventually perform better than Merge Sort. Out of all the algorithms at n = 10000000 (the highest sized array I was able to get), it performed 3rd best. It is surprising that this algorithm did not have the best time complexity compared to the other Quick Sort algorithms.

Theoretical Complexity Comparisons and Conclusion

As mentioned previously: Merge Sort has a time complexity of $O(nlogn)$, Quick Sort both recursively and iteratively has a best time complexity of $O(n)$ and a worse time complexity of $O(n^2)$, while Quick Sort using the Median of Medians rule has a worse case time complexity of $O(n)$. Theoretically, the Quick Sort using the Median of Medians rule is supposed to perform better than all the other algorithms. However, through my test cases, I was not able to come to that conclusion. Quick Sort iteratively and recursively both performed better than with using the Median of Medians rule. Perhaps this is because time was spent figuring out what was the median of medians. Nevertheless, theoretically, I am sure that once we get to bigger sized arrays, Quick Sort using the Median of Medians rule should start to outperform the other algorithms.

Through the test cases that I performed, Quick Sort using the Median of Medians rule started to outperform Merge Sort at n = 100000. Before then, Quick Sort using the Median of Medians definitely took a lot longer to process compared to Merge Sort. However, Merge Sort began growing in average time much faster than Quick Sort using MM did. Quick Sort using MM grew much more consistently than Merge Sort. Looking at the averages for both Quick Sort recursively and iteratively, they were both very similar. Quick Sort recursively, however, was able to reach a lower average at n = 50, 100, 1000, and 1000000. So, Quick Sort iteratively was not always faster than Quick Sort Recursively.

The strengths of my work are that I did a lot of comparisons. With a total of 10 test cases, I was able to bring a fairly accurate average for all the cases for each algorithm. Within those 10 cases, I took the average of all the different k's. Since there was so many different k's, it took much longer to collect the data. However, doing it this way instead of taking the average right away allowed me to look at how each algorithm performed at each k. With this many cases I was also able to give decent graphs for the data. The constraints of my work have to do with both my computer as well as my Quick Sort using the MM rule. My computer was not able to handle test cases after 10000000. I think if it could, I could have gathered much more detailed results. My algorithm for Quick Sort using the MM rule was also another constraint. The way I implemented it, I was not able to test the algorithm for k = 3n/4 and n. This probably skewed the data a bit since the other algorithms had more numbers that went into the average.