			(Describe what the project will look like when complete)	
Owner	Ideal Completion Date	Project Name	Done When...	
Leslie	11/23/2020	merge sort	there is a working file inside of the github	
Tuyetlinh	11/23/2020	insertion sort	there is a working file inside of the github	
Jacob	11/23/2020	quicksort	there is a working file inside of the github	
Tuyetlinh, Leslie, Jacob	11/25/2020	bubble sort	there is a working file inside of the github	NO LONGER NEEDED
Tuyetlinh, Leslie, Jacob	11/29/2020	radix sort	there is a working file inside of the github	
Tuyetlinh, Leslie, Jacob	12/2/2020	figure out sequence sizes	there is the ability to make graphs/visuals and these have been made	
Tuyetlinh, Leslie, Jacob	12/2/2020	types of sequences		
Tuyetlinh, Leslie, Jacob	12/3/2020	look at counter for measuring algorithm efficiency		
Tuyetlinh, Leslie, Jacob	12/4/2020	plot data for each algorithm's running times		
Tuyetlinh, Leslie, Jacob	12/5/2020	finish SCRUM worksheet	All pages in SCRUM are finalized, including data analysis sections	
All 3	12/12/2020	work on the presentation (linked)	Google Slide presentation is finalized and a mock presentation has been done	
All 3	12/12/2020	make the report (linked)	Project report has been submitted to Gradescope	

						(Highlight tasks blue when finished and moved to DONE			
Sprint	Owner	Additional Team	Priority (1-10)	Project	Done When	Task #1	Task #2	Task #3	Task #4
What week are these tasks getting done? Sprints start on Mondays.	Your Name	Anyone else who is on the team/helping	What is the weight/importance of this project? Assign tasks with Priority as well	Add project currently working on in the product backlog to this spreadsheet	Tasks are added to this sheet, marked green, and Done Statement is fulfilled from the Product Backlog	Split the project into smaller tasks and when you complete them turn them blue. You may also want to assign priority.			
Sprint #1 (11/16/20)	Tuyetlinh	Jacob, Leslie (reviewed)	very important (10)	Insertion Sort	Working program (insertionSort.cpp in IDE https://ide.cs50.io/tuyetlinhnguyen/ide under "Project" folder) **Uploaded to GitHUB**	Comment in pseudocode_+notes of algorithm	implement algorithm	compile and test for many cases to ensure successful implementation	NOTES: https://github.com/URI-CSC/212-fall-2020/blob/master/lectures/sorts.key.pdf https://www.tutorialspoint.com/data_structures_algorithms/insertion_sort_algorithm.htm https://www.tutorialspoint
Sprint #1 (11/16/20)	Leslie	Jacob, Tuyetlinh (reviewed)	very important (10)	MergeSort	Working program in IDE	Comment in pseudocode+notes of algorithm	implement algorithm	compile and test for many cases to ensure successful implementation	
Sprint #1 (11/16/20)	Jacob	Tuyetlinh, Leslie (reviewed)	very important (10)	QuickSort	Working program in IDE	Comment in pseudocode+notes of algorithm	implement algorithm	compile and test for many cases to ensure successful implementation	
Sprint #2 (11/23/20)	Tuyetlinh, Jacob, Leslie		very important (10)	Radix Sort	Working program in IDE	Comment in pseudocode+notes of algorithm	implement algorithm	compile and test for many cases to ensure successful implementation	
Sprint #3 (11/30/20)	Leslie	Tuyetlinh, Jacob	7	Presentation	set up/outlines presentation sections	divide up the speaking parts	finish each section	real world application section	
Sprint #3 (11/30/20)	Tuyetlinh		9	Sorter Class	began implementing Sorter Class to run all 4 algorithms	make it so that each file can be ran independently	be able to run each algorithm as well as an option for sequence type		
Sprint #3 (11/30/20)	Jacob	Leslie	10	Sorter Class	finished and reviewed what Tuyetlinh had started earlier this week	include the main class file	test and make sure it works with all three team members		
Sprint #3 (11/30/20)	Leslie	Jacob	5 (last item that needs to be completed)	Report/Presentation	used Sorter class to compile data and create graphs to discuss findings	make it so that when the program is run for the live coding session proper examples can be given	make a neat and organized output for all examples	sketch out what the readme is going to look like	

Sprint #4 (12/7/30)	Leslie, Tuyetlinh		5	Report/Presentation	reviewed graphs from earlier and edited trendlines and reformatted	group edits			
Sprint #4 (12/7/30)	Jacob, Tuyetlinh Leslie		10	Presentation	filled in presentation sections with appropriate content and visualizations and discussed how the presentation will be organized during class	go through each graph to discuss what the outliers are and which ones are the most efficient	include specific data values	review with the rest of the group	
Sprint #4 (12/7/30)	Jacob, Tuyetlinh Leslie		10	Report	finalized report by adding a discussion piece under graphs and a conclusion piece as well	using the content and visualizations from the presentations make the same observations in the project report	include contributions to the report	add a conclusion that connects to the real world	mention a contemporary example of how knowing the fundamentals of sorting algorithms is useful

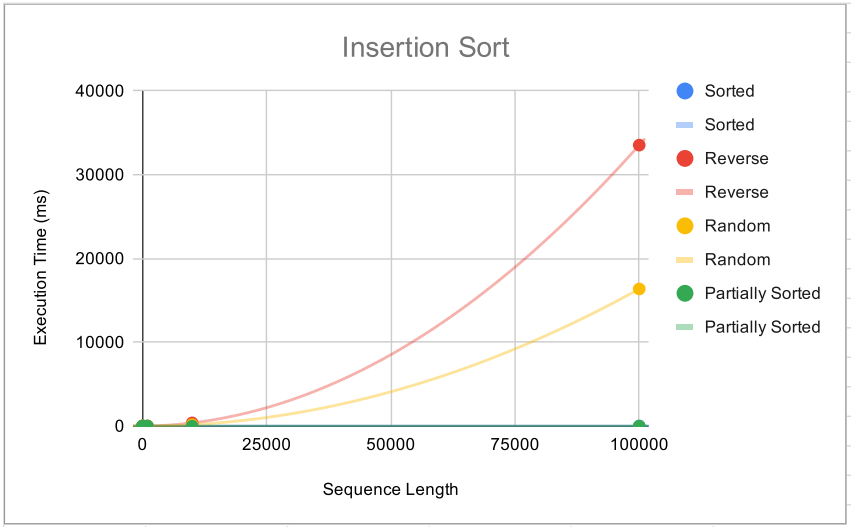
Project	Name	Done by date	Feedback	Review Code	Notes
Finished project goes here	**Who is submitting the project?**	**Put when the project was completed**	**Anything of importance when you finished the project**	**Finalize this part of the code to make sure it works**	**Does it run?**
Insertion sort	Tuyetlinh	11/25/2020	successfully implemented insertion sort & used rand() c++ function to generate random arrays of random size (see comments in cpp file)	YES (TN) YES (JZ) YES (LO)	YES
Merge sort	Leslie	11/25/2020	successfully implemented merge sort implemented same main as insertion sort to test	YES (LO) YES (TN) YES (JZ)	YES
Quick sort	Jacob	11/25/2020	successfully implemented quicksort implemented same main as insertion sort to test	YES (JZ) YES (LO) YES (TN)	YES
Radix Sort	ALL 3	11/30/2020	successfully implemented radix sort implemented same main as insertion sort to test	YES (ALL 3)	YES
Sorter Class	ALL 3	12/4/2020	created a class to allow user to choose which algorithm to write a CSV file for - later implemented an example ability	YES (ALL 3)	YES
Presentation	ALL 3	12/9/2020	created a presentation that overviews the 4 algorithms and discusses ours findings after implementing them	YES (ALL 3)	N/A

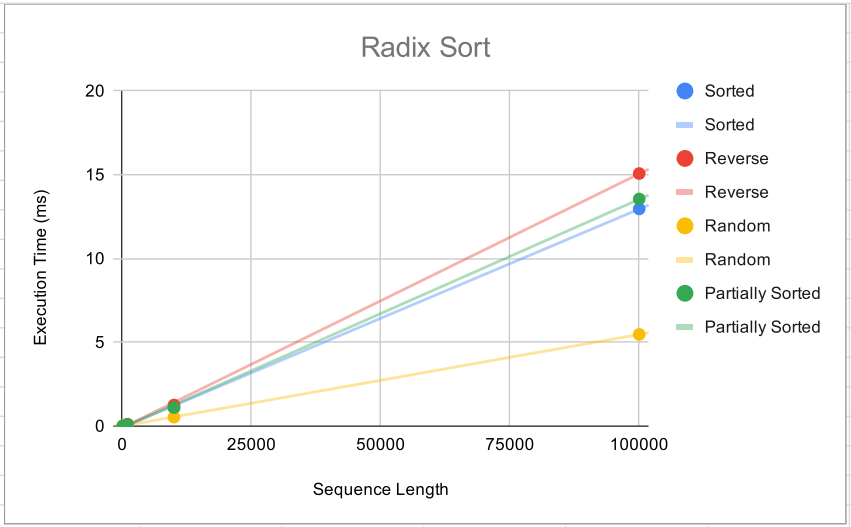
Report	ALL 3	12/12/2020	finalized an official report, similar to the presentation, but with a bigger focus on our findings when benchmarking the algorithms	YES (ALL 3)	N/A

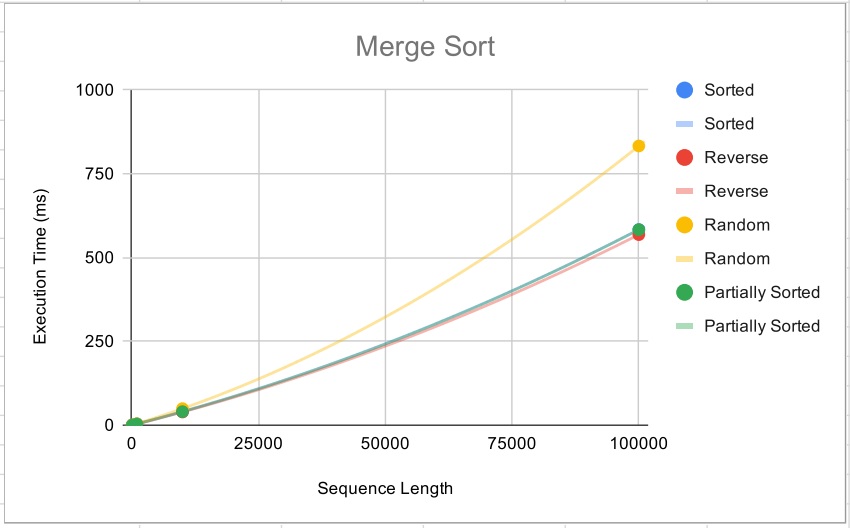
Date	Attendees	Accomplished Tasks	Notes
11/16/2020	Tuyetlinh, Leslie, Jacob	-Went over the rubric -discussed deadlines for the project	Collaborating on Tuyetlinh's IDE: https://ide.cs50.io/tuyetlinhnguyen/ide
11/20/2020	Lesie, Jacob, Tuyetlinh	-end of week meeting -discuss gameplan and setting up each algorithm -deciding on which unique algorithm to choose	
11/30/2020	Tuyetlinh, Leslie, Jacob	-post-break check-in -discussed working algorithms -peer-debugged/check all sorts	
11/16/2020	Jacob	-Asking TA about project, how to use GitHUB	
12/3/2020	Jacob, Tuyetlinh	-asking TA questions	Key Points: the smaller the main file is the better; structure presentation as though you are teaching it - show how it runs - how benchmarks were performed; make one progam to run; classes are highly recommended -contain the data sets -contain the algorithms to be run
12/4/2020	Leslie, Tuyetlinh, Jacob	-end of week meeting -talking about what we are going to do over the weekend - creating one file to run all of the code on -discussed the presentation and a tentative plan for a rehearsal	
12/11/2020	Leslie, Tuyetlinh, Jacob	-finalized presentation together and briefly went through the slides together -went reviewed and edited the project report together -picked a day to meet for a mock presentation	

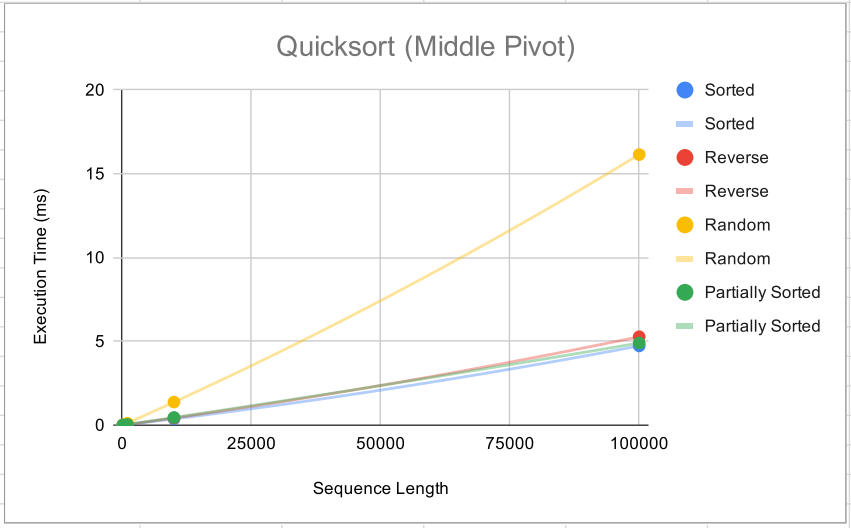
12/13/2020	Leslie, Tuyetlinh, Jacob	-mock presentation -finalized report and submitted to gradescope	last meeting with the group before presentation; discussing each of our roles in the presentation and made critiques; asked questions after the presentation about the subject material

		Execution Time (milliseconds ms)						
Sequence Type	Sequence Length	Insertion Sort	Radix Sort	Merge Sort	Quicksort		Least Efficient	Most Efficient
Sorted	1	0.002	0.001	0.002	0.001			
Sorted	10	0.002	0.002	0.025	0.001			
Sorted	100	0.002	0.006	0.307	0.003			
Sorted	1000	0.009	0.089	3.386	0.03			
Sorted	10000	0.089	1.099	40.532	0.372		Merge Sort	Insertion Sort
Sorted	100000	0.885	12.963	583.696	4.727		583.696	0.885
Reverse	1	0.001	0.001	0.001	0.001			
Reverse	10	0.001	0.002	0.045	0.001			
Reverse	100	0.034	0.008	0.295	0.003			
Reverse	1000	3.169	0.114	3.335	0.036			
Reverse	10000	384.967	1.259	39.234	0.429		Insertion Sort	Quicksort
Reverse	100000	33539.898	15.072	568.438	5.276		33539.898	5.276
Random	1	0.001	0.001	0.001	0.001			
Random	10	0.001	0.002	0.123	0.001			
Random	100	0.017	0.01	0.371	0.01			
Random	1000	1.51	0.06	4.137	0.114			
Random	10000	165.197	0.533	49.342	1.376		Insertion Sort	Radix Sort
Random	100000	16369.747	5.473	833.059	16.147		16369.747	5.473
Partial	1	0.001	0.001	0	0.001			
Partial	10	0.001	0.001	0.131	0.001			
Partial	100	0.001	0.007	0.308	0.005			
Partial	1000	0.011	0.089	3.37	0.036			
Partial	10000	0.108	1.101	40.263	0.456		Merge Sort	Insertion Sort
Partial	100000	1.078	13.554	583.258	4.901		583.258	1.078

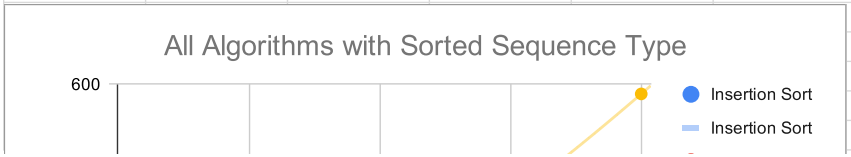


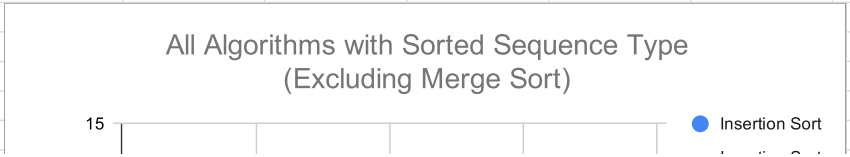


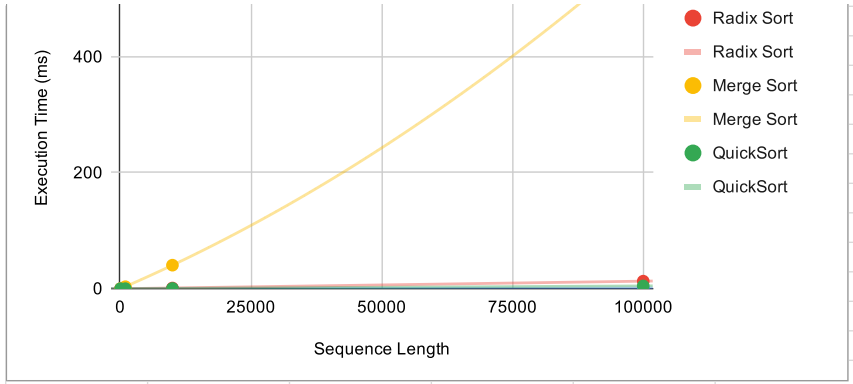


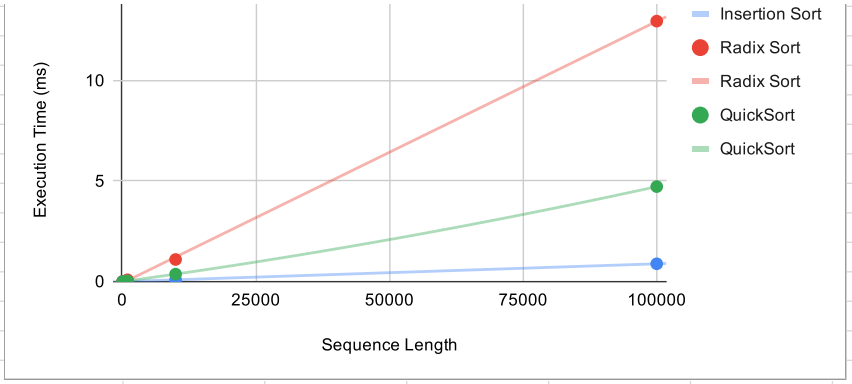


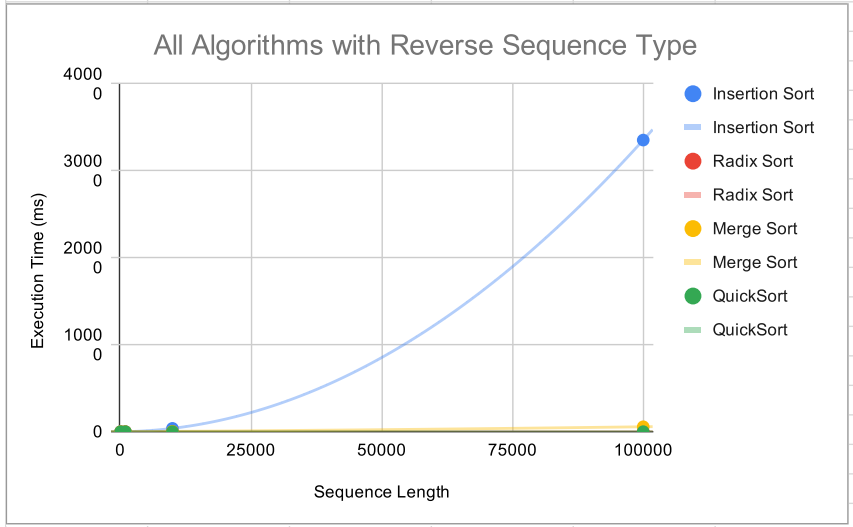
Comparing the runtimes of each algorithm at 100,000 sequence size per sequence type

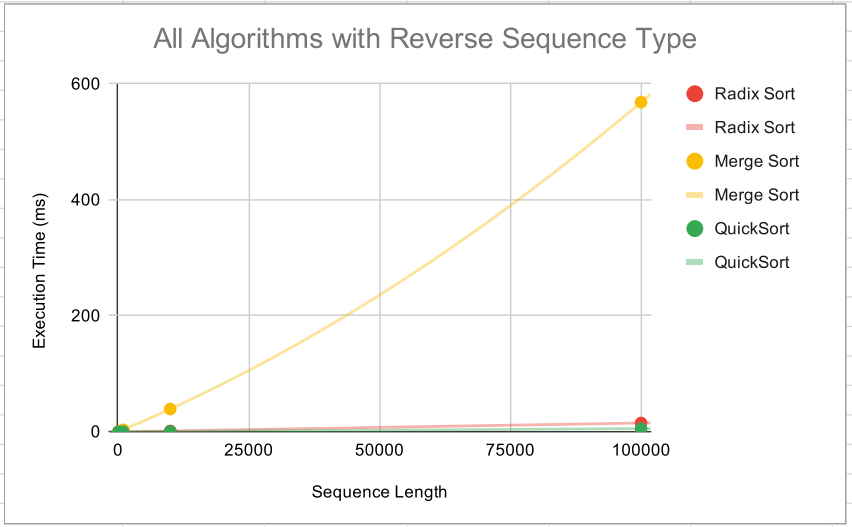


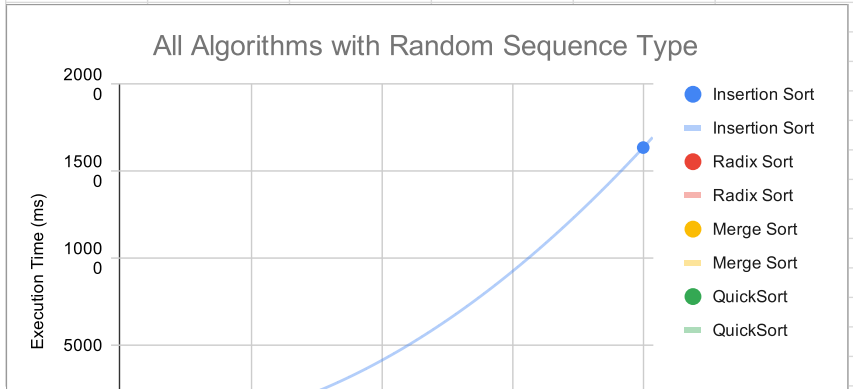


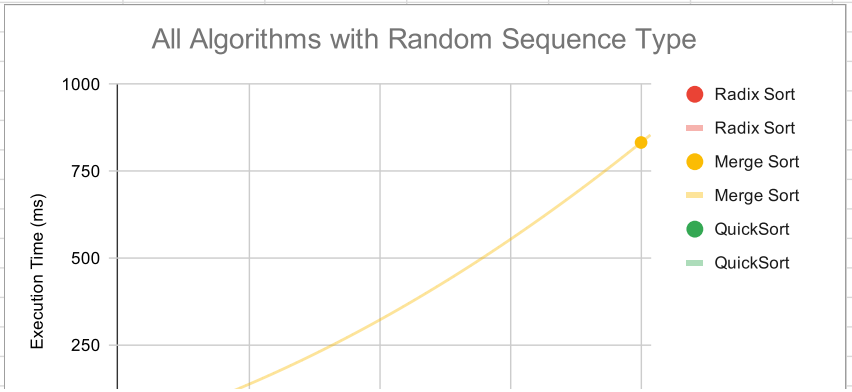

Execution Time (ms)
400
200
0
0
25000
50000
75000
100000
Sequence Length
Radix Sort
Radix Sort
Merge Sort
Merge Sort
QuickSort
QuickSort

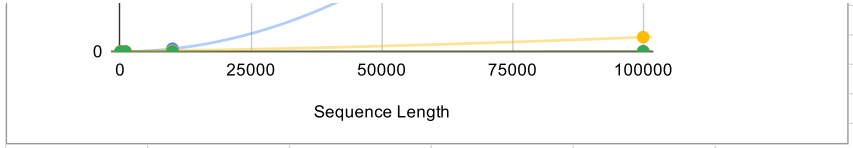

Execution Time (ms)
10
5
0
0
25000
50000
75000
100000
Sequence Length
Insertion Sort
Radix Sort
Radix Sort
QuickSort
QuickSort

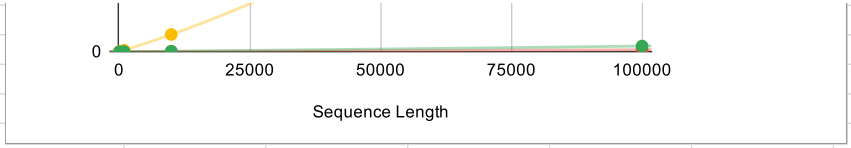

All Algorithms with Reverse Sequence Type
Execution Time (ms)
4000
0
3000
0
2000
0
1000
0
0
0
25000
50000
75000
100000
Sequence Length
Insertion Sort
Insertion Sort
Radix Sort
Radix Sort
Merge Sort
Merge Sort
QuickSort
QuickSort

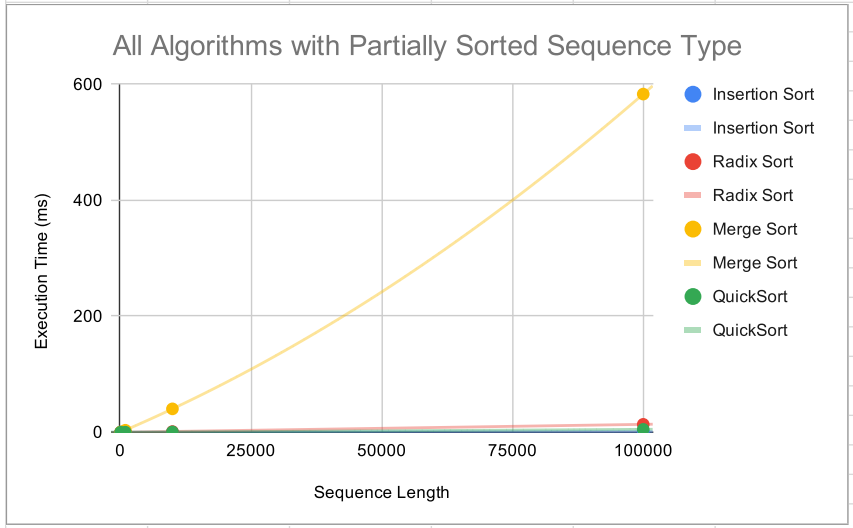

All Algorithms with Reverse Sequence Type
Execution Time (ms)
600
400
200
0
0
25000
50000
75000
100000
Sequence Length
Radix Sort
Radix Sort
Merge Sort
Merge Sort
QuickSort
QuickSort

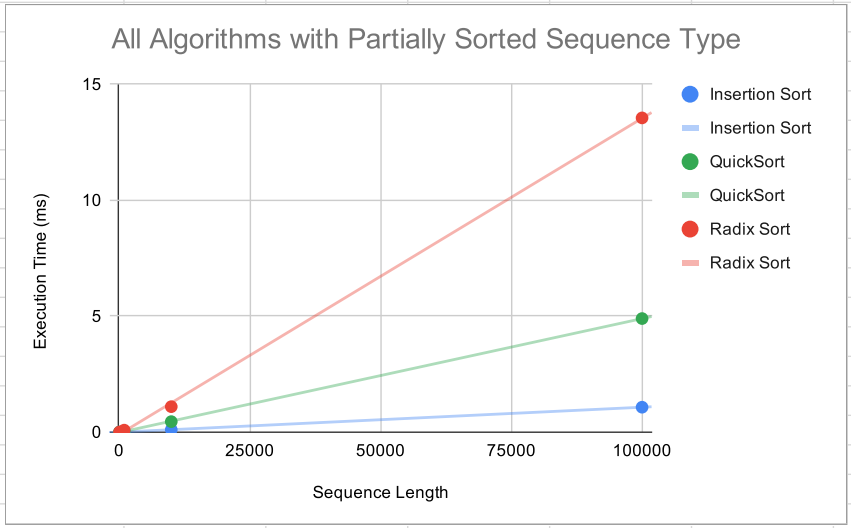

All Algorithms with Random Sequence Type
Execution Time (ms)
2000
0
1500
0
1000
0
5000
Insertion Sort
Insertion Sort
Radix Sort
Radix Sort
Merge Sort
Merge Sort
QuickSort
QuickSort


All Algorithms with Random Sequence Type
Execution Time (ms)
1000
750
500
250
Radix Sort
Radix Sort
Merge Sort
Merge Sort
QuickSort
QuickSort


0
0
25000
50000
75000
100000
Sequence Length


0
0
25000
50000
75000
100000
Sequence Length


All Algorithms with Partially Sorted Sequence Type
Execution Time (ms)
600
400
200
0
0
25000
50000
75000
100000
Sequence Length
Insertion Sort
Insertion Sort
Radix Sort
Radix Sort
Merge Sort
Merge Sort
QuickSort
QuickSort


All Algorithms with Partially Sorted Sequence Type
Execution Time (ms)
15
10
5
0
0
25000
50000
75000
100000
Sequence Length
Insertion Sort
Insertion Sort
QuickSort
QuickSort
Radix Sort
Radix Sort