Tyler Bartlett
02/10/2019
Dr. Breeann Flesch
CS361 – Algorithms

Lab 2 Report

I verified the radix Sort worked by printing off the first ten numbers read into the file and printed them again after the sort. I also ran it through the *flgIsSorted()* method to verify sort as well. Output of this:

```
data read in from file:
5449937
7646474
1202789
3140323
5406434
6083164
4088282
9737449
8323914
4530583
```

```
Array after radix sort:
1202789
3140323
4088282
4530583
5406434
5449937
6083164
7646474
8323914
9737449
```

```
checking if array is sorted...
arr is sorted!
```

I waited to the last minute to write the lab and did not get the bucket sort algorithm to work in time. I have skipped it for this lab and may come back later to fix it. I will not be able to test my algorithm at 1 million or 10 million numbers read in as the time taken will exceed what I can do for this report to be turned in on time. I copied the merge sort from lab 1 to lab 2 to compare my radix sort against. Below is a table of measured times.

Lines Read:	Run Number:	time (seconds) taken to read file:	time (seconds) taken on radix sort:	time (seconds) taken to verify sort after radix sort:	time (seconds) taken on merge sort:	time (seconds) taken to verify sort after merge sort:
1000	Run 1:	0	0.006	0	1.204	0
	Run 2:	0.001	0.005	0.001	1.002	0
	Run 3:	0.001	0.011	0.001	1.311	0
	Average:	0.000666667	0.007333333	0.000666667	1.172333333	0
10000	Run 1:	0.008	0.104	0	14.725	0
	Run 2:	0.005	0.144	0.001	14.941	0
	Run 3:	0.003	0.119	0	14.601	0
	Average:	0.005333333	0.122333333	0.000333333	14.75566667	0
100000	Run 1:	0.074	1.118	0.001	129.86	0.002
	Run 2:	0.074	1.353	0.001	115.414	0.001
	Run 3:	0.043	1.1	0.002	104.743	0.002
	Average:	0.063666667	1.190333333	0.001333333	116.6723333	0.001666667
1000000	Run 1:	0.691	17.275	0.016		
	Run 2:					
	Run 3:					
	Average:	0.691	17.275	0.016	#DIV/0!	#DIV/0!