CS&E A311, Spring 2020
Assignment 4
10 points
Due 18 Mar 2020 at 11:59pm to *cse2*

This assignment will give you practice writing Python code for implementing and empirically testing the performance of CountSort. Be sure to submit your files to the grading directory on cse2: `/usr/local/class/csce311s20/a4/username/`. The latest file timestamp in your directory will be used to determine the assignment submission time. Style and documentation points are worth approximately 10% of the point total and are awarded independent of the correct functioning of your program.

1. (10 pts) Non-Comparison-Based Sorting Implementation, Python

In a3, you used Python to write an implementation of both HeapSort and QuickSort to sort an array of random integers with values ranging from 0 to 100,000. Both algorithms are guaranteed or expected O($n$lg$n$) runtime.

For this assignment, write a Python implementation of CountSort (using the stable count sort pseudocode provided in lecture) for the same problem. First, test your algorithm on 25 random integers ranging from 1 to maximum integer, k=20. Print array A, C, and B to show me that your algorithm works.

Then, run your algorithm with the same size arrays from a3 with integers ranging from 1 to k=100000. With a O(n) expected runtime, how does it fare in practice compared to HeapSort and QuickSort? Update your plot from a3 to show how CountSort compares.

How big of an effect does k have on the efficiency of CountSort? Why? Try it with other values of k, such as [1000, 1e6, 1e7].

Under what conditions would HeapSort be faster than CountSort?

Submit 2 files to cse2

- Your plot (a png or jpg file) with runtimes for QuickSort, HeapSort, and CountSort (for k=[100000, 1e6, 1e7])
- Your Python code with a comment that answers the above questions and explains why.