I implemented four different sorting algorithms to compare their performance in my program. I expected sorting algorithms with O(n^2) to perform slower than O(n log n). However, different algorithms have unique trade-offs which affect the margin at which O(n log n) performs faster than O(n^2). I expected the radix sort to be the quickest algorithm with a time complexity of O(NK). Radix sort did have the fastest run time of 0.006 ms. Merge sort had the second-fastest runtime, which was expected since it is in the order of O(nlogn). The remaining two algorithms, selection and bubble sort were in the same order of O(n^2) time. However, bubble sort had a runtime 1.333 ms slower than selection sort. Therefore, all my expectations were met by the algorithms since bubble sort's comparisons and swaps mare O(n^2), while selections sort have a faster runtime of (N·(N−1))/2 and O(n), respectively.

The selection sort algorithm takes on the order of O(n^2) steps since it takes (N·(N−1))/2 comparisons and N swaps to sort. The advantage of selection sort is the number of swaps is O(n), and it is an in-place algorithm. However, the primary trade-off is the number of comparisons that take on the order of O(n^2) time. The merge sort algorithm has a worst-case runtime complexity of O(n log n). This is since the algorithm divides the array into two halves and takes linear time to merge the two halves. Merge sort's divide and conquer method can sort large amounts of data faster, in the order of O(n log n), compared to selection sort. However, since merge sort is an out-of-place algorithm, the trade-off is the extra space required to run. The bubble sort algorithm runtime is in the order of O(n^2), which means it is slower on larger inputs. Bubble sort is slower because it requires O(n^2) swaps and comparisons. Despite having the same worst-case runtime complexity as selection sort, selection sort is faster than bubble sort in its comparisons and swaps. I also implemented radix sort as a different sorting algorithm to compare results since it can efficiently sort integers in O(n) time. However, radix sort isn't a flexible algorithm and only can sort integers.