Homework Assignment 5

Output:

For N = 10:
Average Insertion Sort Time: 0.0007458 milliseconds
Average Bucket Sort Time: 0.0114041 milliseconds
Average Radix Sort Time: 0.0052417 milliseconds

For N = 100:
Average Insertion Sort Time: 0.0372459 milliseconds
Average Bucket Sort Time: 0.052446 milliseconds
Average Radix Sort Time: 0.0195503 milliseconds

For N = 1000:
Average Insertion Sort Time: 3.95211 milliseconds
Average Bucket Sort Time: 0.317704 milliseconds
Average Radix Sort Time: 0.229292 milliseconds

For N = 10000:
Average Insertion Sort Time: 163.988 milliseconds
Average Bucket Sort Time: 1.0259 milliseconds
Average Radix Sort Time: 0.752746 milliseconds

Which algorithm is faster for small N?

For smaller N insertion sort is faster than Bucket sort and Radix sort.

What about for large N?

For the large N Radix sort is faster with the impressive time of 0.752746 milliseconds.

How do the wall clock times and growth rates compare to the big O average time complexities?

Insertion Sort:

The wall clock times increase significantly as N grows, consistent with the expected quadratic time complexity of O(N^2) for Insertion Sort.

Bucket Sort:

Bucket Sort exhibits relatively stable performance across different N, aligning with its linear time complexity (O(N + k)). Variations are influenced by factors such as the number of buckets (k) and the distribution of input data.

Radix Sort:

Radix Sort demonstrates efficiency, especially for larger N, in line with its linear time complexity (O(d * (N + k))), where d is the number of digits in the maximum number, showcasing its ability to handle substantial datasets by sorting numbers digit by digit.