10) On 10-sorted data, the algorithms performed more efficiently than on fully random data, with less drastic differences between the runtimes. The ranking was mostly consistent with the random data results, but insertion sort and shell sort showed significantly better performance due to their efficiency with nearly sorted data. Bubble sort remained the slowest due to its inefficiency regardless of data order.

12) When plotting very large input sizes, runtimes became too large to handle accurately due to overflow or limitations in timing precision. This reflects the computational strain and precision loss when dealing with extremely large datasets, highlighting the practical limits of some sorting algorithms and testing methods.