Fairly easy to determine that heap sort is way better. Both followed their time trends, O(nlog(n)) and O(n2) respectively. It is obvious because the bubble sort needs to loop through the entire data structure multiple times to completely sort it, meanwhile the heap sort takes n operations to build the heap and then logn to find the next largest element and add it to the sorted list. The heap sort excels as it uses binary tree logic to keep priority of elements and reduce search time since complete binary trees allow for logn performances. If the tree were not always complete, the time complexity would likely be worse since more operations would be needed to fill in the gaps in the array used to store the data. The key part for good heap sorting is the completion of the tree and maintaining min/max heap rules. Bubble sort is just so bad because again, the larger the data set, the more the time complexity will tend to O(n2). However, if the unsorted array somehow only needed one passthrough of swaps, then the time complexity could be O(n). But that is extremely unrealistic and very unlikely, so the trends follow the former.