Project 1 Write-Up

Explanation of Implementation:

The first issue is encountered as far as a design decision goes was how to check for duplicate time stamps. Each input loop iteration required you to check if the current line's timestamp was already present in the saved data. There were two ways to do this, linear search, or sorting and then using binary search. Ultimately, though, sorting would have caused a larger issue in my code. It would make it impossible to determine which lines come from what files because the input order would be changed. For this reason, I decided to implement the time-stamp check using linear search.

For a linear search each iteration over n lines of data, the worst-case time complexity of this implementation is O(n^2).

The second main issue was the ability to search for given instances of data. The main problem with this was modifying binary and linear search to account for multiple instances of data and return them all. Binary search was not directly modified, but instead modified outside of the method. It would return an index if the data was found, and then I had two separate loops running up and down the data from that index. They would run until they encountered data not equal to the key. Then, a third loop would go from the low index to the high index and print out each line in between.

For this implementation, in the case that all indices of the array are equal to the key, the worst-case would be log(n) + 2(n/2) + n, where log(n) is the initial binary search and 2(n/2) is the iterating up and down the list from the key index. Finally, n represents the final iteration through all array elements to print them to standard out. Therefore, the Big-O of this implementation is O(n).

If the data was unsorted, the linear search was modified slightly to accept a starting index. This would allow it to iteratively check smaller subsets of the data each time a valid key was encountered. To effectively iterate over all the data and print it out would require, in the worst-case, 2n operations, one to check the data and one to print it out. The Big-O of this implementation would also be O(n).

So, in the worst-case, these implementations are going to be in the same time complexity class, will linear search even out-performing the binary search implementation. However, it could be assumed that the majority of the time there would only be a few matching keys, and whereas linear search would have a constant runtime of n, the binary search implementation will regularly outperform this with a log(n)—plus some constant—runtime.