Algorithm:

Read in speedrunners from file to DS

& MergeSort DS by Doom time

Write speedrunners to output file

Analysis: Reading and writing both take O(n) for the n loops ran to get each speedrunner

Mergesort takes O(n log n)

the O(n) component comes from the comparisons of each subset when merging and the O(log n) component comes from the fact that our set is split in half with each recursive call to mergesort and merged to be twice as large when merging so there are log n + 1 levels (accounting for the base case with +1)

Thus we have $O(n) \times O(\log n) = O(n \log n)$

Proof. Assume we have our greedy algorithm A that outputs the order $\{a_1, \ldots a_n\}$ and an optimal algorithm O that outputs order $\{o_1, \ldots o_n\}$. We now that an optimal algorithm outputs a schedule with 0 idle time and 0 inversions. Due to the fact that each task begins immediately after the other, it must be the case that there is 0 idle time for both of these schedules. Now if both of these orders have 0 idle time and 0 inversions then they both have the same max lateness (or finish time). Now if we order our schedule in such a way that our slowest Doom 2 players go first, this will lead to us minimizing the amount of inversions such that in the event that one speedrunner takes longer to finish Doom 2 than the next speedrunner takes to finish Mario and Doom 2, we will still minimize the max finish time.

If we assume both schedules have the same amount of inversions and idle time, then they also have the same max lateness. If O and A have the same max idle time and inversions, then we can swap the order of O to match A and still have the same max lateness. Thus if O is optimal and has the same finish time as A then A is also optimal