

Mathematical Analysis:

The program is a sequence containing light and dark disks that moves in an alternating pattern, where there cannot be a repeating light or dark disk. Building upon and deleting some parts from the linked list algorithm displays a time complexity of $O(N)$ because there is a one-to-one cardinality between the actions. However, time complexity is different under the core part of the algorithm, which is the recursiveSort function. Each call from the recursive function has a time complexity of $O(1)$ because there is a constant number of comparisons and possible swaps from the left and right. The worst-case scenario for the program in terms of time complexity would result in a maximum of $O(n^2)$, which is dependent on the depth and the distance of travel it needs. Through induction,

Assume that for any list of Size k , the algorithm sorts itself in $O(k^2)$ time, where the base case immediately returns to $O(1)$ through a single node.

The inductive step is $(n = k + 1)$ where the work would be: $T(k+1) = O(k) + O(k^2) = O((k+1)^2)$

Thus, by induction, the algorithm runs in quadratic time.