Initial Array:{ D, L, D, L, D, L}

First Pass:

At index 0: D swaps with L. Array becomes: L D D L D L (1 move).

At index 1: No swap (as it's a 'D' and next is also 'D').

At index 2: D swaps with L. Array becomes: L D L D D L (2 moves).

At index 3: D swaps with L. Array becomes: L D L L D D (3 moves).

Result:

Final Array: L L L D D D

Total Moves: 3

The time complexity of the alternate\_disks function is **O(n^2)**, where n is the length of the array. The outer loop executes n times, and for each 'D' disk encountered, the inner while loop could run up to n iterations in the worst-case scenario.