Copy List with Random Pointer

For this problem we need to construct a deep copy of the list, including both next and random pointers.

Optimal Approach: [O(n) time, O(1) space

Key Idea: 3-steps in-place algorithm
We can avoid extra hash maps by modifying the list temporarily

Steps:

Step 1: Interweave copied nodes with original nodes.
- For each node in the original list, create a copy and insert it right after the original node.
- After this step, list looks like:
  Original: A → B → C
  Modified: A → A' → B → B' → C → C'

Step 2: Assign random pointers to copied nodes
- For each original node, set copy.random = original.random.next (if random exists).

Step 3: Separate the two lists
- Detach the copied list from the interleaved structure to restore the original list.

Complexity Analysis:
- Time: O(n) (Traverse list 3 times)
- Space: O(1) (In-place without extra hash maps)