Problem of the Week – Deep Clone a Linked List with Random Pointer

Company: Snapchat

Scenario:

You are given a **singly linked list** where each node contains two pointers:

- `next`: Points to the next node in the list
- `random`: Points to any node in the list (or `null`)

Your task is to **create a deep copy** of this list. That means you should create a new list where each node is a **new object**, and has the same value and same structure (both `next` and `random` pointers) as the original list.

Input Format:

- A head node of a singly linked list. Each node contains:
  - `int val`
  - `Node* next`
  - `Node* random`

Output Format:

- Return the head of the **deep cloned linked list**.

Example:

Input:

A linked list represented as:

```
Node1(val=7) → Node2(val=13) → Node3(val=11) → Node4(val=10) → Node5(val=1)

Random pointers:
Node2.random → Node1
Node3.random → Node5
Node4.random → Node3
Node5.random → Node1
```

Output:

A deep clone with same structure but different memory references.

Expected Output Verification:

You should verify:

- Values are identical
- `next` and `random` pointers point to corresponding new nodes
- Original and cloned lists are disconnected (changing one doesn't affect the other)

Approach Hints:

- Use a **hash map** to store the mapping of original nodes → cloned nodes
- Traverse the list in two passes:
    1. Clone all nodes and store them in map
    2. Assign `next` and `random` using the map

Or

- Use **O(1) space** by interleaving the cloned nodes with original list, then separating later.

Practice Links:

- LeetCode – Copy List with Random Pointer
- GFG – Clone a linked list with next and random pointer

Video Solutions:

- YouTube (NeetCode) – Copy List with Random Pointer
- Take U Forward – Clone Linked List | Random Pointer