DAY-37.

(1). Clone a linked list with next and random pointer in O(1) space.

Given a linked list having two pointers in each node. The first ~~node~~ one points to the next node of the list, however, the other pointer is random and can point to any node of the list.

Approach:-

→ Create a copy of node1 and insert it between node1 and node2 in original linked list, create a copy of 2 and insert it between 2 & 3, Continue in this fashion, add the copy of N after the Nth node.

→ Now copy the random linked list in this fashion:--

original → next → random = original → random → next.

[(Traverse two nodes).

This works because original → next is nothing but copy of original → random → next is nothing but copy of random.

→ Now restore the original and copy linked lists in this fashion in a single loop.

original → next = original → next → next

copy → next = copy → next → next.

→ Ensure that original → next is NULL and return the cloned list.

Implementation:-

```
def clone(original_root) :
        curr = original_root
        while  curr != None:
              new = Node(curr.data)
              new.next = curr.next
              curr.next = new
              curr = curr.next.next
''(the above is to insert node after every node)'''
        curr = original_root
        while   curr != None:
              curr.next.random
                    = curr.random.next
              curr = curr.next.next
''' it's to adjust the random pointers
      of newly added nodes'''

''' Detaching original and duplicate list'''
      curr = original_root
      dup_root = original_root.next
      while curr.next != None:
            tmp = curr.next
            curr.next = curr.next.next
            curr = tmp.
      return dup_root.
```