Clone a linked list

BRUTE FORCE : Hashing

Store the original node and create a new node and place it with the original node in the map. This new node is the clone of original node. Now traverse in the map and find the original and corresponding nodes to it one by one. Find the next and random pointers and point accordingly.

```
#include <bits/stdc++.h>

/************************************************************

    Following is the LinkedListLinkedListNode<int> class structure

    template <typename T>
    class LinkedListNode
    {
        public:
        T data;
        LinkedListNode<T> *next;
        LinkedListNode<T> *random;
        LinkedListNode(T data)
        {
            this->data = data;
            this->next = NULL;
        }
    };

************************************************************/

LinkedListNode<int> *cloneRandomList(LinkedListNode<int> *head)
{
    // Write your code here.
    if(head==NULL)
            return NULL;
    LinkedListNode<int>* temp=head;
    unordered_map<LinkedListNode<int>*,LinkedListNode<int>*> m;
    while(temp!=NULL)
    {
        LinkedListNode<int>* newNode=new LinkedListNode<int>(temp->data);
        m[temp]=newNode;
        temp=temp->next;
    }
    temp=head;
    while(temp!=NULL)
    {
        auto it=m.find(temp);
        auto nex=m.find(it->first->next);
        auto randm=m.find(it->first->random);
        if(nex!=m.end())
            it->second->next=nex->second;
        else
            it->second->next=NULL;
        if(randm!=m.end())
            it->second->random=randm->second;
        else
            it->second->random=NULL;
        temp=temp->next;
    }
    auto it=m.find(head);
    return it->second;
}
```

- Time Complexity : O(N)
- Space Complexity : O(N)

Optimal Approach :

Attach a new node to the next node of original node so that both lists get connected alternatively. Now connect the random pointer of cloned list. using iter→next→random = iter→random→next because the random next of original list will be the cloned node.

Step 3 is to connect next pointers and to connect next pointer we need to break the links between the cloned and original node.

```
/*
// Definition for a Node.
class Node {
public:
    int val;
    Node* next;
    Node* random;

    Node(int _val) {
        val = _val;
        next = NULL;
        random = NULL;
    }
};
*/
```

```
class Solution {
public:
    Node* copyRandomList(Node* head) {
        if(head==NULL)
        return NULL;
    Node *iter=head;

    //attaching new nodes
    while(iter!=NULL)
    {
        Node *newNode=new Node(iter->val);
        newNode->next=iter->next;
        iter->next=newNode;
        iter=iter->next->next;
    }
    iter=head;
    //setting random pointers
    while(iter!=NULL)
    {
        if(iter->random!=NULL)
            iter->next->random=iter->random->next;
        iter=iter->next->next;
    }
    iter=head;
    Node *dummy=new Node(0);
    Node *copy=dummy;
    Node *front;
    while(iter!=NULL)
    {
        front=iter->next->next;
        copy->next=iter->next;
        iter->next=front;
        copy=copy->next;
        iter=front;
    }
    return dummy->next;
    }
};
```

- Time Complexity : O(N)
- Space Complexity : O(1)