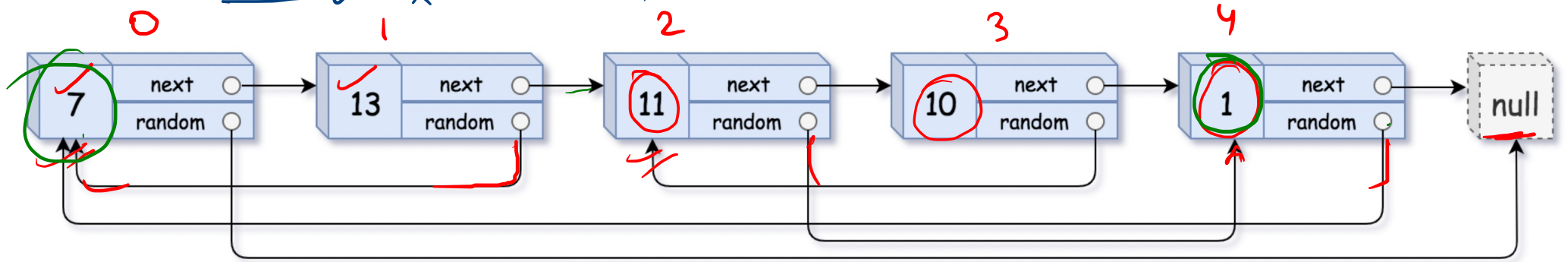


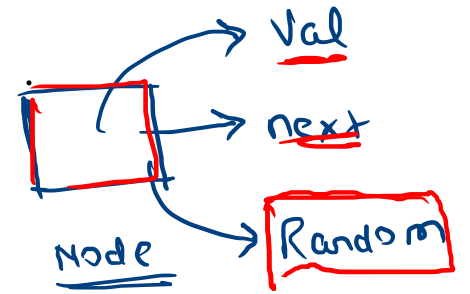
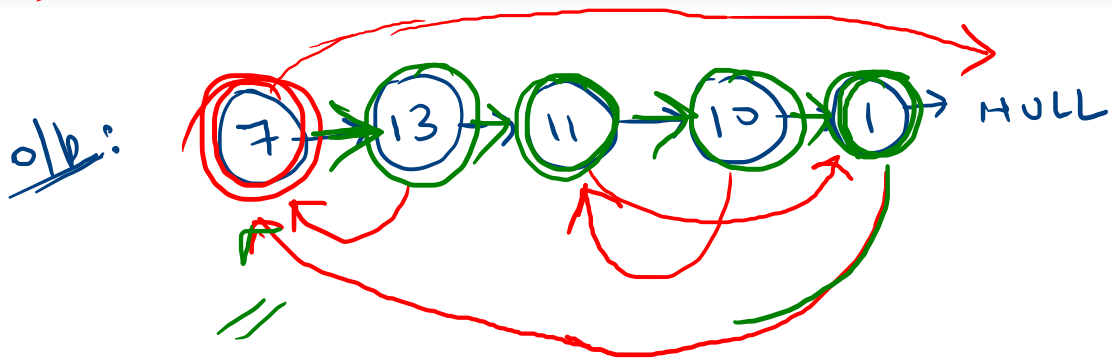
Copy List with Random pointer

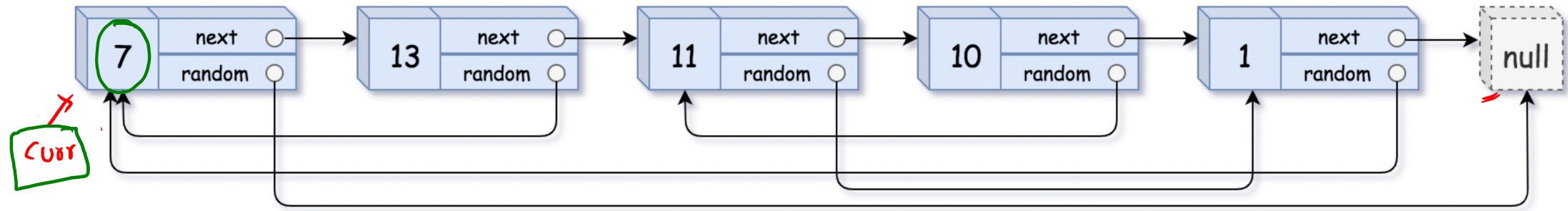
I/b:



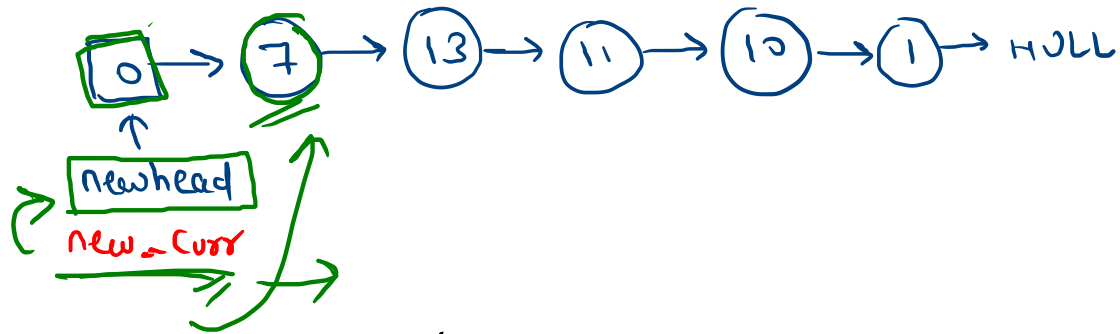
Input: head = [[7, null], [13, 0], [11, 4], [10, 2], [1, 0]]

Output: [[7, null], [13, 0], [11, 4], [10, 2], [1, 0]]



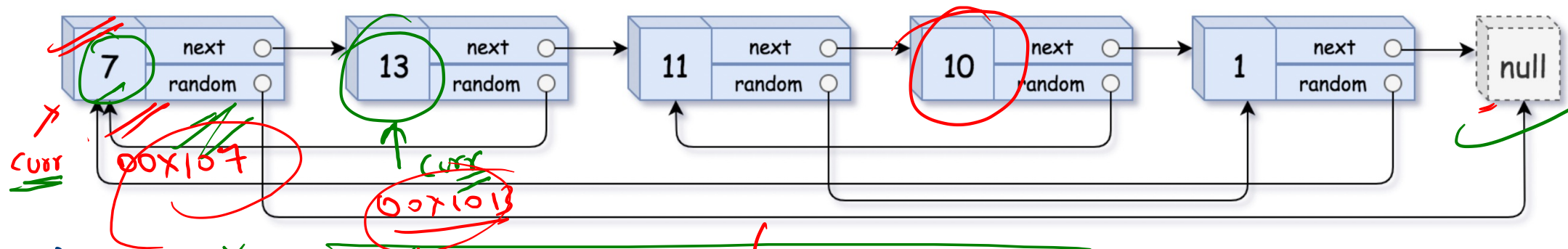


Sol:

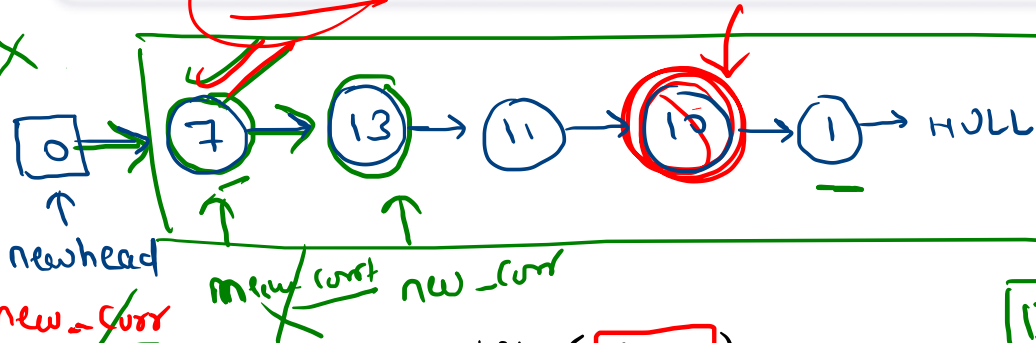


Node* newhead = new Node(0);
 while creating all new Node
 (HASH TABLE)

<u>Old Node</u>	<u>New Node</u>
7-old	7-new
13-old	13-new
11-old	11-new
10-old	10-new
1-old	1-new



Sol:



return

newhead -> next;

```

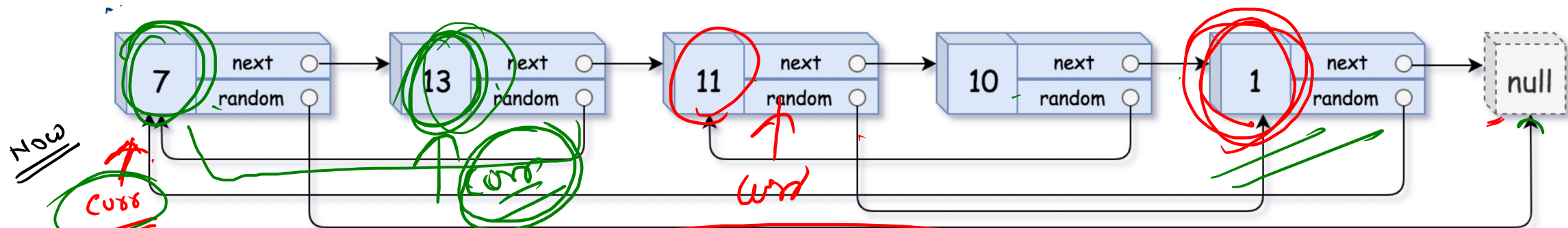
while (curr)
{
    Node* temp = new Node(curr->val);
    umap.insert(&curr, temp);
    curr->new -> next = temp;
    curr->new = curr->new -> next;
    curr = curr->next;
}

```

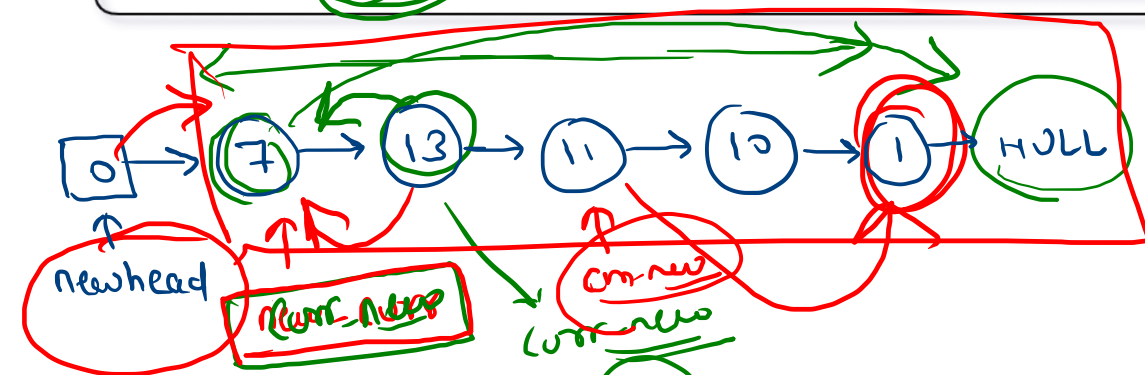
Node* newhead = new Node(0);
while creating all new Node

(HASH TABLE)

Old Node	New Node
7-old	7-new
13-old	13-new
11-old	11-new
10-old	10-new
1-old	1-new



Sol:



```

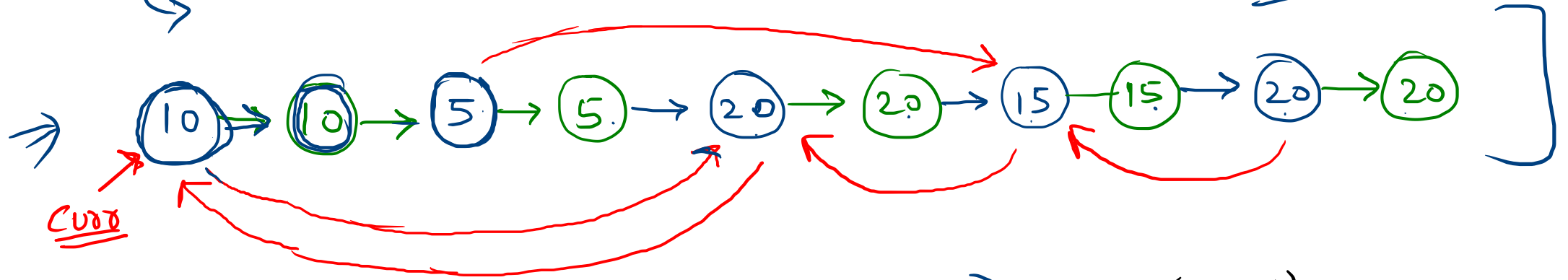
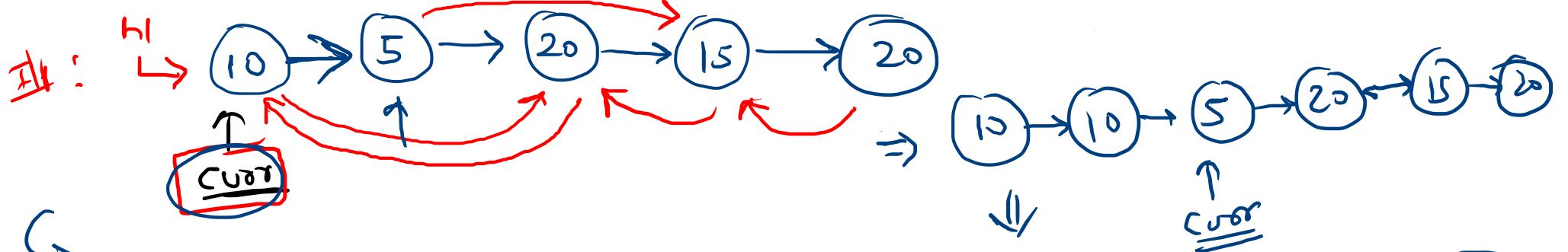
while (curr)
{
    Node* random = curr->random;
    Node* newNode = map[random];
    curr_new->random = newNode;
    curr = curr->next;
    curr_new = curr_new->next;
}

```

Node* newhead = new Node(0);
 while creating all new Node
 (HASH TABLE)

Old Node	New Node
7-old	7-new
13-old	13-new
11-old	11-new
10-old	10-new
1-old	1-new

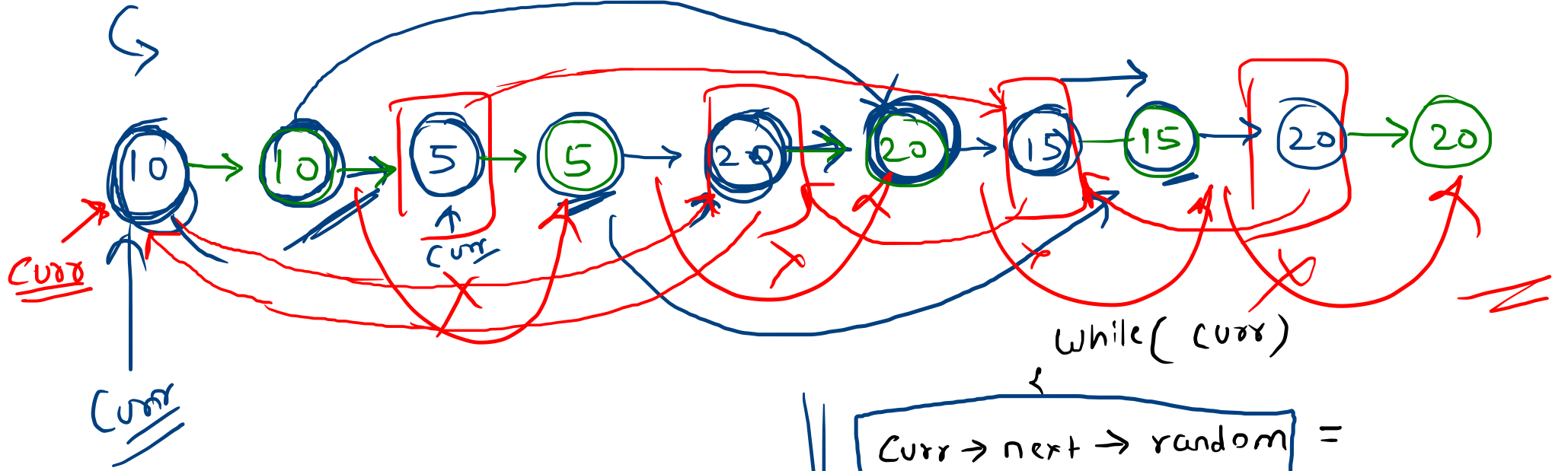
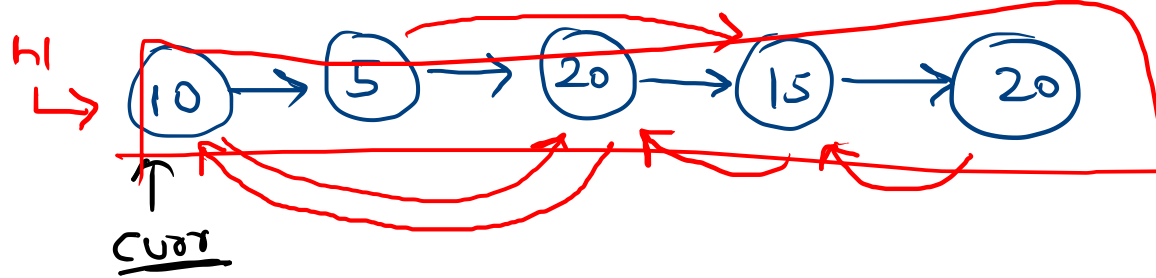
Tricky Approach



⇒ while (curr)

```
{  
    temp = curr → next;  
    curr → next = new Node(curr → val);  
    curr → next → next = temp;  
    curr = temp;  
}
```

Tricky Approach



while (curr)

curr -> next -> random =


(curr -> random != NULL) ?

curr -> random -> next;

NULL

(curr = curr -> next -> next)

```
class Solution {
public:
    Node* copyRandomList(Node* head) {
        if(head == NULL) return head;
        Node* curr = head;
        while(curr){
            Node* temp = curr->next;
            curr->next = new Node(curr->val);
            curr->next->next = temp;
            curr = temp;
        }
        curr = head;
        while(curr){
            curr->next->random = (curr->random != NULL) ? curr->random->next : NULL;
            curr = curr->next->next;
        }
        curr = head->next;
        while(curr and curr->next){
            curr->next = curr->next->next;
            curr = curr->next;
        }
        return head->next;
    }
};
```



Testcase

Run Code Result

Debugger 

Wrong Answer Runtime: 3 ms

Your input

```
[[7,null],[13,0],[11,4],[10,2],[1,0]]
```

Output

Next pointer of node with label 7 from the original list was modified.

Expected

```
[[7,null],[13,0],[11,4],[10,2],[1,0]]
```