# Odd-Even Linked List

---

leetcode link : Click

---

### Approach

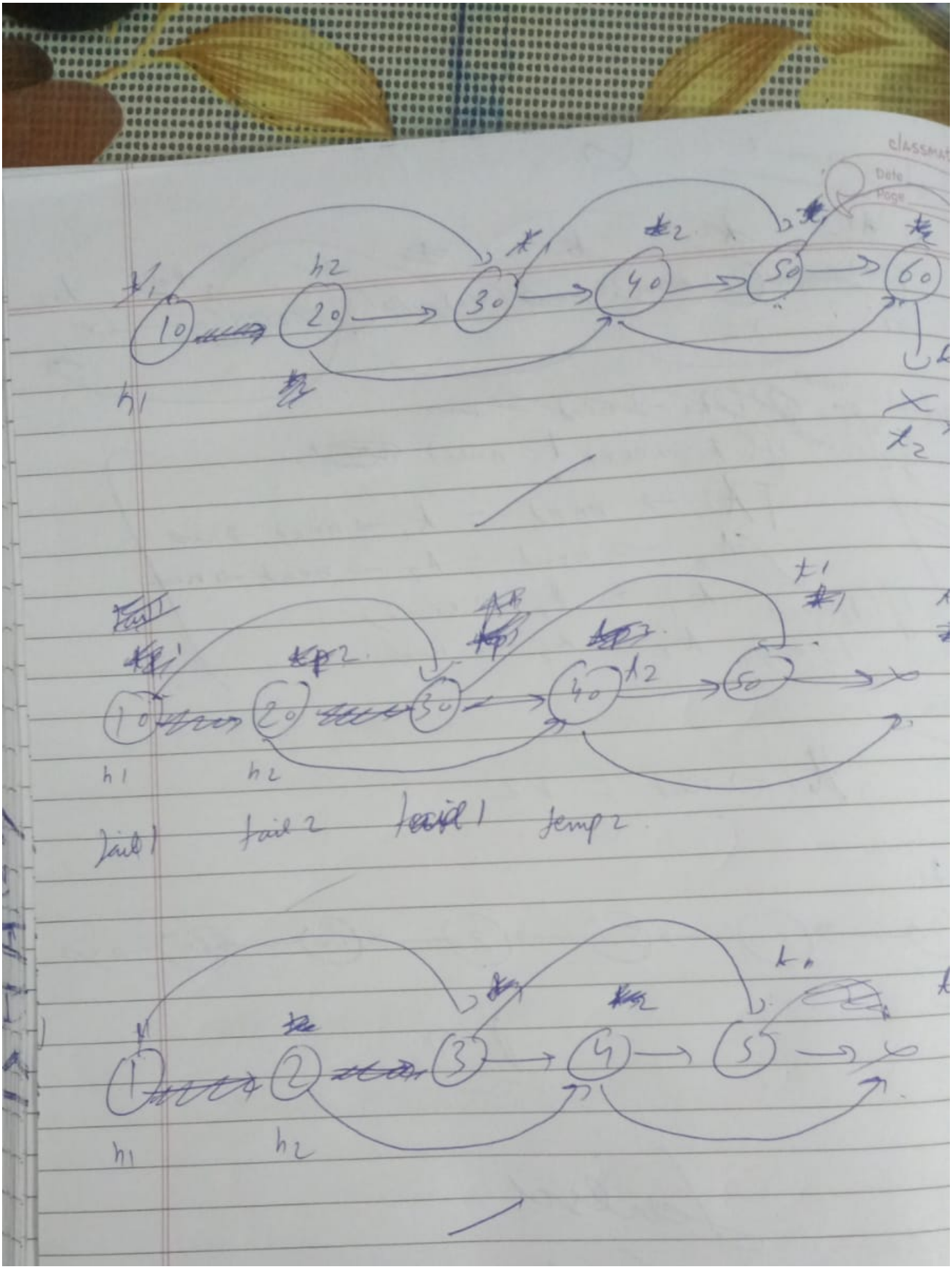
A simple iterative approach (in 1 traversal)

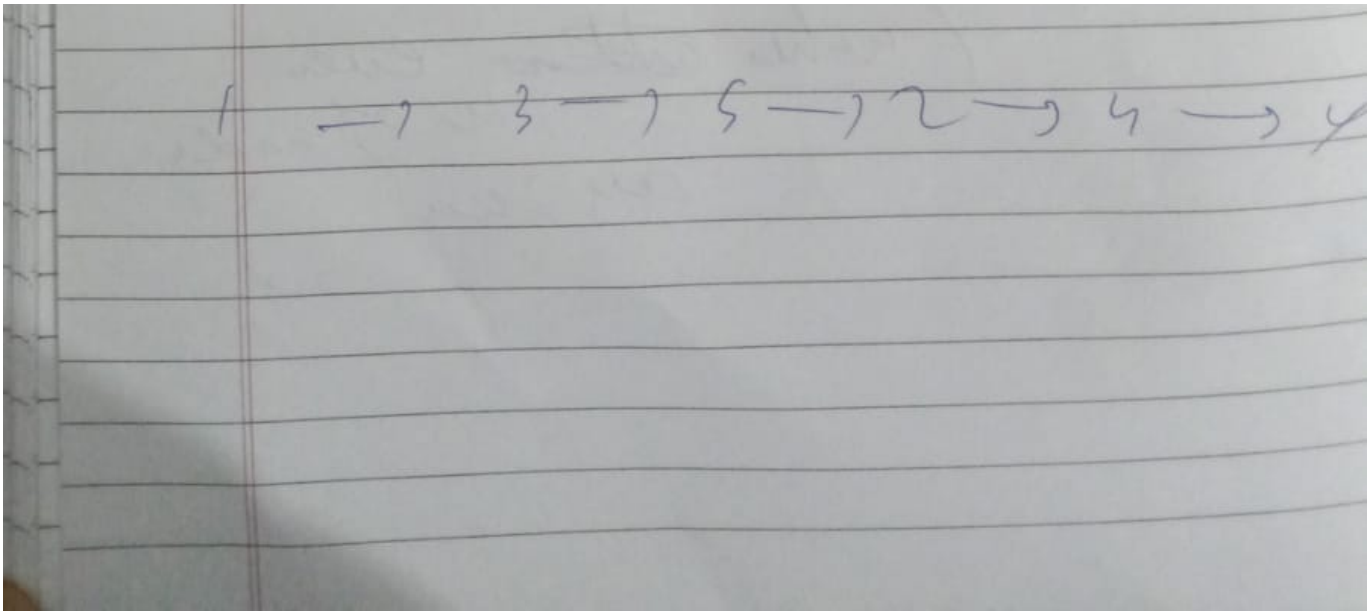
### Complexity

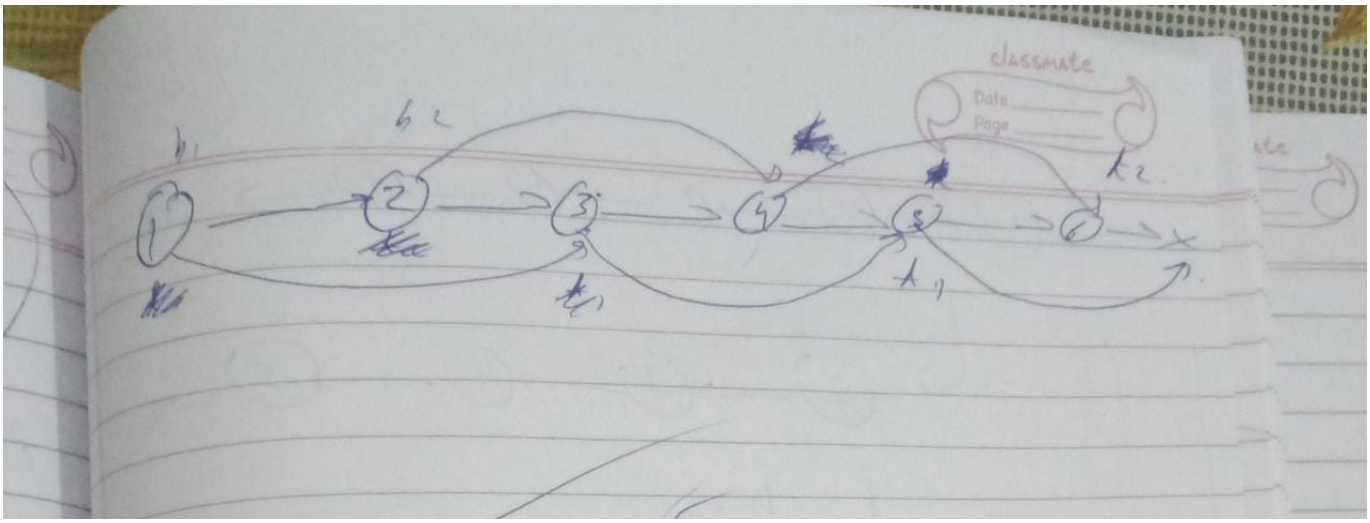
- Time complexity: $O(N)$
- Space complexity: $O(1)$

### Code

```
// Main function

    ListNode* oddEvenList(ListNode* head) {

        // exception case

        if(head == nullptr || head -> next == nullptr) return head;
        // since we will be dividing the linked list into 2 parts, so we will need to
store the 2 heads of the linked lists, for future use

        ListNode* head1 = head;
        ListNode* head2 = head -> next;

        // also we will be traversing the whole 2 lists, and change the 'next' links
so , lets save 2 pointers to nodes for that purpose

        ListNode* temp1 = head1;
        ListNode* temp2 = head2;

        // lets change the links while our temp nodes are not null

        while(temp1 && temp2){

            // changing links

            if(temp1 -> next){

                temp1 -> next = temp1 -> next -> next;

                // after linking the next odd node, make sure that if the 'next' node
is null, then break then loop
                if(temp1 -> next == nullptr) break;
            }

            if(temp2 -> next) temp2 -> next = temp2 -> next -> next;

            // updating pointers, updating tails
            temp1 = temp1 -> next;
            // when the temp1's next is null, that means now we need to join the 2
linked lists, and break this loop
            temp2 = temp2 -> next;

        }
        // when our link thing is done, we will join the end of list1 to the start of
list2 and return the merged list
        temp1 -> next = head2;

        return head1;

    }
```


10
20
30
40
50
60
h1
h2
1
2
3
4
5




----------END----------