12 Jan 2023

FIRST :-

Given a linked list of 0s, 1s and 2s, sort it. Easy

Given a linked list of **N** nodes where nodes can contain values **0s**, **1s,** and **2s** only. The task is to segregate **0s**, **1s,** and **2s** linked list such that all zeros segregate to head side, 2s at the end of the linked list, and 1s in the mid of 0s and 2s.

Example 1:

```
Input:
N = 8
value[] = {1,2,2,1,2,0,2,2}
Output: 0 1 1 2 2 2 2 2
Explanation: All the 0s are segregated
to the left end of the linked list,
2s to the right end of the list, and
1s in between.
```

Example 2:

```
Input:
N = 4
value[] = {2,2,0,1}
Output: 0 1 2 2
Explanation: After arranging all the
0s,1s and 2s in the given format,
the output will be 0 1 2 2.
```

Your Task:

The task is to complete the function **segregate**() which segregates the nodes in the linked list as asked in the problem statement and returns the head of the modified linked list. The **printing** is done **automatically** by the **driver code**.

Expected Time Complexity: O(N).
Expected Auxiliary Space: O(N).

Constraints:
1 <= N <= 10^3
CODE SECTION:-

```
Node *segregate(Node *head)
    {
        // Add code here
        Node *p = head;
        int z = 0, one = 0, two = 0;
        while (p)
        {
            if (p->data == 0)
                z++;
            if (p->data == 1)
                one++;
            if (p->data == 2)
                two++;
            p = p->next;
        }
        p = head;
        while (p)
        {
            if (z != 0)
            {
                p->data = 0;
                z--;
            }
            else if (z == 0 && one != 0)
            {
                p->data = 1;
                one--;
            }
            else
            {
                p->data = 2;
                two--;
            }

            p = p->next;
        }
        return head;
    }
```

SECOND:-

Segregate even and odd nodes in a Link List : : Medium

Given a link list of size N, modify the list such that all the even numbers appear before all the odd numbers in the modified list. The order of appearance of numbers within each segregation should be same as that in the original list.

NOTE: Don't create a new linked list, instead rearrange the provided one.

Example 1:

```
Input:
N = 7
Link List:
17 -> 15 -> 8 -> 9 -> 2 -> 4 -> 6 -> NULL

Output: 8 2 4 6 17 15 9

Explaination: 8,2,4,6 are the even numbers
so they appear first and 17,15,9 are odd
numbers that appear later.
```

Example 2:

```
Input:
N = 4
Link List:
1 -> 3 -> 5 -> 7

Output: 1 3 5 7

Explaination: There is no even number.
So ne need for modification.
```

Your Task:
You do not need to read input or print anything. Your task is to complete the function **divide()** which takes N and head of Link List as input parameters and returns the head of modified link list. Don't create a new linked list, instead rearrange the provided one.

Expected Time Complexity: O(N)
Expected Auxiliary Space: O(1)

Constraints:
1 ≤ N ≤ 10^5
1 ≤ Each element of the list ≤ 10^5

CODE SECTION:-

```
Node *divide(int N, Node *head)
   {
       // code here
       Node *o = new Node(-1);
       Node *e = new Node(-1);
       Node *even = e;
       Node *odd = o;
       Node *p = head;
       while (p)
       {
           if ((p->data % 2) == 0)
           {
               e->next = p;
               e = e->next;
           }
           else
           {
               // cout<<p->data<<" ";
               o->next = p;
               o = o->next;
           }
           p = p->next;
       }
       e->next = odd->next;
       o->next = NULL;
       return even->next;
   }
```

-: DONE FOR THE DAY :-