Practical No.6

Create Two separate linklist for even and odd terms from Inputted List

```
#include<stdio.h>
#include<stdlib.h>
typedef struct node
{
    struct node*next;
    struct node*pre;
    int data;
}Node;
void Display(Node *);
void EvenNumber(Node **,Node * ,Node **);
void OddNumber(Node **,Node * ,Node **);

int count;
int main()
{
     Node *node,*head=NULL,*tail=NULL,*ptr,*p;
    int co,count=0;
    printf("Enter Number of nodes to be created : ");
    scanf("%d",&co);
    while(co>0)
    {
        node = (Node*)malloc(sizeof(Node));
        printf("\nEnter the element : ");
        scanf("%d",&node->data);
        node->next=NULL;
        node->pre=NULL;
        if(head==NULL && tail==NULL)
        {
            head=node;
            tail=node;
            co--;
        }
        else
        {
            ptr=head,p=tail;
            while(ptr->next!=NULL && p->pre!=NULL)
            {
                 ptr=ptr->next;
                 p=p->pre;
            }
            ptr->next=node;
            node->pre=ptr;
            tail=node;
            co--;
        }
    }
Display(head);
```

```
Node *even_head=NULL,*even_tail=NULL;
EvenNumber(&even_head,head,&even_tail);
 printf("\n\t\tList Of Even Numbers\n");
 Display(even_head);
 Node *odd_head=NULL,*odd_tail=NULL;
 OddNumber(&odd_head,head,&odd_tail);
  printf("\n\t\tList of Odd numbers\n");
 Display(odd_head);
}
void EvenNumber(Node **even_head,Node *head,Node**even_tail)
{
    Node *ptr=*even_head,*p=*even_tail;
    while(head!=NULL)
    {
        int n=head->data;

        if(n%2==0)
            {
                Node* node = (Node*)malloc(sizeof(Node));
                node->next=NULL;
                node->pre=NULL;
                node->data=n;
                 if(*even_head==NULL && *even_tail==NULL)
                     {
                       *even_head=node;
                       *even_tail=node;
                     }
                else
                     {
                            ptr=*even_head,p=*even_tail;
                            while(ptr->next!=NULL && p->pre!=NULL)
                                {
                                    ptr=ptr->next;
                                    p=p->pre;
                                }
                            ptr->next=node;
                            node->pre=ptr;
                            *even_tail=node;
                    }
            }
            head=head->next;
    }

}
void OddNumber(Node **odd_head,Node *head,Node**odd_tail)
{
    Node *ptr=*odd_head,*p=*odd_tail;
    while(head!=NULL)
    {
        int n=head->data;

        if(n%2!=0)
            {
                Node* node = (Node*)malloc(sizeof(Node));
                node->next=NULL;
                node->pre=NULL;
                node->data=n;
                 if(*odd_head==NULL && *odd_tail==NULL)
                     {
                       *odd_head=node;
                       *odd_tail=node;
                     }
                else
                     {
                            ptr=*odd_head,p=*odd_tail;
                            while(ptr->next!=NULL && p->pre!=NULL)
                                {
                                    ptr=ptr->next;
                                    p=p->pre;
                                }
                            ptr->next=node;
                            node->pre=ptr;
                            *odd_tail=node;
                    }
            }
            head=head->next;
    }

}
void Display(Node *ptr)
{
    count = 0;
    printf("\nLIST : ");
    while(ptr!=NULL)
    {
        printf("%d  ",ptr->data);
        count++;
        ptr=ptr->next;
    }
    printf("\nNumber of Nodes : %d\n",count);
}
```