Flattening a linked list

Using recursion traverse till we reach the last linked list i.e when root==NULL || root→next==NULL return root this will be the base condition once we get the last list we can select last two list and perform merge on them.

```
Node* merge(Node* head1,Node* head2)
{

    Node* temp=new Node(0);
    Node* res=temp;
    while(head1!=NULL && head2!=NULL)
    {
        if(head1->data < head2->data)
        {
            temp->bottom=head1;
            temp=temp->bottom;
            head1=head1->bottom;
        }
        else
        {
            temp->bottom=head2;
            temp=temp->bottom;
            head2=head2->bottom;
        }
    }
    if(head1)
        temp->bottom=head1;
    if(head2)
        temp->bottom=head2;
    return res->bottom;
}
Node *flatten(Node *root)
{
   if(root==NULL || root->next==NULL)
   {
       return root;
   }
   root->next=flatten(root->next);
   root=merge(root,root->next);
   return root;
}
```

- Time Complexity : O(N*k)
- Space Complexity : O(N*K)