

```

1.  /**
2.   * Definition for singly-linked list.
3.   * class ListNode {
4.   *     public int val;
5.   *     public ListNode next;
6.   *     ListNode(int x) { val = x; next = null; }
7.   * }
8.   */
9.  public class Solution {
10.     public ListNode reverseBetween(ListNode head, int a, int b) {
11.         int size=getSize(head);
12.
13.         if(a==1 && b!=size){
14.             ListNode s=head;
15.             ListNode e=head;
16.             for(int i=1;i<b;i++) e=e.next;
17.             ListNode afb=e.next;
18.             e.next=null;
19.             head=reverseList(s);
20.             s.next=afb;
21.
22.             return head;
23.         }
24.         else if(b==size && a!=1){
25.             ListNode bFa=head;
26.             for(int i=1;i<a-1;i++) bFa=bFa.next;
27.             ListNode s=bFa.next;
28.             bFa.next=reverseList(s);
29.
30.             return head;
31.         }
32.         else if(a>1 && b<size){
33.             ListNode bFa=head;
34.             for(int i=1;i<a-1;i++) bFa=bFa.next;
35.             ListNode s=bFa.next;
36.
37.             ListNode e=head;
38.             for(int i=1;i<b;i++) e=e.next;
39.             ListNode afb=e.next;
40.             e.next=null;
41.
42.             bFa.next=reverseList(s);
43.             s.next=afb;
44.
45.             return head;
46.         }
47.     }
48.     else{
49.         return reverseList(head);
50.     }
51. }
52.

```

```
53.
54.
55.
56.
57.
58. public int getSize(ListNode h){
59.     ListNode t=h;
60.     int c=0;
61.     while(t!=null){
62.         c++;
63.         t=t.next;
64.     }
65.     return c;
66. }
67. public ListNode reverseList(ListNode A) {
68.     ListNode curr=A;
69.     ListNode previous=null;
70.     ListNode nex=null;
71.     while(curr!=null) {
72.         nex=curr.next;
73.         curr.next=previous;
74.         previous=curr;
75.         curr=nex;
76.     }
77.     return previous;
78. }
79. }
```

Problem Link: [reverse-link-list-ii InterviewBit](#)