Delete node in Doubly Linked List

Given a doubly linked list and a position. The task is to delete a node from given position in a doubly linked list.

Example 1:

```
Input: LinkedList = 1 <--> 3 <--> 4
x = 3
Output: 1 3  Explanation: After deleting the node at
position 3 (position starts from 1),
the linked list will be now as 1->3.
```

Example 2:

```
Input: LinkedList = 1 <--> 5 <--> 2 <--> 9
x = 1
Output: 5 2 9
```

Your Task:
The task is to complete the function **deleteNode**() which should delete the node at given position and return the head of the linkedlist.

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

Constraints:
2 <= size of the linked list <= 1000
1 <= x <= N

```python
#User function Template for python3

'''class Node:
    def __init__(self, data):
        self.data = data
        self.next = None
        self.prev = None
'''
class Solution:
    def deleteNode(self,head, x):
        # Code here
        if x == 1:
            head = head.next
```

```
            head.prev = None
            return head
        count = 1
        curr = head
        while count<x:
            curr = curr.next
            count+=1
        if curr.next == None:
            temp = curr.prev
            curr.prev = None
            temp.next = None
            return head
        else:
            last = curr.prev
            forward = curr.next
            curr.prev = None
            curr.next = None
            last.next = forward
            forward.prev = last
            return head
```