# Doubly linked list Insertion at given position

Given a doubly-linked list, a position **p,** and an integer **x.** The task is to add a new node with value **x** at the position just after **p$^{th}$** node in the doubly linked list.

**Example 1:**

```
Input: LinkedList: 2<->4<->5
p = 2, x = 6
Output: 2 4 5 6 Explanation: p = 2, and x = 6. So, 6 is
inserted after p, i.e, at position 3
(0-based indexing).
```

**Example 2:**

```
Input: LinkedList: 1<->2<->3<->4
p = 0, x = 44
Output: 1 44 2 3 4 Explanation: p = 0, and x = 44 . So, 44
is inserted after p, i.e, at position 1
(0-based indexing).
```

**Your Task:**
The task is to complete the function **addNode**() which head reference, position and data to be inserted as the arguments, with no return type.

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

**Constraints:**

1 <= N <= $10^4$
0 <= p < N

```
# Your task is to complete this function
# function should add a new node after the pth position
# function shouldn't print or return any data

'''
class Node:
    def __init__(self, data):
        self.data = data
        self.next = None
        self.prev = None
```

```
'''
#Function to insert a new node at given position in doubly linked list.
def addNode(head, p, data):
    # Code here
    if p==0 and head.next!=None:
        node = Node(data)
        node.next = head.next
        head.next = node
        node.prev = head
        node.next.prev = node
        return head
    count = 0
    curr = head
    last = None
    while count<p:
        last = curr
        curr = curr.next
        count = count+1
    if curr.next==None:
        node = Node(data)
        curr.next = node
        node.prev = curr
        node.next = None
        return head
    else:
        temp = curr.next
        node = Node(data)
        curr.next = node
        temp.prev = node
        node.prev = curr
        node.next = temp
    return head
```