Remove Duplicates from Sorted List II

Question: Given a sorted linked list, delete all nodes that have duplicate numbers, leaving only distinct numbers from the original list.

For example:

Given 1->2->3->3->4->4->5, return 1->2->5.

Given 1->1->1->2->3, return 2->3.

Solutions:

```
class ListNode:
    def __init__(self, x):
        self.val = x
        self.next = None


class Solution:
    # @param head, a ListNode
    # @return a ListNode
    def deleteDuplicates(self, head):
        if head == None: return None
        dummy = ListNode(10**10)
        dummy.next, head = head, dummy # add a dummy node
        pprev, prev, curr, dupFlag = head, head.next, head.next.next, False
        while True:
            if dupFlag == True:
                if curr == None:
                    pprev.next = None
                    break
```

```
            if prev.val != curr.val:
                pprev.next, prev, dupFlag = curr, curr, False
        else:
            if curr == None: break
            if prev.val == curr.val:
                dupFlag = True
            else:
                pprev, prev = pprev.next, prev.next
        curr = curr.next
    return head.next

  def printll(self, node):
    while node:
        print ( node.val )
        node = node.next

if __name__ == '__main__':
  ll1, ll1.next, ll1.next.next = ListNode(2), ListNode(2), ListNode(5)
  Solution().printll( Solution().deleteDuplicates(ll1) )
```