

**50) Insertion Sort List** Given the head of a singly linked list, sort the list using insertion sort, and return the sorted list's head.

```
CODE: class ListNode:
    def __init__(self, val=0, next=None):
        self.val = val
        self.next = next

    def insertionSortList(head):
        dummy = ListNode(float('-inf')) # Create a dummy node to serve as the head of the sorted list
        current = head # Start with the first node of the original list

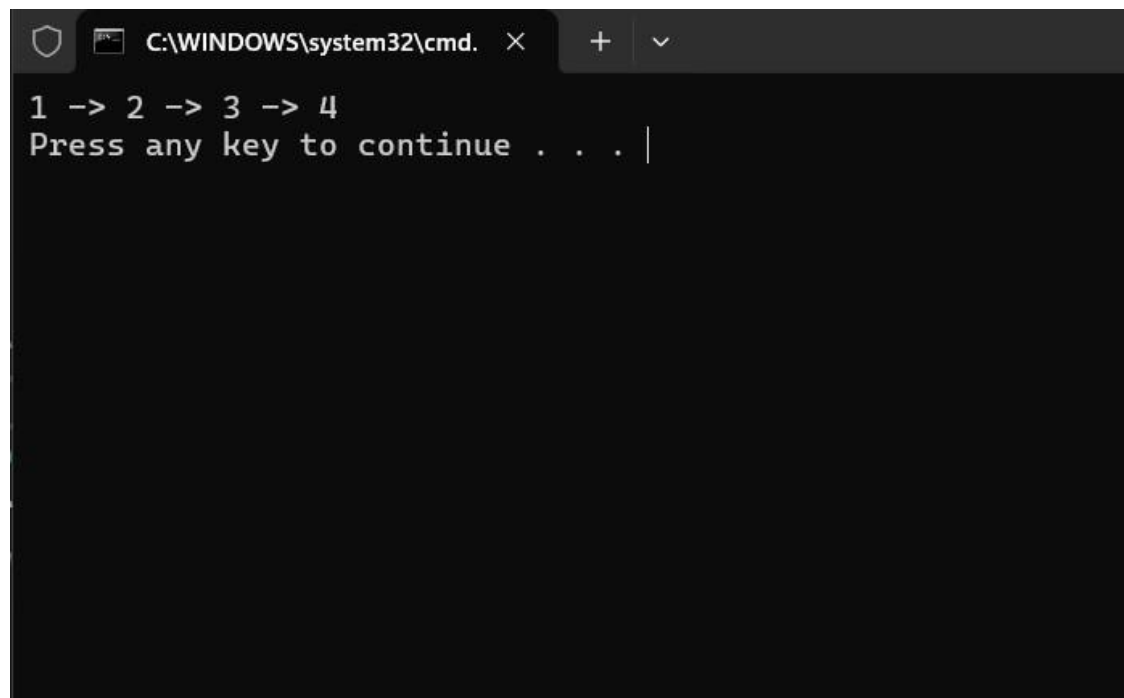
        # Traverse the original list and insert each node into the sorted list
        while current:
            prev, next_node = dummy, dummy.next
            while next_node and next_node.val < current.val:
                prev, next_node = next_node, next_node.next
            temp = current.next
            current.next = prev.next
            prev.next = current
            current = temp

        return dummy.next

    def printLinkedList(head):
        result = []
        while head:
            result.append(head.val)
            head = head.next
        print(" -> ".join(map(str, result)))

# Example usage:
head = ListNode(4)
head.next = ListNode(2)
head.next.next = ListNode(1)
head.next.next.next = ListNode(3)
sorted_head = insertionSortList(head)
printLinkedList(sorted_head)
```

OUTPUT:



A screenshot of a Windows command prompt window. The title bar shows the path 'C:\WINDOWS\system32\cmd.' and standard window controls. The command prompt displays the text '1 -> 2 -> 3 -> 4' on the first line and 'Press any key to continue . . . |' on the second line. The cursor is positioned at the end of the second line.

TIME COMPLEXITY :  **$O(n^2)$**