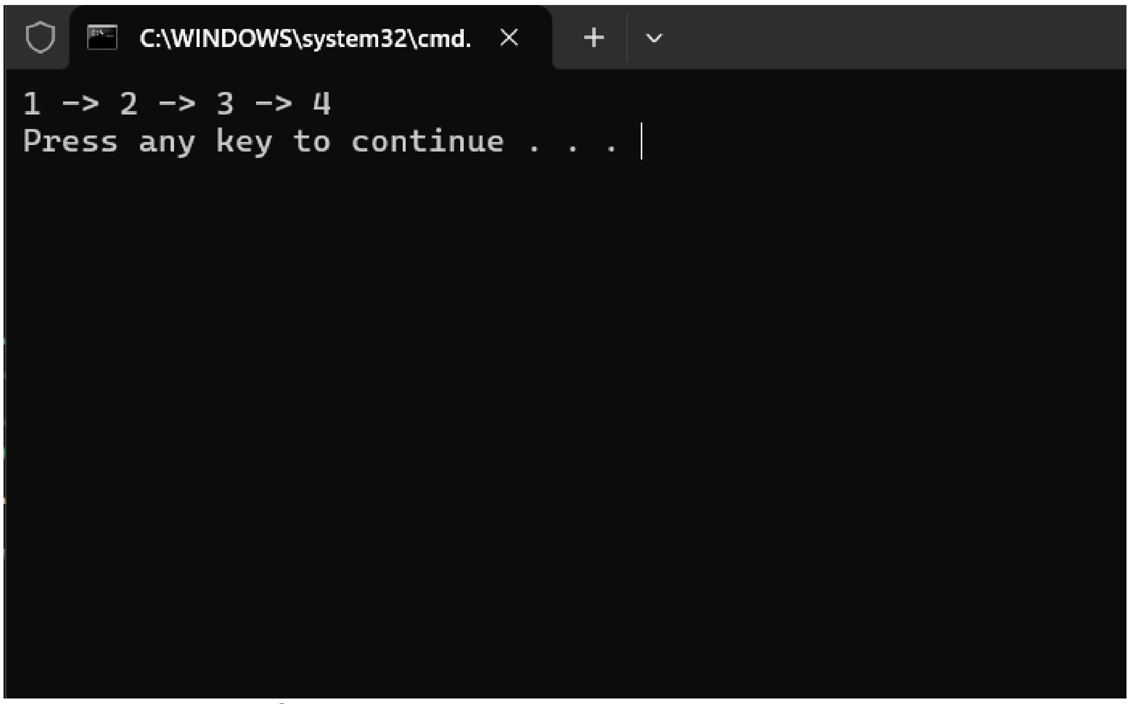
50) Insertion Sort List Given the head of a singly linked list, sort the list using insertion sort, and return the sorted list's head.

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 = next_node
        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:



```
1 -> 2 -> 3 -> 4
Press any key to continue . . .
```

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