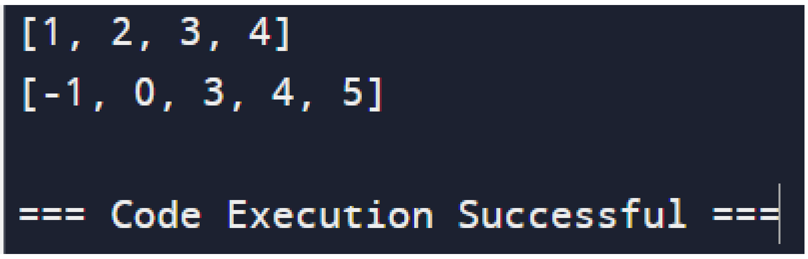
50. Insertion Sort List Given the head of a singly linked list, sort the list using insertion sort, and return the sorted list's head. The steps of the insertion sort algorithm: 1. Insertion sort iterates, consuming one input element each repetition and growing a sorted output list. 2. At each iteration, insertion sort removes one element from the input data, finds the location it belongs within the sorted list and inserts it there. 3. It repeats until no input elements remain. The following is a graphical example of the insertion sort algorithm. The partially sorted list (black) initially contains only the first element in the list. One element (red) is removed from the input data and inserted in-place into the sorted list with each iteration.

PROGRAM:-

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

def insertionSortList(head):
    if not head or not head.next:
        return head

    sorted_head = ListNode(0)  # Dummy node to simplify insertion logic
    sorted_head.next = head
    current = head
    while current and current.next:
        if current.val <= current.next.val:
            current = current.next
        else:
            # Find the position to insert current.next
            to_insert = current.next
            prev = sorted_head
            while prev.next.val < to_insert.val:
                prev = prev.next
            # Insert current.next to the correct position
            current.next = to_insert.next
            to_insert.next = prev.next
            prev.next = to_insert

    return sorted_head.next

# Helper function to create a linked list from a list of values.
def create_linked_list(arr):
    if not arr:
        return None
    head = ListNode(arr[0])
    current = head
    for val in arr[1:]:
        current.next = ListNode(val)
        current = current.next
    return head

# Helper function to print the linked list.
```

```
def print_linked_list(head):
    result = []
    while head:
        result.append(head.val)
        head = head.next
    print(result)

# Example usage:
head = create_linked_list([4, 2, 1, 3])
sorted_head = insertionSortList(head)
print_linked_list(sorted_head)  # Output: [1, 2, 3, 4]

# Another example:
head = create_linked_list([-1, 5, 3, 4, 0])
sorted_head = insertionSortList(head)
print_linked_list(sorted_head)  # Output: [-1, 0, 3, 4, 5]
```

OUTPUT:-



```
[1, 2, 3, 4]
[-1, 0, 3, 4, 5]

=== Code Execution Successful ===
```

TIME COMPLEXITY:-O(n^2)