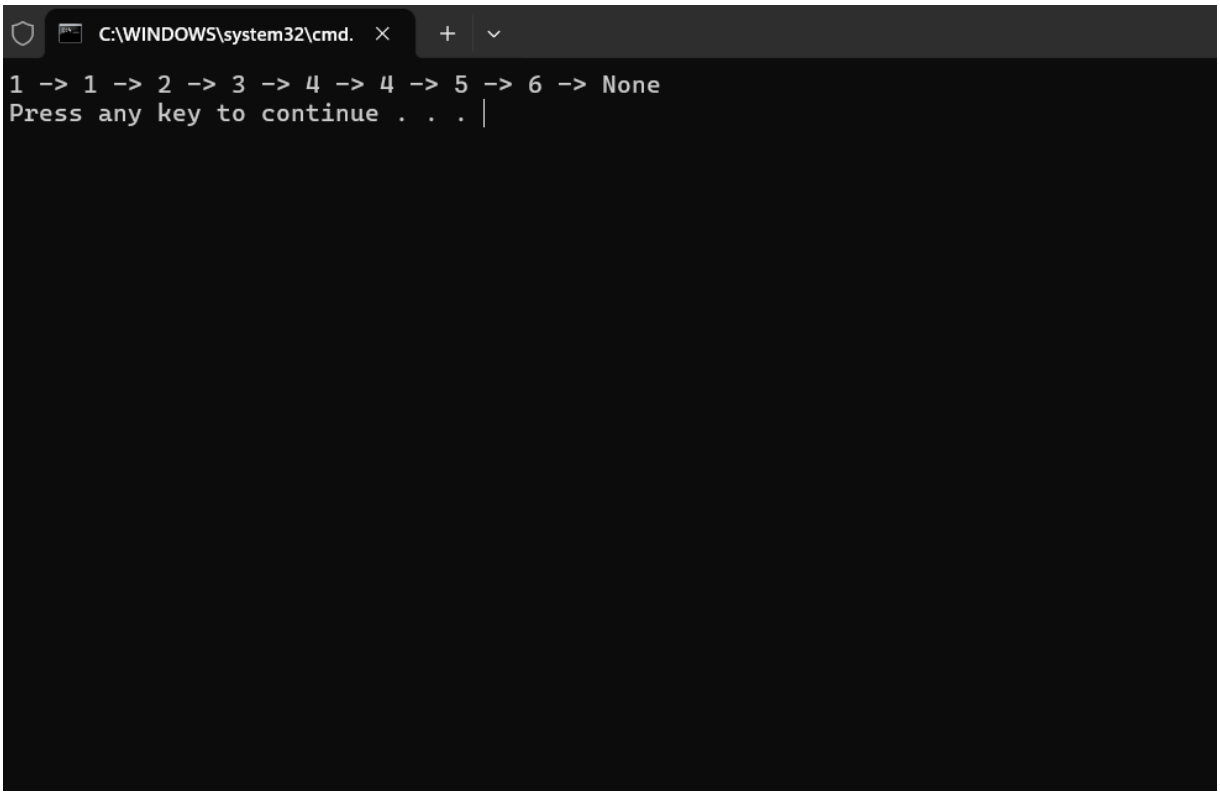
42) 2. Merge k Sorted Lists You are given an array of k linked-lists lists, each linked-list is sorted in ascending order. Merge all the linked-lists into one sorted linked-list and return it.

CODE:

```
from heapq import heappush, heappop

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

    def __lt__(self, other):
        return self.val < other.val

def mergeKLists(lists):
    min_heap = []

    # Initialize the heap with the head node of each list
    for l in lists:
        if l:
            heappush(min_heap, l)

    dummy = ListNode()
    current = dummy

    # Extract the smallest element from the heap and add it to the merged list
    while min_heap:
        node = heappop(min_heap)
        current.next = node
        current = current.next
        if node.next:
            heappush(min_heap, node.next)

    return dummy.next

# Helper function to create a linked list from a list
def create_linked_list(lst):
    dummy = ListNode()
    current = dummy
    for val in lst:
        current.next = ListNode(val)
        current = current.next
    return dummy.next

# Helper function to print a linked list
def print_linked_list(node):
    result = []
    while node:
        result.append(str(node.val))
        node = node.next
    print(" -> ".join(result) + " -> None")

# Example usage:
lists = [
    create_linked_list([1, 4, 5]),
    create_linked_list([1, 3, 4]),
    create_linked_list([2, 6])
]

merged_list = mergeKLists(lists)
print_linked_list(merged_list)
```

OUTPUT:



TIME COMPLEXITY : O((k+N)logk)