Question:

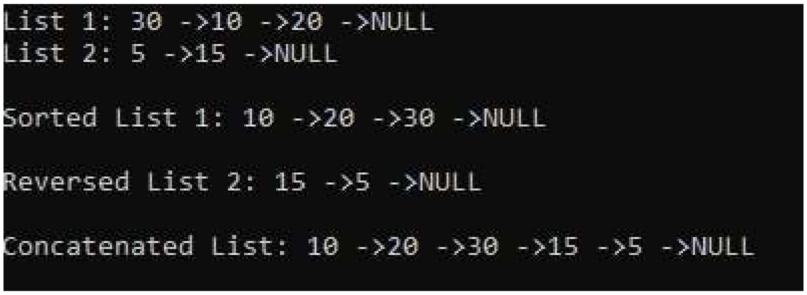
WAP to Implement Single Link List with following operations: Sort the linked list, Reverse the linked list, Concatenation of two linked lists.

Input:

```
#include <stdio.h>
#include <stdlib.h>

struct Node {
    int data;
    struct Node *next;
};

struct Node* createNode(int data) {
    struct Node* newNode = (struct Node*)malloc(sizeof(struct Node));
    newNode->data = data;
    newNode->next = NULL;
    return newNode;
}

void insertEnd(struct Node **head, int data) {
    struct Node *newNode = createNode(data);
    if (*head == NULL) {
        *head = newNode;
        return;
    }

    struct Node *temp = *head;
    while (temp->next != NULL)
        temp = temp->next;

    temp->next = newNode;
}

void display(struct Node *head) {
    while (head != NULL) {
        printf("%d ->", head->data);
        head = head->next;
    }
    printf("NULL\n");
}

void sortList(struct Node *head) {
    struct Node *i, *j;
    int temp;

    if (head == NULL) return;

    for (i = head; i->next != NULL; i = i->next) {
        for (j = i->next; j != NULL; j = j->next) {
            if (i->data > j->data) {
                temp = i->data;
                i->data = j->data;
                j->data = temp;
            }
        }
    }
}

void reverseList(struct Node **head) {
    struct Node *prev = NULL, *curr = *head, *next = NULL;

    while (curr != NULL) {
        next = curr->next;
        curr->next = prev;
        prev = curr;
        curr = next;
    }
    *head = prev;
}

struct Node* concatenate(struct Node *head1, struct Node *head2) {
    if (head1 == NULL) return head2;
    if (head2 == NULL) return head1;

    struct Node *temp = head1;
    while (temp->next != NULL)
        temp = temp->next;

    temp->next = head2;
    return head1;
}
```

```
int main() {
    struct Node *list1 = NULL, *list2 = NULL;

    insertEnd(&list1, 30);
    insertEnd(&list1, 10);
    insertEnd(&list1, 20);

    insertEnd(&list2, 5);
    insertEnd(&list2, 15);

    printf("List 1: ");
    display(list1);

    printf("List 2: ");
    display(list2);

    sortList(list1);
    printf("\nSorted List 1: ");
    display(list1);

    reverseList(&list2);
    printf("\nReversed List 2: ");
    display(list2);


    struct Node *mergedList = concatenate(list1, list2);
    printf("\nConcatenated List: ");
    display(mergedList);

    return 0;
}
```

Output:



```
List 1: 30 ->10 ->20 ->NULL
List 2: 5 ->15 ->NULL

Sorted List 1: 10 ->20 ->30 ->NULL

Reversed List 2: 15 ->5 ->NULL

Concatenated List: 10 ->20 ->30 ->15 ->5 ->NULL
```