6a. WAP to Implement Single Link List with following operations: Sort the linked list, Reverse the linked list, Concatenation of two linked lists.

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

// Structure for node
struct Node {
    int data;
    struct Node *next;
};

// Create a linked list
struct Node* createList() {
    struct Node *head = NULL, *temp, *newNode;
    int n, value;

    printf("Enter number of nodes: ");
    scanf("%d", &n);

    for (int i = 0; i < n; i++) {
        newNode = (struct Node*)malloc(sizeof(struct Node));
        printf("Enter data: ");
        scanf("%d", &value);
        newNode->data = value;
        newNode->next = NULL;

        if (head == NULL)
            head = newNode;
        else {
            temp = head;
            while (temp->next != NULL)
                temp = temp->next;
            temp->next = newNode;
```

```
        }
    }
    return head;
}

// Display the linked list
void display(struct Node *head) {
    if (head == NULL) {
        printf("List is empty\n");
        return;
    }
    while (head != NULL) {
        printf("%d ", head->data);
        head = head->next;
    }
    printf("\n");
}

// Sort the linked list (Bubble Sort)
struct Node* sortList(struct Node *head) {
    struct Node *i, *j;
    int temp;

    for (i = head; i != 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;
            }
        }
    }
```

```
    printf("List sorted\n");
    return head;
}

// Reverse the linked list
struct Node* 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;
    }
    printf("List reversed\n");
    return prev;
}

// Concatenate two linked lists
struct Node* concatenate(struct Node *head1, struct Node *head2) {
    if (head1 == NULL)
        return head2;

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

    temp->next = head2;
    printf("Lists concatenated\n");
    return head1;
}
```

```
// Main function
int main() {
    struct Node *list1 = NULL, *list2 = NULL;
    int choice;

    while (1) {
        printf("\nMenu:\n");
        printf("1. Create First List\n");
        printf("2. Create Second List\n");
        printf("3. Sort First List\n");
        printf("4. Reverse First List\n");
        printf("5. Concatenate Lists\n");
        printf("6. Display First List\n");
        printf("7. Exit\n");
        printf("Enter choice: ");
        scanf("%d", &choice);

        switch (choice) {
            case 1:
                list1 = createList();
                break;
            case 2:
                list2 = createList();
                break;
            case 3:
                list1 = sortList(list1);
                break;
            case 4:
                list1 = reverseList(list1);
                break;
            case 5:
                list1 = concatenate(list1, list2);
```

```
            break;
        case 6:
            display(list1);
            break;
        case 7:
            return 0;
        default:
            printf("Invalid choice\n");
      }
   }
}
```

```
Menu:
1. Create First List
2. Create Second List
3. Sort First List
4. Reverse First List
5. Concatenate Lists
6. Display First List
7. Exit
Enter choice: 1
Enter number of nodes: 4
Enter data: 10
Enter data: 20
Enter data: 30
Enter data: 40

Menu:
1. Create First List
2. Create Second List
3. Sort First List
4. Reverse First List
5. Concatenate Lists
6. Display First List
7. Exit
Enter choice: 2
Enter number of nodes: 4
Enter data: 40
Enter data: 30
Enter data: 20
Enter data: 10

Menu:
1. Create First List
2. Create Second List
3. Sort First List
4. Reverse First List
5. Concatenate Lists
6. Display First List
7. Exit
Enter choice: 3
List sorted

Menu:
1. Create First List
2. Create Second List
3. Sort First List
4. Reverse First List
5. Concatenate Lists
6. Display First List
7. Exit
Enter choice: 4
List reversed

Menu:
1. Create First List
2. Create Second List
3. Sort First List
4. Reverse First List
5. Concatenate Lists
6. Display First List
7. Exit
Enter choice: 5
Lists concatenated
```

```
Menu:
1. Create First List
2. Create Second List
3. Sort First List
4. Reverse First List
5. Concatenate Lists
6. Display First List
7. Exit
Enter choice: 6
40 30 20 10 40 30 20 10

Menu:
1. Create First List
2. Create Second List
3. Sort First List
4. Reverse First List
5. Concatenate Lists
6. Display First List
7. Exit
Enter choice: 7

Process returned 0 (0x0)   execution time : 41.800 s
Press any key to continue.
```