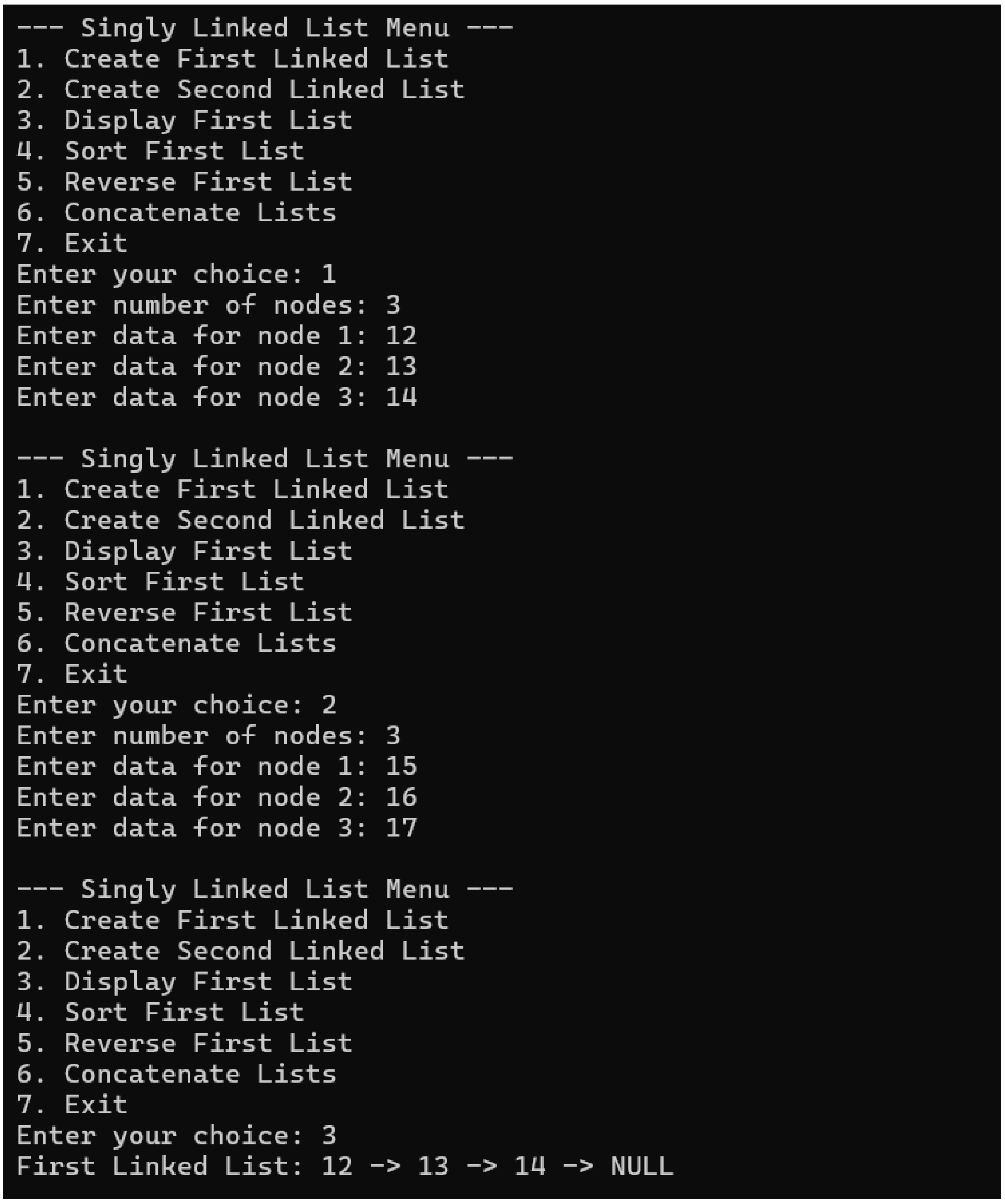
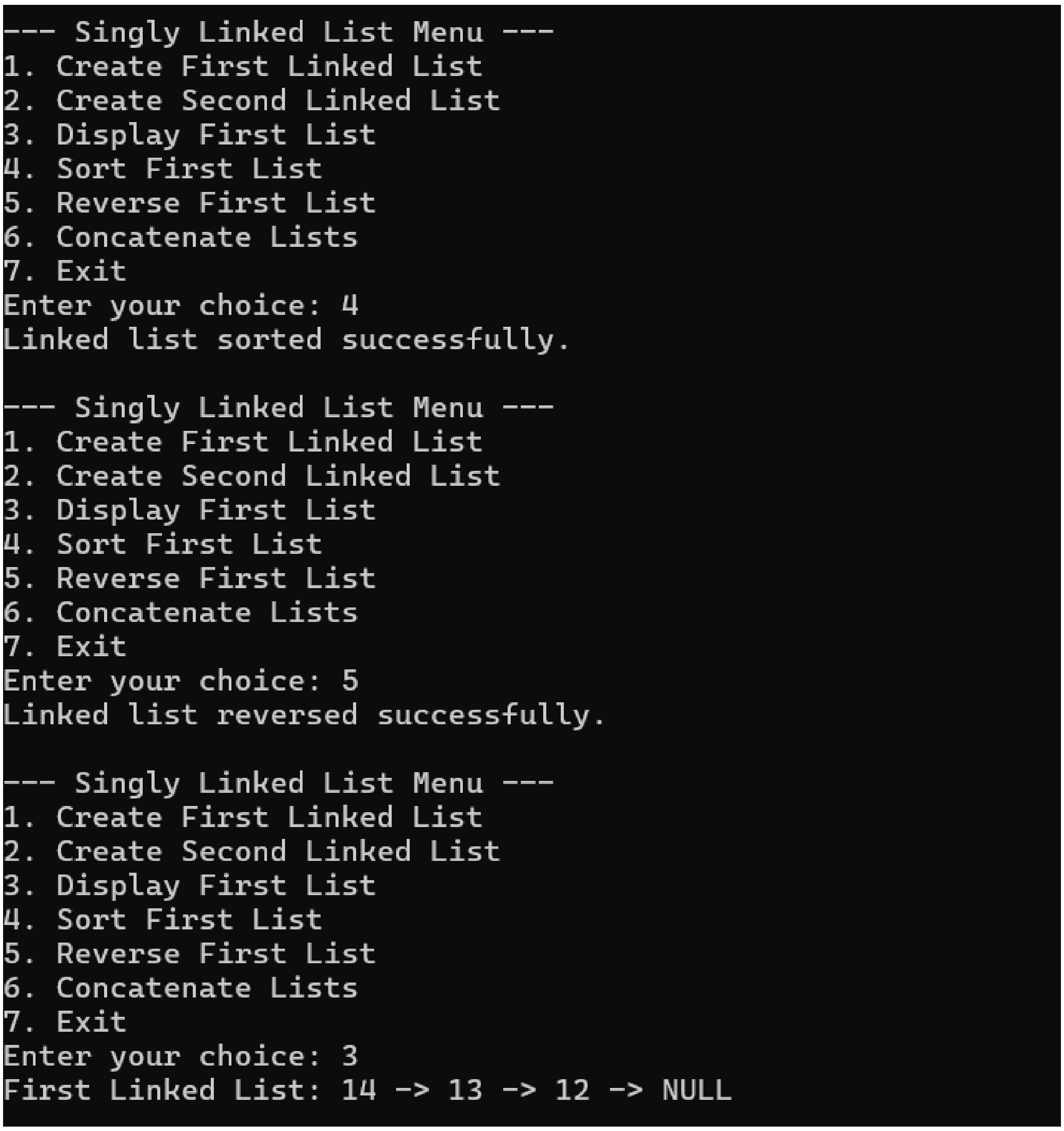
Lab program 6a:-

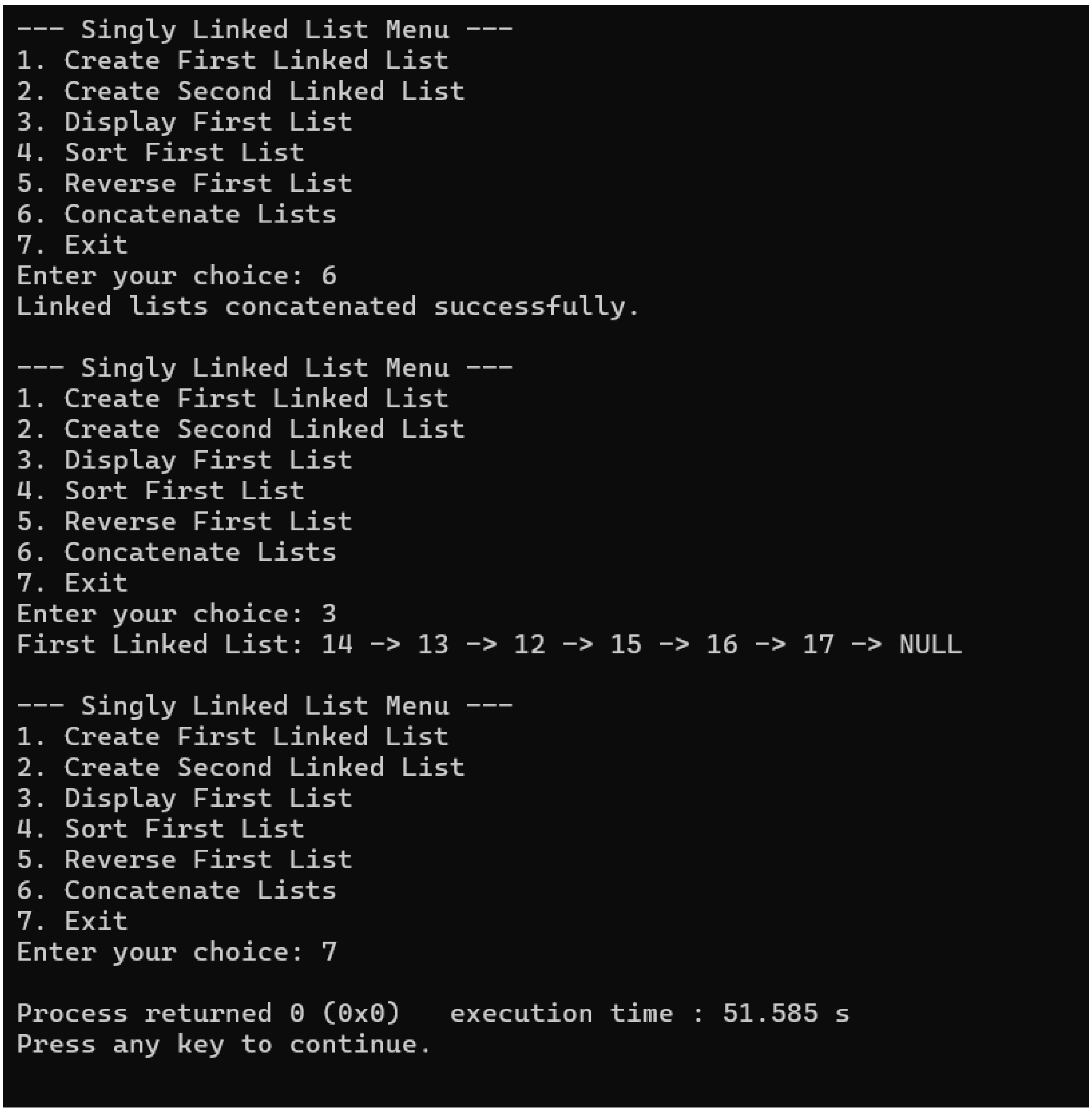
WAP to Implement Single Link List with following operations: Sort the linked list, Reverse the linked list, Concatenation of two linked lists.

```
#include <stdio.h>

#include <stdlib.h>


// Node structure

struct Node {

    int data;

    struct Node *next;

};


// Function to create a linked list

struct Node* createList() {

    struct Node *head = NULL, *temp = NULL, *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 for node %d: ", i + 1);

        scanf("%d", &value);


        newNode->data = value;

        newNode->next = NULL;


        if (head == NULL) {

            head = newNode;
            temp = newNode;
        } else {
            temp->next = newNode;
            temp = newNode;
        }
    }
    return head;
}

// Display linked list
void display(struct Node *head) {
    struct Node *temp = head;
    if (head == NULL) {
        printf("List is empty.\n");
        return;
    }

    while (temp != NULL) {
        printf("%d -> ", temp->data);
        temp = temp->next;
    }
    printf("NULL\n");
}

// Sort linked list (Bubble Sort)
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;
            }
        }
    }
    printf("Linked list sorted successfully.\n");
}

// Reverse linked list
struct Node* reverseList(struct Node *head) {
    struct Node *prev = NULL, *curr = head, *nextNode;

    while (curr != NULL) {
        nextNode = curr->next;
        curr->next = prev;
        prev = curr;
        curr = nextNode;
    }
    printf("Linked list reversed successfully.\n");
    return prev;
}

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

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

    temp->next = head2;
    printf("Linked lists concatenated successfully.\n");
    return head1;
}

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

    while (1) {
        printf("\n--- Singly Linked List Menu ---\n");
        printf("1. Create First Linked List\n");
        printf("2. Create Second Linked List\n");
        printf("3. Display First List\n");
        printf("4. Sort First List\n");
        printf("5. Reverse First List\n");
        printf("6. Concatenate Lists\n");
        printf("7. Exit\n");
        printf("Enter your choice: ");
        scanf("%d", &choice);
        switch (choice) {
            case 1:
                list1 = createList();
                break;
            case 2:
                list2 = createList();
                break;
            case 3:
                printf("First Linked List: ");
                display(list1);
                break;
            case 4:
                sortList(list1);
                break;
            case 5:
                list1 = reverseList(list1);
                break;
            case 6:
                list1 = concatenate(list1, list2);
                list2 = NULL;
                break;
            case 7:
                exit(0);
            default:
                printf("Invalid choice!\n");
        }
    }
    return 0;
}
```


```
--- Singly Linked List Menu ---
1. Create First Linked List
2. Create Second Linked List
3. Display First List
4. Sort First List
5. Reverse First List
6. Concatenate Lists
7. Exit
Enter your choice: 1
Enter number of nodes: 3
Enter data for node 1: 12
Enter data for node 2: 13
Enter data for node 3: 14

--- Singly Linked List Menu ---
1. Create First Linked List
2. Create Second Linked List
3. Display First List
4. Sort First List
5. Reverse First List
6. Concatenate Lists
7. Exit
Enter your choice: 2
Enter number of nodes: 3
Enter data for node 1: 15
Enter data for node 2: 16
Enter data for node 3: 17

--- Singly Linked List Menu ---
1. Create First Linked List
2. Create Second Linked List
3. Display First List
4. Sort First List
5. Reverse First List
6. Concatenate Lists
7. Exit
Enter your choice: 3
First Linked List: 12 -> 13 -> 14 -> NULL
```



```
--- Singly Linked List Menu ---
1. Create First Linked List
2. Create Second Linked List
3. Display First List
4. Sort First List
5. Reverse First List
6. Concatenate Lists
7. Exit
Enter your choice: 4
Linked list sorted successfully.

--- Singly Linked List Menu ---
1. Create First Linked List
2. Create Second Linked List
3. Display First List
4. Sort First List
5. Reverse First List
6. Concatenate Lists
7. Exit
Enter your choice: 5
Linked list reversed successfully.

--- Singly Linked List Menu ---
1. Create First Linked List
2. Create Second Linked List
3. Display First List
4. Sort First List
5. Reverse First List
6. Concatenate Lists
7. Exit
Enter your choice: 3
First Linked List: 14 -> 13 -> 12 -> NULL
```


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

--- Singly Linked List Menu ---
1. Create First Linked List
2. Create Second Linked List
3. Display First List
4. Sort First List
5. Reverse First List
6. Concatenate Lists
7. Exit
Enter your choice: 3
First Linked List: 14 -> 13 -> 12 -> 15 -> 16 -> 17 -> NULL

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

Process returned 0 (0x0)   execution time : 51.585 s
Press any key to continue.
```