Experiment 4

Student Name: Harsh Kumar **UID: 22BCS15754**
Branch: BE-CSE **Section/Group: 603_FL_IOT-B**
Semester: 5th **Date of Performance:12/08/2024**
Subject Name: AP **Subject Code: 22CSP-314**

1. **Aim:** Write a program to reverse a singly linked list.

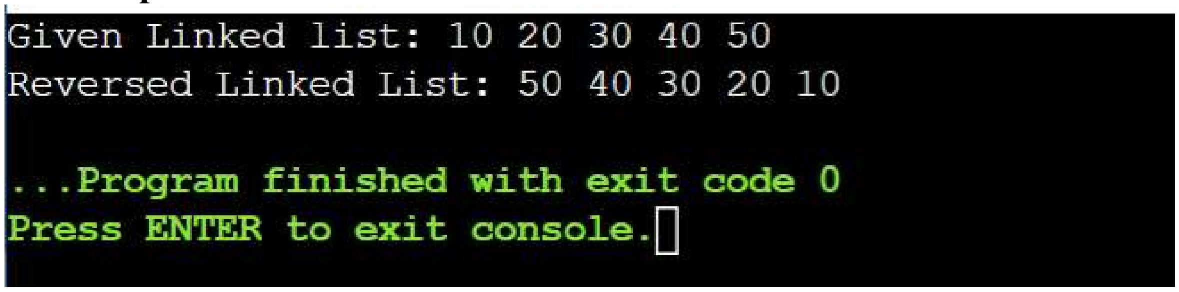
2. **Objective:** The objective of this program is to reverse the linked list by changing the links between nodes.

3. **Implementation/Code:**

```
#include <iostream>
using namespace std;

class Node {
public:
    int data;
    Node* next;
    Node(int new_data) {
        data = new_data;
        next = nullptr;
    }
};
// Given the head of a list, reverse the list and return the
// head of reversed list
Node* reverseList(Node* head) {

      // Initialize three pointers: curr, prev and next
    Node *curr = head, *prev = nullptr, *next;

      // Traverse all the nodes of Linked List
    while (curr != nullptr) {

        // Store next
        next = curr->next
```

```
            // Reverse current node's next pointer
            curr->next = prev;
            // Move pointers one position ahead
            prev = curr;
            curr = next;
        }
          // Return the head of reversed linked list
        return prev;
    }
    // This function prints the contents
    // of the linked list starting from the head
    void printList(Node* node) {
        while (node != nullptr) {
            cout << " " << node->data;
            node = node->next;
        }
    }
    // Driver code
        int main() {

        // Create a hard-coded linked list:
        // 1 -> 2 -> 3 -> 4 -> 5
        Node* head = new Node(10);
        head->next = new Node(20);
        head->next->next = new Node(30);
        head->next->next->next = new Node(40);
        head->next->next->next->next = new Node(50);

          // Print the original linked list
        cout << "Given Linked list:";
        printList(head);

          // Function call to return the reversed list
        head = reverseList(head);

          // Print the reversed linked list
        cout << "\nReversed Linked List:";
        printList(head);

        return 0;
    }
```

4. Output:



```
Given Linked list: 10 20 30 40 50
Reversed Linked List: 50 40 30 20 10

...Program finished with exit code 0
Press ENTER to exit console.
```

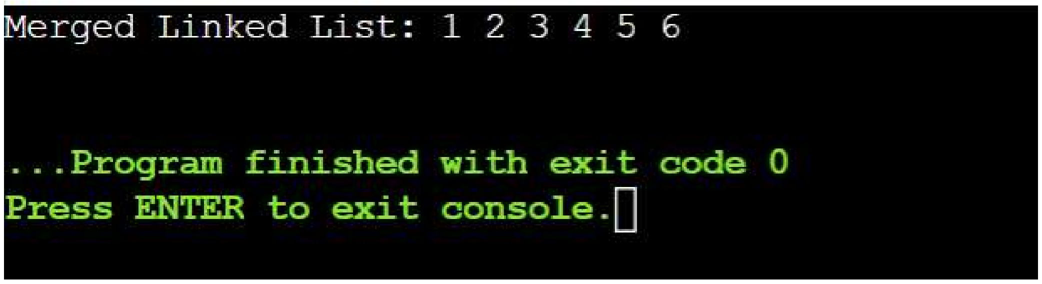
1.(ii) Aim: Write a program to merge two sorted singly linked list.

2.(ii) Objective: The objective of this program is to merge two sorted singly linked lists into a single sorted linked list.This is achieved by comparing the nodes of the two lists and systematically linking them in the correct sequence, resulting in a merged list that includes all elements from both input lists in sorted order.

3.(ii) Implementation/Code:

```
#include <iostream>
using namespace std;
struct Node {
    int data;
    Node* next;
    Node(int x) : data(x), next(nullptr) {}
};
Node* mergeTwoLists(Node* l1, Node* l2) {
    if (!l1) return l2;
    if (!l2) return l1;
    Node* head = nullptr;
    if (l1->data <= l2->data) {
        head = l1;
        head->next = mergeTwoLists(l1->next, l2);
    } else {
        head = l2;
        head->next = mergeTwoLists(l1, l2->next);
    }
    return head;
}
// Function to print the linked list
void printList(Node* node) {
    while (node) {
        cout << node->data << " ";
        node = node->next;
    }
    cout << endl;
}
// Test the mergeTwoLists function
int main() {
    Node* l1 = new Node(1);
    l1->next = new Node(3);
    l1->next->next = new Node(5);

    Node* l2 = new Node(2);
    l2->next = new Node(4);
    l2->next->next = new Node(6);

    Node* mergedList = mergeTwoLists(l1, l2);
    cout << "Merged Linked List: ";
    printList(mergedList);
    return 0;
}
```

4.(ii)Output:



```
Merged Linked List: 1 2 3 4 5 6


...Program finished with exit code 0
Press ENTER to exit console.
```