```cpp
#include<bits/stdc++.h>
using namespace std;

struct node
{
    int data;
    node* next;

    node(int value)
    {
        data = value;
        next = nullptr;
    }
};

class LinkedList
{
public:
    node* head;

    LinkedList()
    {
        head = nullptr;
    }

    void append(int value)
    {
        node* newnode = new node(value);
        if(head == nullptr)
            head = newnode;
        else
        {
            node* temp = head;
            while(temp->next != nullptr)
            {
                temp = temp->next;
            }
            temp->next = newnode;
        }
    }

    void display()
    {
        node* temp = head;
        while(temp != nullptr)
        {
            cout << temp->data << " ";
            temp = temp->next;
        }
        cout << endl;
    }

    // Function to split the list into front and back halves
    void splitList(LinkedList& frontList, LinkedList& backList)
    {
        if (head == nullptr)  // Empty list
            return;

        // Step 1: Find the total length of the list
        int length = 0;
        node* temp = head;
        while (temp != nullptr)
        {
            length++;
            temp = temp->next;
        }
```

```
        // Step 2: Find the splitting point (middle of the list)
        int splitPoint = (length + 1) / 2; // Front half gets the extra element in case of odd number

        // Step 3: Split the list into front and back halves
        frontList.head = head;   // Front list starts from the head
        temp = head;
        for (int i = 1; i < splitPoint; i++)  // Traverse to the split point
        {
            temp = temp->next;
        }

        backList.head = temp->next;  // Back list starts from the next node
        temp->next = nullptr;        // Split the list by breaking the link
    }
};

int main()
{
    LinkedList List;

    // Append elements to the list
    List.append(2);
    List.append(3);
    List.append(5);
    List.append(7);
    List.append(11);

    cout << "Original LinkedList: ";
    List.display();

    LinkedList frontList;
    LinkedList backList;

    // Split the list into front and back halves
    List.splitList(frontList, backList);

    cout << "Front half: ";
    frontList.display();

    cout << "Back half: ";
    backList.display();

    return 0;
}
```