Single Linked List Implementation

Node definition:

```
class Node {
public:
    int data;
    Node* next;
    Node(int value) : data(value), next(nullptr) {}
};
```

Create a linked list:

```
class LinkedList {
public:
    Node* head;
    LinkedList() : head(nullptr) {}

    void append(int data) {
        Node* new_node = new Node(data);
        if (head == nullptr) {
            head = new_node;
            return;
        }
        Node* current = head;
        while (current->next) {
            current = current->next;
        }
        current->next = new_node;
    }
};

// Example usage:
int main() {
    LinkedList llist;
    llist.append(5);
    llist.append(10);
    llist.append(15);
    return 0;
}
```

Printing a linked list:

```
void printLinkedList(Node* head) {
    Node* current = head;
    while (current) {
        cout << current->data << " -> ";
        current = current->next;
    }
    cout << "nullptr" << endl;
}

// Example usage:
printLinkedList(llist.head);
```

Finding an element in a linked list:

```
bool findElement(Node* head, int target) {
    Node* current = head;
    while (current) {
        if (current->data == target) {
            return true;
        }
        current = current->next;
    }
    return false;
}

// Example usage:
cout << findElement(llist.head, 10) << endl;  // Output: 1 (true)
cout << findElement(llist.head, 20) << endl;  // Output: 0 (false)
```

Deleting an element in a linked list:

```
void deleteElement(Node*& head, int target) {
    if (head == nullptr) {
        return;
    }

    if (head->data == target) {
        Node* temp = head;
        head = head->next;
        delete temp;
        return;
    }
```

```
    Node* current = head;
    while (current->next) {
        if (current->next->data == target) {
            Node* temp = current->next;
            current->next = current->next->next;
            delete temp;
            return;
        }
        current = current->next;
    }
}

// Example usage:
deleteElement(llist.head, 10);
printLinkedList(llist.head);
```