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

// Node structure for doubly linked list
struct Node {
    int data;
    Node* next;
    Node* prev;
};

// Function to insert a new node at the end of the list
void append(Node*& head, int data) {
    Node* newNode = new Node();
    newNode->data = data;
    newNode->next = nullptr;

    if (head == nullptr) {
        newNode->prev = nullptr;
        head = newNode;
        return;
    }

    Node* temp = head;
    while (temp->next != nullptr)
        temp = temp->next;

    temp->next = newNode;
    newNode->prev = temp;
}

// Function to display the binary number stored in the doubly linked list
void display(Node* head) {
    while (head != nullptr) {
        cout << head->data;
        head = head->next;
    }
    cout << endl;
}

// Function to compute 1's complement of a binary number
Node* onesComplement(Node* head) {
    Node* result = nullptr;
    Node* temp = head;
    while (temp != nullptr) {
```

```
        append(result, temp->data == 0 ? 1 : 0);
        temp = temp->next;
    }
    return result;
}

// Function to compute 2's complement of a binary number
Node* twosComplement(Node* head) {
    Node* onesComp = onesComplement(head);
    Node* temp = onesComp;

    // Add 1 to the least significant bit
    bool carry = true;
    while (temp != nullptr) {
        if (temp->data == 1 && carry) {
            temp->data = 0;
            carry = true;
        } else {
            temp->data = temp->data + (carry ? 1 : 0);
            carry = false;
        }
        temp = temp->next;
    }

    if (carry)
        append(onesComp, 1);

    return onesComp;
}

// Function to add two binary numbers
Node* addBinary(Node* head1, Node* head2) {
    Node* result = nullptr;
    Node* temp1 = head1;
    Node* temp2 = head2;
    int carry = 0;

    while (temp1 != nullptr || temp2 != nullptr || carry) {
        int sum = carry;
        if (temp1 != nullptr) {
            sum += temp1->data;
            temp1 = temp1->next;
        }
        if (temp2 != nullptr) {
```

```
            sum += temp2->data;
            temp2 = temp2->next;
        }
        append(result, sum % 2);
        carry = sum / 2;
    }

    return result;
}

int main() {
    Node* binary1 = nullptr;
    Node* binary2 = nullptr;

    // Example binary numbers
    append(binary1, 1);
    append(binary1, 0);
    append(binary1, 1);
    append(binary1, 0);

    append(binary2, 1);
    append(binary2, 1);
    append(binary2, 0);
    append(binary2, 1);

    cout << "Binary 1: ";
    display(binary1);

    cout << "Binary 1's Complement: ";
    Node* onesComp = onesComplement(binary1);
    display(onesComp);

    cout << "Binary 1's 2's Complement: ";
    Node* twosComp = twosComplement(binary1);
    display(twosComp);

    cout << "Binary 2: ";
    display(binary2);

    cout << "Sum of Binary 1 and Binary 2: ";
    Node* sum = addBinary(binary1, binary2);
    display(sum);

    return 0;
```

```
}
```