```
/*Implement circular linked list and perform operations
on it. */
#include <iostream>
using namespace std;

struct Node {
    int data;
    Node* next;
};

class CircularLinkedList {
    private:
        Node* head;
    public:
        CircularLinkedList() {
            head = NULL;
        }

        void insertAtEnd(int data) {
            Node* newNode = new Node();
            newNode->data = data;
            newNode->next = head;
            Node* temp = head;
            if (head != NULL) {
                while(temp->next != head) {
                    temp = temp->next;
                }
                temp->next = newNode;
            } else {
                newNode->next = newNode;
            }
            head = newNode;
        }

        void insertAtBeginning(int data) {
            Node* newNode = new Node();
            newNode->data = data;
            newNode->next = head;
            Node* temp = head;
            if (head != NULL) {
                while(temp->next != head) {
                    temp = temp->next;
                }
                temp->next = newNode;
            } else {
                newNode->next = newNode;
            }
            head = newNode;
        }

        void deleteAtEnd() {
            if (head == NULL) {
                return;
            }
            Node* temp = head;
            if (temp->next == head) {
                head = NULL;
                delete temp;
                return;
```

```
            }
            while(temp->next->next != head) {
                temp = temp->next;
            }
            Node* toDelete = temp->next;
            temp->next = head;
            delete toDelete;
        }

        void deleteAtBeginning() {
            if (head == NULL) {
                return;
            }
            Node* temp = head;
            if (temp->next == head) {
                head = NULL;
                delete temp;
                return;
            }
            while(temp->next != head) {
                temp = temp->next;
            }
            Node* toDelete = head;
            head = head->next;
            temp->next = head;
            delete toDelete;
        }

        void display() {
            Node* temp = head;
            if (head != NULL) {
                do {
                    cout << temp->data << " ";
                    temp = temp->next;
                } while(temp != head);
            }
        }
};

int main() {
    CircularLinkedList cll;
    cll.insertAtEnd(1);
    cll.insertAtEnd(2);
    cll.insertAtEnd(3);
    cll.insertAtBeginning(0);
    cll.deleteAtEnd();
    cll.deleteAtBeginning();
    cll.display();
    return 0;
}
```