# [Create] Circular Linked List

```
public class MainClass {
    public static void main(String[] args) {
        CircularLinkedList list = new CircularLinkedList();
        list.addNode(0);
        list.addNode(1);
        list.addNode(2);
        list.addNode(3);
        list.addNode(4);
        list.addNode(5);

        list.printNodes();

        list.removeNode(2);
        list.printNodes();

        list.addFirst(-1);
        list.printNodes();

        list.addLast(6);
        list.printNodes();

        list.removeFirst();
        list.removeLast();
        list.printNodes();
    }
}

class Node {
    public Node nextNode;

    public int value;

    public Node(int value) {
        this.value = value;
        this.nextNode = null;
    }
}

class CircularLinkedList {
```

```
Node head;
Node tail;

public void addNode(int value) {
    Node node = new Node(value);
    if (head == null) {
        head = node;
        tail = node;
    } else {
        tail.nextNode = node;
        tail = node;
        node.nextNode = head;
    }
}

public boolean removeNode(int value) {
    Node temp = head;
    Node prev = null;

    if (temp == null) {
        return false;
    }

    // if value is at the top.
    if (temp != null && temp.value == value) {
        head = head.nextNode;
        return true;
    }

    // Search value
    while (temp != null && temp.value != value) {
        prev = temp;
        temp = temp.nextNode;
    }
    // Result not found.
    if (temp == null) {
        return false;
    }
    // Implement next node.
    prev.nextNode = temp.nextNode;
    return true;
}
```

```
public void printNodes() {
    Node temp = head;
    System.out.print("[");

    // No item available.
    if (head == null) {
        return;
    }

    do {
        if (temp.nextNode != null) {
            System.out.print(temp.value + ", ");
        } else {
            System.out.print(temp.value);
        }
        temp = temp.nextNode;
    } while (temp != head);
    System.out.print("]");
    System.out.println();
}

public void addFirst(int value) {
    Node node = new Node(value);
    if (head == null) {
        head = node;
        tail = node;
    } else {
        node.nextNode = head;
        head = node;
        tail.nextNode = head;
    }
}

public void addLast(int value) {
    Node node = new Node(value);
    if (head == null) {
        head = node;
        tail = node;
    } else {
        tail.nextNode = node;
        node.nextNode = head;
        tail = node;
    }
```

```
    }

    public void removeFirst() {
        if (head == null) {
            return;
        }
        head = head.nextNode;
        tail.nextNode = head;
    }

    public void removeLast() {
        Node temp = head;
        Node prev = null;
        if (head == null) {
            return;
        }
        do {
            prev = temp;
            temp = temp.nextNode;
        } while (temp.nextNode != head);

        prev.nextNode = head;
        tail = prev;
    }
}
```