Circular LinkedList

```
class Node:
    def __init__(self, data):
        self.data = data
        self.next = None

class CircularLinkedList:
    def __init__(self):
        self.head = None

    def insert_at_front(self, data):
        new_node = Node(data)
        if not self.head:
            self.head = new_node
            new_node.next = self.head
        else:
            current = self.head
            while current.next != self.head:
                current = current.next
            new_node.next = self.head
            current.next = new_node
            self.head = new_node

    def insert_at_back(self, data):
        new_node = Node(data)
        if not self.head:
            self.head = new_node
            new_node.next = self.head
        else:
            current = self.head
            while current.next != self.head:
                current = current.next
            current.next = new_node
            new_node.next = self.head

    def delete_from_front(self):
        if not self.head:
            print("The list is empty.")
            return
        if self.head.next == self.head:
            self.head = None
        else:
```

```
            current = self.head
            while current.next != self.head:
                current = current.next
            current.next = self.head.next
            self.head = self.head.next

    def delete_from_back(self):
        if not self.head:
            print("The list is empty.")
            return
        if self.head.next == self.head:
            self.head = None
        else:
            current = self.head
            while current.next.next != self.head:
                current = current.next
            current.next = self.head

    def display(self):
        if not self.head:
            print("The list is empty.")
            return
        current = self.head
        nodes = []
        while True:
            nodes.append(str(current.data))
            current = current.next
            if current == self.head:
                break
        print(" -> ".join(nodes) + " -> (head)")

cll = CircularLinkedList()
cll.insert_at_back(15)
cll.insert_at_back(25)
cll.insert_at_back(35)
print("Circular Linked List after insertion at back:")
cll.display()

cll.insert_at_front(5)
print("\nCircular Linked List after insertion at front:")
cll.display()

cll.delete_from_front()
print("\nCircular Linked List after deletion from front:")
```

```
cll.display()

cll.delete_from_back()
print("\nCircular Linked List after deletion from back:")
cll.display()
```

Output

```
Circular Linked List after insertion at back:
15 -> 25 -> 35 -> (head)

Circular Linked List after insertion at front:
5 -> 15 -> 25 -> 35 -> (head)

Circular Linked List after deletion from front:
15 -> 25 -> 35 -> (head)

Circular Linked List after deletion from back:
15 -> 25 -> (head)
```