**الشيفرة:**

class Node {

int data;

Node next;

public Node(int data) {

this.data = data;

this.next = null;

}

}

class SingularlyLinkedList {

Node head;

public SingularlyLinkedList() {

head = null;

}

// طريقة rotate التي تحاكي addLast(removeFirst())

public void rotate() {

if (head == null || head.next == null) {

// إذا كانت القائمة فارغة أو تحتوي على عقدة واحدة فقط، لا حاجة للقيام بشيء

return;

}

Node current = head;

// البحث عن العقدة الأخيرة

while (current.next != null) {

current = current.next;

}

// نقل العقدة الأولى إلى آخر القائمة

Node firstNode = head;

head = head.next; // جعل الرأس يشير إلى العقدة التالية

// جعل العقدة الأخيرة تشير إلى العقدة الأولى

current.next = firstNode;

firstNode.next = null; // التأكد من أن العقدة الأولى تشير إلى null بعد نقلها

}

// يمكنك إضافة طرق أخرى مثل الإدخال، والحذف، إلخ هنا

}

**الشرح:**

1. **التحقق من الحالة الخاصة**:
   * إذا كانت القائمة فارغة (head == null) أو تحتوي على عقدة واحدة فقط (head.next == null)، فلا نحتاج إلى التدوير، لأن التدوير في هذه الحالات لن يؤدي إلى أي تغيير.
2. **البحث عن العقدة الأخيرة**:
   * نقوم بالانتقال عبر القائمة حتى نجد العقدة الأخيرة (العقدة التي لا تحتوي على عقدة تالية).
3. **نقل العقدة الأولى إلى آخر القائمة**:
   * نقوم بتخزين العقدة الأولى (head)، ثم نغير head ليشير إلى العقدة التالية.
   * نقوم بتوصيل العقدة الأخيرة إلى العقدة الأولى عن طريق تعديل الإشارة next للعقدة الأخيرة.
   * أخيرًا، نعدل الإشارة next للعقدة الأولى لتكون null لأننا نريد أن تكون العقدة الأولى هي الأخيرة الآن.