```
class Node<S> {
    S data;
    Node<S> next;

    public Node(S data) {
        this.data = data;
        this.next = null;
    }
}

class SinglyLinkedList<S> {
    private Node<S> head;
    private int size;

    public SinglyLinkedList() {
        this.head = null;
        this.size = 0;
    }

    public boolean isEmpty() {
        return size == 0;
    }

    public int size() {
        return size;
    }

    public void addFirst(S data) {
        Node<S> newNode = new Node<>(data);
```

```
                newNode.next = head;
                head = newNode;
                size++;
            {

        public void addLast(S data) {
    Node<S> newNode = new Node<>(data);
                if (isEmpty()) {
                    head = newNode;
                else { {
                Node<S> current = head;
            while (current.next != null) {
                current = current.next;
                    {
                current.next = newNode;
                    {
                size++;
                {

        public void removeFirst() {
                if (isEmpty()) {
القائمة فارغة، لا يمكن إزالة أي عنصر.")؛System.out.println("
                    return;
                    {
                head = head.next;
                    size--;
                {

        public void printList() {
```

```
        Node<S> current = head;
        while (current != null) {
            System.out.print(current.data + " ");
            current = current.next;
        }
        System.out.println();
    }

    // تنفيذ دالة equals للمقارنة بين قائمتين مترابطتين
    @Override
    public boolean equals(Object obj) {
        if (this == obj) {
            return true;
        }
        if (obj == null || getClass() != obj.getClass()) {
            return false;
        }
        SinglyLinkedList<?> otherList = (SinglyLinkedList<?>) obj;
        if (this.size != otherList.size) {
            return false;
        }
        Node<S> currentThis = this.head;
        Node<?> currentOther = otherList.head;
        while (currentThis != null) {
```

```
if (!currentThis.data.equals(currentOther.data)) {
    return false; // إذا كانت أي قيمة مختلفة، القائمتان ليستا متساويتين.
{
currentThis = currentThis.next;
currentOther = currentOther.next;
{

return true; // إذا تم مقارنة جميع العناصر وكانت متساوية.
{
{

public class Main {
public static void main(String[] args) {
SinglyLinkedList<Integer> list1 = new SinglyLinkedList<>();
SinglyLinkedList<Integer> list2 = new SinglyLinkedList<>();
SinglyLinkedList<Integer> list3 = new SinglyLinkedList<>();

list1.addFirst(10);
list1.addFirst(20);
list1.addFirst(30);

list2.addFirst(10);
list2.addFirst(20);
list2.addFirst(30);

list3.addFirst(10);
list3.addFirst(20);

System.out.println("هل list1 متساوية مع list2؟ " + list1.equals(list2)); // يجب أن تكون true
```

```
"System.out.println("هل list1 متساوية مع list3؟ " + list1.equals(list3)); // يجب أن تكون false
    }
}
```