import java.util.Collection;

import java.util.Iterator;

import java.util.List;

import java.util.ListIterator;

public class LinkedList implements List {

Node head;

static int *size*;

Node tail;

public LinkedList() {

head = new Node(null);

*size* = 0;

tail = head;

}

class Node {

Object value;

Node next;

public Node(Object value) {

this.value = value;

}

public Node(Object value, Node next) {

this.value = value;

this.next = next;

}

}

@Override

public int size() {

return *size*;

}

@Override

public boolean isEmpty() {

return *size* == 0;

}

@Override

public boolean contains(Object o) {

return false;

}

@Override

public Iterator iterator() {

return null;

}

@Override

public Object[] toArray() {

return new Object[0];

}

@Override

public boolean add(Object o) {

Node node = new Node(o);

tail.next = node;

tail = node;

*size*++;

return true;

}

@Override

public boolean remove(Object o) {

return false;

}

@Override

public boolean addAll(Collection c) {

return false;

}

@Override

public boolean addAll(int index, Collection c) {

return false;

}

@Override

public void clear() {

}

@Override

public Object get(int index) {

return null;

}

@Override

public Object set(int index, Object element) {

return null;

}

@Override

public void add(int index, Object element) {

Node currNode = head;

for (int i= 0; i<index - 1; i++) {

currNode = currNode.next;

}

Node node = new Node(element);

node.next = currNode.next;

currNode.next = node;

*size*++;

}

@Override

public Object remove(int index) {

return null;

}

@Override

public int indexOf(Object o) {

return 0;

}

@Override

public int lastIndexOf(Object o) {

return 0;

}

@Override

public ListIterator listIterator() {

return null;

}

@Override

public ListIterator listIterator(int index) {

return null;

}

@Override

public List subList(int fromIndex, int toIndex) {

return null;

}

@Override

public boolean retainAll(Collection c) {

return false;

}

@Override

public boolean removeAll(Collection c) {

return false;

}

@Override

public boolean containsAll(Collection c) {

return false;

}

@Override

public Object[] toArray(Object[] a) {

return new Object[0];

}

@Override

public String toString() {

String elements = "";

Node currNode = head;

while (currNode != null) {

elements += currNode.value;

if (currNode != tail)

elements += ", ";

currNode = currNode.next;

}

return "[Size: " + *size* + " - " + elements + "]";

}

public static void main(String[] args) {

LinkedList list = new LinkedList();

list.add(4);

list.add(5);

list.add(7);

list.add(2);

System.*out*.println(list);