Get Value In Linked List

Question

1. You are given a partially written LinkedList class.  
2. Here is a list of existing functions:  
2.1 addLast - adds a new element with given value to the end of Linked List  
2.2. display - Prints the elements of linked list from front to end in a single line. All elements are separated by space  
2.3. size - Returns the number of elements in the linked list.  
2.4. removeFirst - Removes the first element from Linked List.   
3. You are required to complete the body of getFirst, getLast and getAt function   
3.1. getFirst - Should return the data of first element. If empty should return -1 and print "List is empty".  
3.2. getLast - Should return the data of last element. If empty should return -1 and print "List is empty".  
3.3. getAt - Should return the data of element available at the index passed. If empty should return -1 and print "List is empty". If invalid index is passed, should return -1 and print "Invalid arguments".  
4. Input and Output is managed for you.

Input Format

Input is managed for you

Output Format

Output is managed for you

Constraints

None

Sample Input

addLast 10  
getFirst  
addLast 20  
addLast 30  
getFirst  
getLast  
getAt 1  
addLast 40  
getLast  
addLast 50  
removeFirst  
getFirst  
removeFirst  
removeFirst  
getAt 3  
removeFirst  
removeFirst  
getFirst  
quit

Sample Output

10  
10  
30  
20  
40  
20  
Invalid arguments  
List is empty

import java.io.\*;

import java.util.\*;

public class Main {

public static class Node {

int data;

Node next;

}

public static class LinkedList {

Node head;

Node tail;

int size;

void addLast(int val) {

Node temp = new Node();

temp.data = val;

temp.next = null;

if (size == 0) {

head = tail = temp;

} else {

tail.next = temp;

tail = temp;

}

size++;

}

public int size(){

return size;

}

public void display(){

for(Node temp = head; temp != null; temp = temp.next){

System.out.print(temp.data + " ");

}

System.out.println();

}

public void removeFirst(){

if(size == 0){

System.out.println("List is empty");

} else if(size == 1){

head = tail = null;

size = 0;

} else {

head = head.next;

size--;

}

}

public int getFirst(){

if(head==null){

System.out.println("List is empty");

return -1;

}else{

return head.data;

}

}

public int getLast(){

if(head==null){

System.out.println("List is empty");

return -1;

}else{

return tail.data;

}

}

public int getAt(int idx){

if(idx==0){

return getFirst();

}else if(idx==size-1){

return getLast();

}else if(idx<0 || idx>=size){

System.out.println("Invalid arguments");

return -1;

}else{

Node temp=head;

while(idx>0){

temp=temp.next;

idx--;

}

return temp.data;

}

}

}

public static void main(String[] args) throws Exception {

BufferedReader br = new BufferedReader(new InputStreamReader(System.in));

LinkedList list = new LinkedList();

String str = br.readLine();

while(str.equals("quit") == false){

if(str.startsWith("addLast")){

int val = Integer.parseInt(str.split(" ")[1]);

list.addLast(val);

} else if(str.startsWith("size")){

System.out.println(list.size());

} else if(str.startsWith("display")){

list.display();

} else if(str.startsWith("removeFirst")){

list.removeFirst();

} else if(str.startsWith("getFirst")){

int val = list.getFirst();

if(val != -1){

System.out.println(val);

}

} else if(str.startsWith("getLast")){

int val = list.getLast();

if(val != -1){

System.out.println(val);

}

} else if(str.startsWith("getAt")){

int idx = Integer.parseInt(str.split(" ")[1]);

int val = list.getAt(idx);

if(val != -1){

System.out.println(val);

}

}

str = br.readLine();

}

}

}