```java
package LinkedList;
public class DSA1_LinkedList {
    public static class Node{
        private int data;
        private Node next;
    }
    public static class LinkedList{
        private Node head;
        private Node tell;
        private int size;

        public void addLast(int val){
            Node node=new Node();
            node.data=val;
            node.next=null;
            if(size==0){
                head=tell=node;
            }
            else{
                tell.next=node;
                tell=node;
            }
            size++;
        }
        public void addFirst(int val){
            Node node=new Node();
            node.data=val;
            node.next=head;
            head=node;
            size++;
        }
        public void addAtIndex(int val,int index){
            Node node=new Node();
            node.data=val;
            Node temp=head;
            for(int i=0;i<index-1;i++){
                temp=temp.next;
            }
            node.next=temp.next;
            temp.next=node;
            size++;
        }
        public void removeFirst(){
            if(this.size==0){
                System.out.println("LinkedList is empety");
            }
            else if(size==1){
                head=tell=null;
                size=0;
            }
            else{
                head=head.next;
                size--;
            }
        }
        public void removeLast(){
            Node temp=head;
            for(int i=0;i<size-2;i++){
                temp=temp.next;
            }
            tell=temp;
            tell.next=null;
            size--;
        }
        void display(){
            Node temp=head;
            while(temp!=null){
                System.out.print(temp.data+"->");
                temp=temp.next;
            }
            System.out.println();
        }
        private void reverseLinkedList(){
            int i=0;
            int j=size-1;
            while(i<j){
                Node left=getNodeAt(i);
                Node right=getNodeAt(j);
                int temp=left.data;
                left.data=right.data;
                right.data=temp;
                i++;
                j--;
            }
        }
        public void size(){
            System.out.println("Size of LinkedList->"+size);
        }
        public int getFirst(){
            return head.data;
        }
        public int getLast(){
            return tell.data;
        }
        public int getElement(int index){
            if(size==0){
                System.out.println("List is empty");
                return 0;
            }
            else if(index<0||index>=size){
                System.out.println("Invalid Input");
                return 0;
            }
            Node temp=head;
            for(int i=0;i<index;i++){
                temp=temp.next;
            }
            return temp.data;
        }
        private Node getNodeAt(int index){
            if(size==0){
                System.out.println("List is empty");
                // return 0;
            }
            else if(index<0||index>=size){
                System.out.println("Invalid Input");
                // return 0;
            }
            Node temp=head;
            for(int i=0;i<index;i++){
                temp=temp.next;
            }
            return temp;
        }
        public void removeIndexElement(int index){
            Node temp=head;
            for(int i=0;i<index-1;i++){
                temp=temp.next;
            }
            temp.next=temp.next.next;
            size--;
        }
        public void normalPrint(int getElement){
            System.out.println("********************");
            System.out.println("1st Element ->"+getFirst()+
            "\nLast Element ->"+getLast()+
            "\nElement "+getElement+"th Index ->"+getElement(getElement)+
            "\nSize ->"+size);
            System.out.println("********************");

        }
    }
    Run | Debug
    public static void main(String[] args) throws NullPointerException{
        LinkedList lst=new LinkedList();
        for(int i=1;i<=10;i++){
            lst.addLast(i*10);
        }
        lst.size();
        lst.addFirst(64);
        lst.addLast(34);
        lst.addAtIndex(43, 8);
        lst.display();
        lst.removeLast();
        lst.removeIndexElement(2);
        lst.reverseLinkedList();
        lst.display();
        lst.normalPrint(5);
    }
}
```