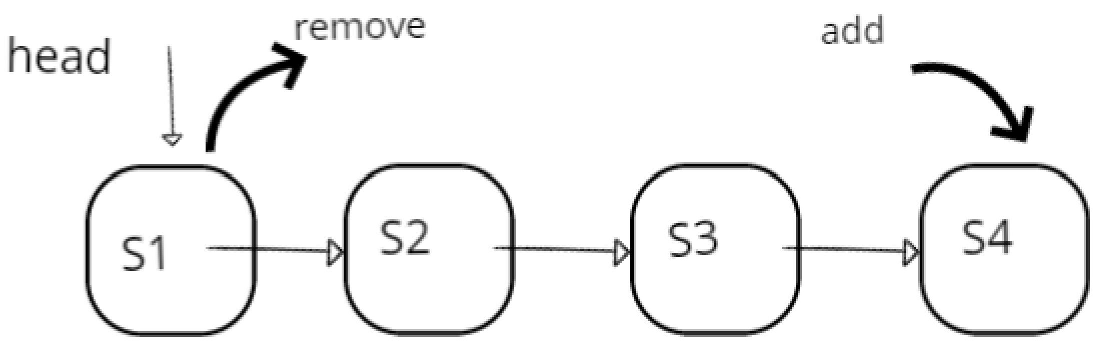
```
import java.util.NoSuchElementException;

public class Queue<E> {

    private Node head;//mientras no se les asigne un valor estan en null
    private int size;

    public Queue() {
        size = 0;
    }
    /**
     * this class keeps track of each element information
     * @author java2novice
     *
     */
    private class Node {
        E element;
        Node next;//estos nodos valen null

        public Node(E element) {
            this.element = element;
            this.next = null;
        }
    }
    /**
     * returns the size of the linked list
     * @return
     */
    public int size() { return size; }

    /**
     * return whether the list is empty or not
     * @return
     */
    public boolean isEmpty() { return size == 0; }

    /**
     * adds element at the end of the linked list
     * @param element
     */
    public void add(E element) {
            System.out.println("Añadiendo nodo *******************");
        Node tmp = new Node(element);
        System.out.printf("TEMP: %s, %s\n", tmp.element, tmp);
        System.out.printf("HEAD ANTES: %s\n", head);
        if(head == null) {
            head=tmp;
        }else {

            Node n = head;
            while(n.next != null) {
                    n=n.next;
            }
            n.next = tmp;
        }

        System.out.printf("HEAD DESUES: %s\n", head);
        size++;
        System.out.println("adding: " + element);
    }

    /**
     * this method walks forward through the linked list
     */
    public void iterateForward(){

        System.out.println("iterating forward..");
        Node tmp = head;
        while(tmp != null){
            System.out.println(tmp.element);
            tmp = tmp.next;
        }
    }


    /**
     * this method removes element from the start of the linked list
     * @return
     */

    public E peek() {
            if (size == 0) throw new NoSuchElementException();
            return head.element;
    }

    public Node remove() {

        if (size == 0) throw new NoSuchElementException();
        size--;
        System.out.println("removing" + head.element);
        head = head.next;
        return head;
    }
    public static void main(String a[]){

        Queue<Integer> dll = new Queue<Integer>();
        dll.add(10);
        dll.add(34);
        dll.add(56);
        dll.iterateForward();
        dll.remove();
        dll.add(78);
        dll.iterateForward();
    }
}
```


head
remove
add
S1
S2
S3
S4