Generics, Arrays and Containers

Create a generic, singly linked list class called SList, which, to keep things simple, does not implement the List interface.

Each Link object in the list should contain a reference to the next element in the list, but not the previous one (LinkedList, in contrast, is a doubly linked list, which means it maintains links in both directions).

Create your own SListIterator which, again for simplicity, does not implement ListIterator. The only method in SList other than toString( ) should be iterator( ), which produces an SListIterator.

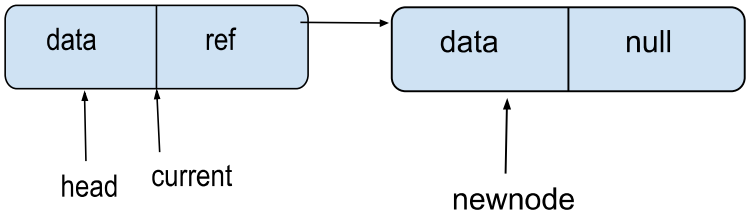
The only way to insert and remove elements from an SList is through SListIterator. Write code to demonstrate SList.

Class Node<T>
The Node class represents a node in the singly linked list, containing data and a reference to the next node.

Class Slist<T>
It is a outer class which holds the reference for the head and tail

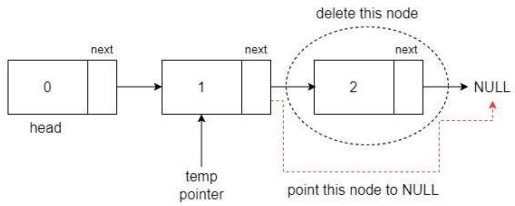
Class SListiterator
The SListiterator class provides methods to add, remove, and traverse nodes in the singly linked list.

Adding operation



current.next=newnode
newnode.next=null

Deletion operation.




```
 while (current.next != tail) {
            current = current.next;
        }
        current.next = null;
        tail = current;
}
```

Output-