Single Linked List

Problem Statement:

You are required to implement a single linked list in Java. The provided skeleton code outlines the basic structure for the linked list, including the `Node` class that represents each element in the list, and various methods to manipulate the list. Your task is to complete the implementation and enhance it with additional features.

Tasks:

1. Implement the Constructor: Complete the constructor `SingleLinkedList()` to initialize an empty linked list.
2. Implement `getFirst()` Method: Write the `getFirst()` method to return the first element in the list.
3. Implement `getLast()` Method: Write the `getLast()` method to return the last element in the list.
4. Implement `addFirst(Object e)` Method: Write the `addFirst(Object e)` method to add an element at the beginning of the list.
5. Implement `addLast(Object e)` Method: Write the `addLast(Object e)` method to add an element at the end of the list.
6. Implement `add(Object e)` Method: Write the `add(Object e)` method to add an element at the end of the list (similar to `addLast()`).
7. Implement `removeFirst()` Method: Write the `removeFirst()` method to remove and return the first element in the list.
8. Implement `removeLast()` Method: Write the `removeLast()` method to remove and return the last element in the list.
9. Implement `size()` Method: Write the `size()` method to return the number of elements in the list.

Additional Tasks:

1. Use Generics: Modify the implementation to use generics, allowing the linked list to store elements of any type.
2. Implement `printList()` Method: Implement a method to print all the elements in the list.
3. Reverse the List: Implement a method to reverse the order of elements in the linked list.
4. Find Middle Element: Implement a method to find and return the middle element of the linked list.
5. Check for Cycles: Implement a method to detect if the linked list contains a cycle (i.e., a loop where a node points back to a previous node).
6. Merge Two Lists: Implement a static method to merge two sorted linked lists into a single sorted linked list.
7. Implement `contains()` Method: Add a method `contains(Object e)` to check if a given element exists in the list.
8. Implement `clear()` Method: Add a method `clear()` to remove all elements from the linked list.

9. Convert to Doubly Linked List: As an advanced exercise, convert the implementation into a doubly linked list where each node has a pointer to both the next and previous nodes.