```
public boolean isPalindrome() {
            if (head != null) {
                  Node<T> tempS = head;
                  Node<T> tempE = current;
                  while (tempE.next != null)
                        tempE = tempE.next;
                  while (tempS != tempE && tempE.next != tempS) {
                        if (!tempS.data.equals(tempE.data))
                              return false;
                        tempS = tempS.next;
                        tempE = tempE.previous;
                  }
            }
            return true;
}
```

```
    public static <T> void split(LinkedQueue<T> q, LinkedQueue<T> q1,
LinkedQueue<T> q2) {
          int n = q.length();
          for (int i = 0; i < n; i++) {
                T temp = q.serve();
                q.enqueue(temp);
                if((i+1) % 2 == 1) {
                      if(!q1.full())
                            q1.enqueue(temp);
                }
                else {
                      if(!q2.full())
                            q2.enqueue(temp);
                }
          }
    }
```