|  |
| --- |
| /\*So we need to add some characters to the given string or character and find out what will be the shortest palindrome string by using simple java program.\*/ |
|  | Write a simple code to identify given linked list is palindrome or not by using stack. |
|  | First take a Stack. Traverse through each node of the linked list and push each node value to Stack. |
|  | Once the traversal & copying is done, iterate through linked list from head node again. |
|  | In each iteration, pop one stack element and compare with node value in respective iteration. It is expected to match stack popped value with node value. |
|  | In case of all matches, its a palindrome. Any one element mismatch makes it not a palindrome.\*/ |
|  |  |
|  | import java.util.Stack; |
|  | class Node { |
|  | int data; |
|  | Node next; |
|  |  |
|  | Node(int i) |
|  | { |
|  | this.data = i; |
|  | this.next = null; |
|  | } |
|  | }; |
|  |  |
|  | class Main |
|  | { |
|  |  |
|  | public static boolean isPalindrome(Node head) |
|  | { |
|  | k |
|  | Stack<Integer> s = new Stack<>(); |
|  |  |
|  |  |
|  | Node node = head; |
|  | while (node != null) { |
|  | s.push(node.data); |
|  | node = node.next; |
|  | } |
|  | node = head; |
|  | while (node != null) |
|  | { |
|  |  |
|  | int top = s.pop(); |
|  |  |
|  | if (top != node.data) { |
|  | return false; |
|  | } |
|  |  |
|  | // advance to the next node |
|  | node = node.next; |
|  | } |
|  |  |
|  |  |
|  | return true; |
|  | } |
|  |  |
|  | public static void main(String[] args) |
|  | { |
|  | Node head = new Node(1); |
|  | head.next = new Node(2); |
|  | head.next.next = new Node(3); |
|  | head.next.next.next = new Node(2); |
|  | head.next.next.next.next = new Node(1); |
|  |  |
|  | if (isPalindrome(head)) { |
|  | System.out.print("Linked List is a palindrome."); |
|  | } else { |
|  | System.out.print("Linked List is not a palindrome."); |
|  | } |
|  | } |
|  | } |