Write a function *detectAndRemoveLoop()* that checks whether a given Linked List contains loop and if loop is present then removes the loop and returns true. If the list doesn’t contain loop then it returns false. Below diagram shows a linked list with a loop. *detectAndRemoveLoop()* must change the below list to 1->2->3->4->5->NULL.



Solution:

// Function that detects loop in the list

**void** detectAndRemoveLoop(Node node)

{

// If list is empty or has only one node

// without loop

**if** (node == **null** || node.next == **null**)

**return**;

Node slow = node, fast = node;

// Move slow and fast 1 and 2 steps

// ahead respectively.

slow = slow.next;

fast = fast.next.next;

// Search for loop using slow and fast pointers

**while** (fast != **null** && fast.next != **null**) {

**if** (slow == fast)

**break**;

slow = slow.next;

fast = fast.next.next;

}

/\* If loop exists \*/

**if** (slow == fast) {

Node curr = node;

**if** (curr != fast) {

**while** (curr.next != fast.next) {

curr = curr.next;

fast = fast.next;

}

/\* since fast->next is the looping point \*/

fast.next = **null**; /\* remove loop \*/

}

/\* This case is added if fast and slow pointer meet at first position. \*/

**else** {

**while**(fast.next != slow) {

fast = fast.next;

}

fast.next = **null**;

}

}

}