https://course.acciojob.com/idle?question=130e273c-3811-43a0-95e0-2862dbd39118

- EASY
- Max Score: 30 Points

Linked List Cycle

You are given the `head` node of the linked list. You need to check for any cycle is present in linked list or not, You need to output `1` if cycle is present else `0`.

Note

You need to only complete the function. Dont worry about input, it is for internal reference.

Input Format

The first line of input contains `N` representing the number of nodes in linked list.

The second line of input contains `N` space separated integers, representing elements in linked list.

The third line of input contains a number `x` representing which node is connected to others.

Output Format

In a single line print `1` or `0`.

Example 1

Input

```
3
1 3 4
2
```

Output

```
1
```

Explanation

```
1->3->4
  ^   |
  |___|
```

This is the list given in question. A loop is present in this linked list

Example 2

Input

```
4
1 2 3 4
0
```

Output

```
0
```

Explanation

1->2->3->4 this is list you are given in question, and the answer is 0 as it doesnt contains any cycle.

Constraints

1 <= N <= 1000

1 <= value of node <= 1000

Topic Tags

- **Linked lists**

My code

```
// in java
import java.util.*;
import java.io.*;
import java.lang.*;

class Node
    {
        int data;
        Node next;
        Node(int d) {data = d; next = null; }
    }

class LinkedList
{
    Node head;
    Node tail;
    public void addToTheLast(Node node)
    {
      if (head == null)
      {
```

```
   head = node;
   tail = node;
  }
  else
  {
   tail.next = node;
   tail = node;
  }
}

public static void makeLoop(Node head, int x){
    if (x == 0)
        return;
    Node curr = head;
    Node last = head;

    int currentPosition = 1;
    while (currentPosition < x)
    {
        curr = curr.next;
        currentPosition++;
    }

    while (last.next != null)
        last = last.next;
    last.next = curr;
}
  void printList()
{
    Node temp = head;
    while (temp != null)
    {
```

```
            System.out.print(temp.data+" ");
            temp = temp.next;
        }
        System.out.println();
    }

}

public class Main {
    public static void main(String[] args) throws Throwable {
        Scanner sc = new Scanner(System.in);
        int n = sc.nextInt();
        LinkedList llist= new LinkedList();
        int a1=sc.nextInt();
        Node head= new Node(a1);
        llist.addToTheLast(head);
        for (int i = 1; i < n; i++)
        {
            int a = sc.nextInt();
            llist.addToTheLast(new Node(a));
        }


        int pos = sc.nextInt();
        llist.makeLoop(head, pos);

        Solution x = new Solution();
        boolean a=x.detectLoop(head);

        if(a==true)
            System.out.println(1);
        else
```

```
            System.out.println(0);

    }
}

class Solution
{
    //Function to remove a loop in the linked list.
    public static boolean detectLoop(Node node){
    //your code here
                HashMap<Node,Integer>hm=new HashMap<>();

while(node!=null)
        {
                hm.put(node,1);
                if(hm.containsKey(node.next))
                {
                      //node.next=null;
                        return true;
                }
                node=node.next;
        }
       return false;
    }
}
```