

<https://course.acciojob.com/idle?question=f7994df4-2130-46b1-a9c2-3b3c68b7366c>

• EASY

• Max Score: 30 Points

- 
- 
- 
- 
- 
- 
- 
- 
- 
- 
- 
- 
- 
- 
- 

## Intersection of Two Linked Lists

---

Your are given two linked List consisting of N nodes and M nodes consisting of a common intersection point.

Your task is to find the value of the intersection point of the two linked list.

### Input Format

The first line contains the number of test cases.

The first line of each test case has a string containing the elements of the first linked list.

The second line of each test case has a string containing the elements of the second linked list.

## Output Format

For each test case print an integer in a new line, denoting the value of the intersection point.

## Example 1

Input

```
1
3 6 9 15 30
10 15 30
```

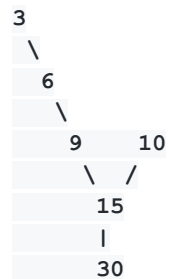
Output

```
15
```

Explanation

L1: 3 - 6 - 9 - 15 - 30

L2: 10 - 15 - 30



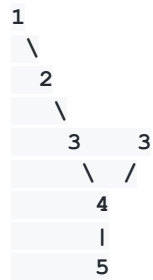
The value of the intersection node is 15. Thus answer is 15

## Example 2

Input

L1: 1 - 2 - 3 - 4 - 5

L2: 3 - 4 - 5



Output

4

Explanation

The value of intersection node is 4. Thus answer is 4.

## Constraints

$1 \leq T \leq 10$

$1 \leq N \leq 10000$

$1 \leq M \leq 10000$

$1 \leq L1[i] \leq 100000$

$1 \leq L2[i] \leq 100000$

### Topic Tags

- **Linked lists**

# My code

```
// n java
import java.util.*;

class Main{

    static Node head1 = null;
    static Node head2 = null;

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

    static Node intersectingNode(Node headA, Node headB)
    {
        //write code here
        int len1=0;
        int len2=0;
        Node a=headA;
```

```

Node b=headB;
while(a!=null)//finding lenth of 1 ll
{
    len1++;
    a=a.next;
}

while(b!=null)//finding lenth of 2 ll
{
    len2++;
    b=b.next;
}
int dif=0;
if(len1>len2)
    dif=len1-len2;
else    dif=len2-len1;

    if(len1>len2)
    {
        while(dif!=0)
        {
            dif--;
            headA= headA.next;
        }
    }
else
    {
        while(dif!=0)
        {
            dif--;

```

```

        headB= headB.next;
    }
}
while(headA!=null)
{
    if( headA== headB)
        return headA;
    headA=headA.next;
    headB= headB.next;
}
return null;
}

```

```

static void formLinkList(int n,int m,int k,int[] a, int[] b)
{
    head1= new Node(a[0]);
    Node temp =head1;
    int i=1;
    Node need= null;
    while(i<n){
        temp.next = new Node(a[i]);
        temp=temp.next;
        if(i==k) need = temp;
        i++;
    }

    head2 = new Node(b[0]);
    i=1;
    temp = head2;
    while(i<m){

```

```

        temp.next = new Node(b[i]);
        temp=temp.next;
        i++;
    }
    temp.next=need;
    return;
}

```

```

public static void main(String[] args)
{
    Scanner sc = new Scanner(System.in);
    int t=0;
    t = sc.nextInt();
    while(t-->0){
        head1=null;
        head2=null;
        int n=0,m=0,k=0;
        n = sc.nextInt();
        m = sc.nextInt();
        k = sc.nextInt();
        int[] a =new int[n];
        int[] b =new int[m];
        for(int i=0;i<n;i++){
            a[i]=sc.nextInt();
        }
        for(int i=0;i<m;i++){
            b[i]=sc.nextInt();
        }
        formLinkList(n,m,k,a,b);
    }
}

```

```
        System.out.println(Math.abs(intersectingNode(head1,
head2).data));
    }
    sc.close();
    return;
}
}
```