

ממש פעולה חיצונית `exist` ב- Java או `Exist` ב- C# .

הפעולה מקבלת עץ בינירי `t` מטיפוס שלם ומספר שלם `x` . הפעולה תחזיר אם יש בעץ צומת שערךו `x` , אחרת — הפעולה תחזיר `false` . אם העץ ריק — הפעולה תחזיר `false` .

.  
ב. לפניה הפעולה ב- Java `check(t1, t2)` וב- C# `Check(t1, t2)`

הפעולה מקבלת שני עצים בינירים לא ריקים מטיפוס שלם, `t1` ו- `t2` , ומחזירה רשימה המכילה את כל המספרים הנמצאים בעץ `t1` ואינם נמצאים בעץ `t2` . הפעולה מזמנת פעולה נוספת המקבלת שלושה פרמטרים.

### Java

```
public static Node<Integer> check(BinNode<Integer> t1, BinNode<Integer> t2)
{
    Node<Integer> first = new Node<Integer> (-1);
    first = check(t1 ,t2 ,first);
    return first.getNext();
}
```

### C#

```
public static Node<int> Check(BinNode<int> t1 ,BinNode<int> t2)
{
    Node<int> first = new Node<int> (-1);
    first = Check(t1 ,t2 ,first);
    return first.GetNext();
}
```

ממש את הפעולה:

:Java

public static Node<Integer> check( BinNode<Integer> t1,
 BinNode<Integer> t2 , Node<Integer> list)

:C#  
או ב-

public static Node<int> Check(BinNode<int> t1 ,BinNode<int> t2 , Node<int> list)

אתה יכול להשתמש בפעולת `شمימשת` בסעיף א.

.ג. מה היא סיבוכיות זמו הרצה של הפעולה `شمימשת` בסעיף ב ? נמה את תשובה.