**4.** בשאלה זו תוכל להשתמש בפעולה החיצונית שלפניך בלי לממש אותה.

| כותרת הפעולה | תיאור הפעולה |
|---|---|
| בשפת **Java**<br>public static Node<Integer> delete (int num, Node<Integer> lst)<br>בשפת **C#**<br>public static Node<int> Delete (int num, Node<int> lst) | הפעולה מקבלת מספר num — והפנָיה לתחילת שרשרת חוליות lst — .<br>הפעולה מוחקת את החוליות שבהן הערך num ומחזירה הפנָיה לתחילת שרשרת החוליות. |

נתונה המחלקה **BiList** — דו־שרשרת, ולה שתי תכונות:

- lst1 — הפניה לתחילת שרשרת חוליות מטיפוס שלם
- lst2 — הפניה לתחילת שרשרת חוליות מטיפוס שלם

לפניך ממשק חלקי של המחלקה **BiList** בשפות **Java** ו־ **C#** .

יש להשתמש בפעולות הממשק ללא צורך לממש אותן.

| כותרת הפעולה | תיאור הפעולה |
|---|---|
| public BiList () | פעולה הבונה את העצם עם הפניות לשתי שרשראות ריקות. |
| בשפת **Java**<br>public void addNum (int num, int codeList)<br>בשפת **C#**<br>public void AddNum (int num, int codeList) | פעולה המוסיפה חוליה שבה הערך num לסוף השרשרת lst1 או לסוף השרשרת lst2 בהתאם ל־ codeList :<br>כאשר codeList = 1 , יוכנס num ל־ lst1 ,<br>וכאשר codeList = 2 , יוכנס num ל־ lst2 .<br>הנח שהערך של הפרמטר codeList תקין. |

כתוב פעולה חיצונית ששמה generateBilist בשפת Java או GenerateBilist בשפת C# המקבלת שרשרת חוליות — lst של מספרים שלמים. מספר החוליות ב־ lst זוגי והמספרים בחוליות שלה שונים זה מזה.

הפעולה תחזיר עצם מטיפוס **BiList** שמתקיימים בו התנאים האלה:

— כל אחד מן המספרים שבשרשרת lst יופיע באחת מן השרשראות lst1 ו־ lst2 .

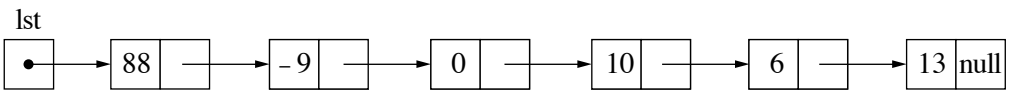
— כל המספרים בשרשרת lst1 יהיו גדולים מכל המספרים בשרשרת lst2 .

— מספר החוליות בשתי השרשראות lst1 ו־ lst2 יהיה זהה.

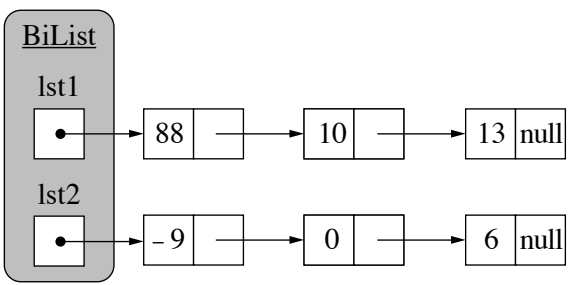
שים לב: אין להוסיף פעולות, גם לא פעולות get ו־ set בשפת Java או Get ו־ Set בשפת C# , למחלקה **BiList** .

דוגמה:

בעבור השרשרת lst שלפניך:




הפעולה תחזיר את העצם הזה:




הערות:

— אין צורך לשמור על השרשרת lst .

— אין חשיבות לסדר האיברים בשרשרת lst1 ובשרשרת lst2 .