

שאלת חזרה בנושא: רשימה, יעילות, רקורסיה

היפוך רשימה

מטרות

מימוש פעולה פנימית במחלקה רשימה תוך שימוש בייצוג המחלקה.

רמת השאלה

בינונית.

מה עליכם לעשות?

א. נוסף למחלקה $List<T>$ המיוצגת על ידי שרשרת חוליות חד-כיוונית, את הפעולה הבאה:

```
public void reverse()
```

הפעולה הופכת את סדר איברי הרשימה הנוכחית.

ממשו את הפעולה כך שסדר הגודל שלה לא יעלה על $O(n)$ כאשר n הוא מספר האיברים ברשימה.

ב. להלן מימוש של הפעולה $reverse()$ כפי שנכתב על ידי התלמיד משה:

```
public void reverse()
{
    Node<T> pos = this.first;

    while(pos != null)
    {
        this.insert(null, pos.getInfo());
        pos = this.remove(pos);
    }
}
```

האם המימוש של משה אכן הופך את הרשימה?

אם כן, נתחו את יעילות הפעולה. אם לא הסבירו איפה הבעיה במימוש של משה?

ג. ממשו את הפעולה $reverse()$ בצורה רקורסיבית.

הנחיות מיוחדות

- שאלה זו בודקת את היכולת של התלמיד לחשוב על רעיון אלגוריתמי שיאפשר מימוש הפעולה תחת מגבלת סדר גודל של יעילות האלגוריתם.
- מאחר והרשימה מיוצגת בעזרת שרשרת חוליות, נכון יהיה לנצל עובדה זו ולשנות את ההפניות מחוליה לחוליה – כלומר, במקום שהחוליה הראשונה תפנה לחוליה השניה, נשנה זאת כך שהחוליה השניה תפנה לראשונה והראשונה ל- $null$ וכך הלאה. לבסוף נקבל רשימה הפוכה. זהו מעין תהליך של **היפוך במקום** ומעניין לחשוף את התלמידים ליכולת שלהם לבצע היפוך שכזה ללא שימוש במבנה עזר.
- רעיון אחר לפתרון הוא על ידי שימוש במחסנית. בסריקה ראשונה של הרשימה נכניס את כל האיברים למחסנית – כלומר, האיבר האחרון ברשימה יהיה בראש המחסנית. נסרוק את הרשימה פעם שניה,

ותוך כדי ההתקדמות נשלוף איברים מהמחסנית ונעדכן את תוכן החוליות ברשימה בערכי האיברים הנשלפים מהמחסנית.

- סעיף ב' של השאלה מראה שמימוש בעזרת הפעולה `remove(...)` של רשימה שהיא פעולת ממשק "יקרה" מגדיל את סדר הגודל של הפעולה.

פתרון

א. מימוש אפשרי ראשון:

```
public void reverse()
{
    Node<T> pos1 = this.first;
    Node<T> pos2 = null;
    Node<T> pos3 = null;

    while(pos1 != null)
    {
        pos2 = pos1.getNext();
        pos1.setNext(pos3);
        pos3 = pos1;
        pos1 = pos2;
    }
    this.first = pos3;
}
```

מימוש אפשרי שני:

```
public void reverse()
{
    Stack<T> stk = new Stack<T>();
    Node<T> pos = this.first;

    while(pos != null)
    {
        stk.push(pos.getInfo());
        pos = pos.getNext();
    }

    pos = this.first;
    while(pos != null)
    {
        pos.setInfo(stk.pop());
        pos = pos.getNext();
    }
}
```

ב. מימוש נכון. סדר גודל ריבועי $O(n^2)$.

מקור השאלה

צוות הפיתוח.