2. נתונות שתי הפניות לשרשרות של חוליות:

numHead – הפניה לשרשרת חוליות של מספרים מטיפוס שלם.

charHead – הפניה לשרשרת חוליות מטיפוס תו (char), שבה מיוצגות פעולות החשבון חיבור וחיסור ('+', '−').

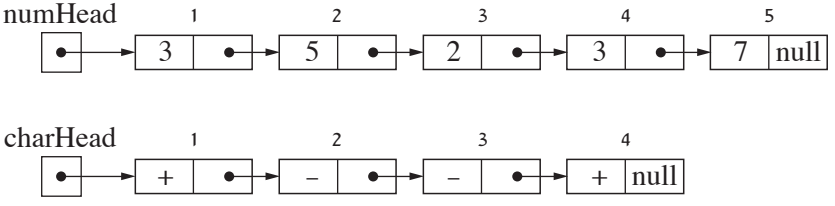
הערה: כמות החוליות בשרשרת המספרים גדולה ב־ 1 מכמות החוליות בשרשרת התווים.

שילוב בין שתי השרשרות מייצג סדרת פעולות חשבון, כמפורט לפניכם:

התו במיקום 1 בשרשרת התווים מתאר את הפעולה שבאה לפני המספר שבמיקום 2 בשרשרת המספרים, התו במיקום 2 מתאר את הפעולה שבאה לפני המספר שבמיקום 3, התו במיקום 3 מתאר את הפעולה שבאה לפני המספר שבמיקום 4, וכן הלאה עד סוף שתי השרשרות.

שימו לב: אין פעולת חשבון לפני המספר הראשון בשרשרת המספרים.

דוגמה: שתי השרשרות שלפניכם מייצגות את סדרת פעולות החשבון: $7 + 3 - 2 - 5 + 3$.



א. ממשו את הפעולה שלפניכם:

Java – public static int eval (Node<Integer> numHead, Node<Character> charHead, int len)

C# – public static int Eval (Node<int> numHead, Node<char> charHead, int len)

len הוא כמות **המספרים** שנרצה לחשב מתוך שרשרת המספרים (אין פעולת חשבון לפני המספר הראשון ולכן כמות התווים היא len–1 בהתאם).

הפעולה תחזיר את תוצאת החישוב של len מספרים ברצף, החל מתחילת שרשרת המספרים ותחילת שרשרת התווים.

הניחו ש־ len גדול מ־ 0 . אין לשנות את השרשרות שהתקבלו.

הערה: אם len גדול מכמות החוליות בשרשרת המספרים, הפעולה תחזיר את תוצאת החישוב של כל המספרים בשרשרת.

דוגמה: בעבור שתי השרשרות בדוגמה שלעיל ו־ $\text{len} = \mathbf{1}$, הפעולה תחזיר 3 , כיוון שהוא המספר הראשון בשרשרת המספרים (היות שהוא המספר הראשון והיחיד, אין שום פעולת חשבון שצריך לבצע).

דוגמה נוספת: בעבור אותן שתי שרשרות ו־ $\text{len} = \mathbf{3}$, הפעולה תחזיר 6 ($3 + 5 - 2 = 6$).

דוגמה נוספת: בעבור אותן שתי שרשרות ו־ $\text{len} = \mathbf{8}$, הפעולה תחזיר 10 ($3 + 5 - 2 - 3 + 7 = 10$).

הסבר: מספר החוליות בשרשרת המספרים קטן מ־ 8 , ולכן הפעולה מחזירה את תוצאת החישוב של כל המספרים בשרשרת.

ב. ממשו את הפעולה שלפניכם:

Java – public static boolean match (Node<Integer> numHead, Node<Character> charHead, int len, int val)

C# – public static bool Match (Node<int> numHead, Node<char> charHead, int len, int val)

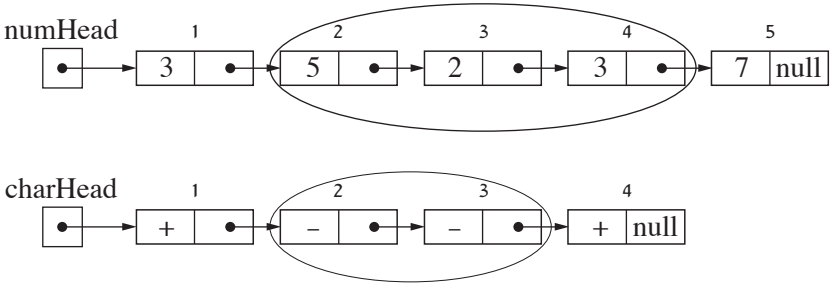
הפעולה תחזיר true אם קיימת חוליה **במיקום כלשהו** בשרשרת המספרים וחוליה **במיקום זהה** בשרשרת התווים, שמהם והלאה יש **len** מספרים ברצף שתוצאת החישוב שלהם היא **val** (כלומר אם מתחילים את החישוב **מן החוליה השנייה** בשרשרת המספרים, גם פעולת החשבון הראשונה תהיה **החוליה השנייה** בשרשרת התווים). אחרת הפעולה תחזיר false .

אפשר להשתמש בפעולה שכתבתם בסעיף א. הניחו ש־ len גדול מ־ 0 . אין לשנות את השרשרות שהתקבלו.

<u>שימו לב</u>: אין פעולת חשבון לפני המספר הראשון בחישוב (מספר התווים הנדרש הוא len–1).

<u>הערה</u>: אם מחוליה כלשהי יש פחות מ־ len מספרים עד סוף השרשרת – החישוב מתבצע עד סוף השרשרת.

<u>דוגמה</u>: בעבור שתי השרשרות המוצגות בדוגמה שלעיל, **len = 3** ו־ **val = 0** , הפעולה תחזיר **true** , כי החל **מן החוליה השנייה** בשרשרת המספרים ו**מן החוליה השנייה** בשרשרת התווים יש 3 מספרים ברצף שתוצאת החישוב שלהם היא 0 (5 − 2 − 3 = 0).



<u>דוגמה נוספת</u>: בעבור שתי השרשרות המוצגות בדוגמה שלעיל, **len = 3** ו־ **val = 10** , הפעולה תחזיר **true** , כי החל **מן החוליה הרביעית** בשרשרת המספרים ו**מן החוליה הרביעית** בשרשרת התווים ועד סוף שרשרת המספרים תוצאת החישוב היא 10 (3 + 7 = 10).