מטלת מנחה (ממ"ן) 14

הקורס: 20441 - מבוא למדעי המחשב ושפת Java

חומר הלימוד למטלה: יחידה 11 | **נושא המטלה:** רשימות מקושרות

מספר השאלות: 4 | **משקל המטלה:** 3 נקודות

סמסטר: 2024ב | **מועד אחרון להגשה:** 22.6.2024

במטלה זו נעסוק ברשימות מקושרות של מספרים שלמים.

בכל השאלות נשתמש ברשימה דו-סטרית. לשם כך מצורפים לממ"ן זה קובץ המחלקה IntNodeTwo.java המייצגת חוליה ברשימה, וקובץ המחלקה IntListTwo.java המייצגת רשימה מקושרת כזו.

פתחו את קובץ המחלקה IntNodeTwo ועברו עליו.
בראש הקובץ נמצאות הגדרות התכונות של המחלקה וכן כמה בנאים ושיטות עזר בהן תוכלו להסתייע בעבודתכם.

פתחו את קובץ המחלקה IntListTwo ועברו עליו. זהו הקובץ בתוכו תממשו את הפתרון שלכם לממ"ן זה.
בראש הקובץ נמצאות הגדרות התכונות של המחלקה וכן כמה בנאים ושיטות עזר בהן תוכלו להסתייע בעבודתכם.

הורידו את שתי המחלקות הללו אל המחשב שלכם ועבדו על הקובץ IntListTwo.java.
אל תמחקו ואל תשנו דבר בקבצים שהעתקתם, רק הוסיפו להם !

הערות חשובות לגבי כל השאלות בממ"ן זה:

- **אסור לכם להעביר את תוכן הרשימה למערך ואז לעבוד על המערך. כך גם לא להעביר את תוכן הרשימה לרשימה אחרת ועבודה עליה. פתרון בגישה כזאת יוביל להורדה של כמעט כל הנקודות על השאלה.**

- מותר לשיטות שאתם כותבים לשנות את תוכן או את מבנה הרשימה עליה אתם עובדים. עם זאת כאשר השיטה מסיימת את עבודתה, המבנה ותוכן הרשימה חייבים להיות בדיוק כמו שהיו לפני הקריאה לשיטה.
- אסור להוסיף תכונות למחלקות IntNodeTwo ו-IntListTwo. קבועים מותר להוסיף.
- אפשר להניח שהפרמטר שמתקבל לכל אחת מהשיטות אינו null. אי אפשר להניח מעבר לכך כל הנחה שהיא לגבי גודל הרשימה. כלומר, יכול להיות שהרשימה תהיה ריקה (כלומר head_ יצביע על null).
- כתבו באנגלית API לכל השיטות הציבוריות שלכם. הוסיפו תיעוד רגיל בתוך השיטות.
- ניתן להשתמש בשיטות עזר פרטיות ככל הנדרש. לשיטות אלו עליכם לכתוב תיעוד רגיל בלבד, לא API, הן לפני כל שיטה (מה היא עושה) והן בתוכה.
- אין להשתמש בשום מחלקה בג'אווה ובשום מרכיבים של השפה שלא נלמדו במהלך הקורס. אי עמידה בכלל זה יוביל להורדה משמעותית ביותר של נקודות.

הערות חשובות לגבי שאלות 1 ו- 2:

- השיטות שתכתבו צריכות להיות יעילות ככל הניתן מבחינת סיבוכיות הזמן. תשובה שאינה יעילה מספיק כלומר, שתהיה בסיבוכיות זמן גדולה יותר מזו הנדרשת לפתרון הבעיה תקבל מעט נקודות בלבד.
- בחישוב הסיבוכיות צריך לחשב גם את הזמן של שיטות העזר, ככל שתרצו לכתוב כאלו.
- סיבוכיות המקום של השיטות שלכם צריכה להיות קבועה : O(1) . בין השאר, אין לכתוב פתרון רקורסיבי לשאלות אלו.
- כתבו (באנגלית בלבד) כחלק מה- API של השאלה מה סיבוכיות הזמן (Time complexity) של השיטות שכתבתם. הסבירו תשובתכם.

שאלה 1 - 25 נקודות

אמיר ותמר משחקים במשחק. על לוח המשחק מונחים K מטבעות בשורה. **K הוא מספר זוגי.** לכל מטבע בשורה יש ערך שהוא מספר חיובי (ממש, גדול מ- 0), והשחקנים יודעים בתחילת המשחק את ערכיהם של כל המטבעות.

המשחק מתנהל בתורות, לסירוגין. בכל תור, השחקן שמשחק בוחר מטבע מאחד משני קצוות השורה ולוקח אותו לקופה שלו. לאחר K תורות נגמרים המטבעות בשורה. בשלב זה סופרים את סכום ערכי המטבעות שבקופה של כל אחד מהשחקנים. השחקן שצבר סכום גדול יותר, מנצח במשחק. במקרה של שוויון בסכומים, המשחק מוכרז כתיקו.

אמיר משחק ראשון. הוא ממש לא רוצה להפסיד. לא אכפת לו אם המשחק יסתיים בתיקו או בניצחון שלו. מצאו אסטרטגיה עבור אמיר שתבטיח שהוא לא יפסיד במשחק.

כאשר תורה של תמר, היא **תמיד** תיקח את הגדול מבין שני המטבעות שבקצוות.

כמובן שאמיר יכול לחשב מראש את עץ המהלכים המלא של המשחק: איך תמר יכולה להגיב לכל מהלך שלו, ואז איך הוא יגיב לכל מהלך שלה, וכו׳. הבעיה בפתרון כזה היא שהעץ הזה עצום: מספר המשחקים השונים שאמיר ותמר יכולים לשחק הוא 2^K ואם K גדול, גם למחשב החזק ביותר בעולם, החישוב הזה ייקח טריליוני שנים.

לכן, אנחנו מעוניינים בפתרון **יעיל** לבעיה – כזה שדורש מאמיר לבצע מעט פעולות כדי לחשב לעצמו אסטרטגיה. כזכור, K (מספר המטבעות) הוא זוגי, כדי שלא יהיה יתרון במספר המטבעות לאחד מהצדדים.

נניח שרשימה מקושרת מסוג IntListTwo מכילה את ערכי המטבעות לפי סדר הופעתם בשורה שעל לוח המשחק.

הוסיפו למחלקה IntListTwo שיטה בשם **win** המבטיחה שאמיר יגיע אל הרווח המירבי בסוף המשחק. מותר לכם להניח שאמיר יודע מהי האסטרטגיה של תמר כפי שהוזכרה לעיל.

השיטה מדפיסה את בחירות השחקנים בכל שלב במשחק. בסוף התהליך יודפס מה היה הסכום הכולל של אמיר ומה הסכום הכולל של תמר.

השיטה מחזירה את ההפרש בין מה שהרוויח אמיר למה שהרוויחה תמר.

חתימת השיטה היא :

```
public int win ()
```

והיא מופיעה ריקה בקובץ IntListTwo.java המצורף לממ״ן זה. מלאו אותה בקוד שלכם.

לדוגמא, אם רשימת המטבעות שבמערך היא (הערך משמאל הוא head_ של הרשימה, הערך מימין הוא tail_ של הרשימה) :

15 ↔ 19 ↔ 21 ↔ 13 ↔ 14 ↔ 30 ↔ 23 ↔ 16 ↔ null

הפלט המצופה ייראה כך :

Amir took 16

Tamar took 23

Amir took 30

Tamar took 15

Amir took 19

Tamar took 21

Amir took 13

Tamar took 14

Final Score:

Amir total 78

Tamar total 73

והשיטה תחזיר 5, תוצאת החיסור 73 – 78.

אפשר להניח שברשימת המטבעות יש לפחות שני מטבעות, שאורך הרשימה זוגי, וכן שערכי כל המטבעות גדולים ממש מ-0.

ושוב, קראו את ההערות הרשומות לעיל, הן בהתייחס לכל השאלות והן בהתייחס לשאלות היעילות (1 ו-2) !

שאלה 2 - 25 נקודות

מצאו במחלקה IntListTwo את השיטות what ו-f.

א. מה מבצעת השיטה what על רשימה מלאה במספרים ? הסבירו בקצרה **מה** מבצעת השיטה ולא כיצד היא מבצעת זאת. כלומר, כתבו מה המשמעות של הערך המוחזר מהשיטה what, ומה מודפס על הפלט? (3 נק')

ב. מה סיבוכיות הזמן והמקום הכוללת של השיטה what ? (1 נק')

ג. כתבו במחלקה IntListTwo שיטה betterWhat שמקבלת גם היא פרמטר יחיד int num, כך שתבצע בדיוק את מה שמבצעת what, אך בסיבוכיות זמן ריצה טובה יותר. השיטה מופיעה ריקה בקובץ IntListTwo.java המצורף לממ"ן זה. מלאו אותה בקוד שלכם. (20 נק')

ד. מה סיבוכיות זמן הריצה של השיטה שכתבתם בסעיף ג ? הסבירו תשובתכם. (1 נק')

את התשובות לסעיפים א ב ד כתבו ב-API של השיטה betterWhat.

אפשר להניח שברשימה עליה עובדת השיטה יש לפחות צומת אחד, וכן שהערכים שבצמתים הם רק חיוביים (גדולים מ-0.)

ושוב, קראו את ההערות הרשומות לעיל, הן בהתייחס לכל השאלות והן בהתייחס לשאלות היעילות (1 ו-2) !

הערות חשובות לגבי שאלות 3 ו-4 :

- **השיטות שתכתבו צריכות להיות ללא כל שימוש בלולאות מכל סוג שהוא, כלומר עליהן להיות רקורסיביות.**
- **מותר לכם לכתוב שיטות עזר, אך גם הן חייבות להיות ללא כל שימוש בלולאות מכל סוג שהוא.**
- **כל שימוש בלולאה בתשובה לאחת מהשאלות 3 או 4 יוביל להורדה משמעותית ביותר של נקודות.**
- **אפשר להשתמש בהעמסת-יתר (overloading).**
- **אין צורך לדאוג ליעילות השיטות הרקורסיביות, ואין צורך לציין מה סיבוכיות הזמן שלהן. בקורסים מתקדמים יותר תלמדו כיצד לשלב רקורסיה ויעילות.**

שאלה 3 - 25 נקודות

נתונה רשימה של מספרים שלמים list. **נגדיר** ש-list1 היא **רשימת-בת** (sub-list) של list אם כל הערכים שנמצאים ב-list1 נמצאים גם ב-list באותו סדר בדיוק, אבל הם לא חייבים להיות רצופים.

למשל אם הרשימה list היא 5 ↔ 4 ↔ 3 ↔ 2 ↔ 1, אזי כל אחת מהרשימות הבאות היא **רשימת-בת** שלה :

1, 2↔3, 1↔3↔4, 2↔5, 1↔2↔3↔4↔5

לעומת זאת הרשימות הבאות אינן רשימות-בת של list :

3↔2, 1↔1↔2, 6, 5↔4↔3↔2↔1, 1↔2↔2↔3

נתונות שתי רשימות של מספרים שלמים, list1 ו-list2 . אנו מחפשים את רשימת-הבת הגדולה ביותר המשותפת ל-list1 ול-list2 .

כיתבו במחלקה IntListTwo שיטה ציבורית longestCommonSublist המקבלת כפרמטר רשימה, ומחזירה את האורך המקסימלי של רשימת-הבת המשותפת לרשימה עליה מופעלת השיטה (this) ולרשימה שהשיטה מקבלת כפרמטר.

חתימת השיטה היא:

```
public int longestCommonSublist(IntListTwo list2)
```

והיא מופיעה ריקה בקובץ IntListTwo.java המצורף לממ״ן זה. מלאו אותה בקוד שלכם.

לדוגמה בהינתן הרשימות הבאות :

L1 : 1↔2↔-3↔4↔5↔-6↔7↔8↔9

L2 : 2↔-3↔10↔0↔-15↔-6↔8

L3 : -15↔10↔4

L4 : -56↔156↔-12

הערך שהשיטה longestCommonSublist תחזיר על L1 עם L2 הוא 4
(האורך של 2↔-3↔-6↔8).

הערך שהשיטה longestCommonSublist תחזיר על L2 עם L3 הוא 1 (האורך של 15- או 10, בשני המקרים רשימה עם איבר בודד).

הערך שהשיטה longestCommonSublist תחזיר על L4 עם כל אחת מהרשימות האחרות הוא 0, כיון שאין להן שום ערך במשותף.

אפשר להניח שברשימה עליה עובדת השיטה יש לפחות צומת אחד. הערכים שבצמתים יכולים להיות כלשהם : חיוביים, שליליים ואפילו 0.

שאלה 4 - 25 נקודות

כתבו במחלקה IntListTwo שיטה ציבורית maxEqualValue המחזירה את האורך המקסימלי של תת-רשימה **רציפה** גדולה ביותר שהיא חלק מהרשימה עליה מופעלת השיטה (this), ואשר כל הערכים בה זהים.

לדוגמה,

אם הרשימה עליה מופעלת השיטה היא 2 ↔ 0 ↔ 3- ↔ 3- ↔ 3- ↔ 3- ↔ 1- ↔ 1- ↔ 1- ↔ 1-

השיטה תחזיר 4, כיון שבין תתי הרשימות הרציפות שכל ערכיהן זהים הארוכה ביותר היא

3- ↔ 3- ↔ 3- ↔ 3- שאורכה 4.

אם הרשימה עליה מופעלת השיטה היא 7 ↔ 9- ↔ 7 השיטה תחזיר 1 כיון שאין בה שני ערכים זהים רצופים זה לזה.

חתימת השיטה היא :

```
public int maxEqualValue()
```

והיא מופיעה ריקה בקובץ IntListTwo.java המצורף לממ״ן זה. מלאו אותה בקוד שלכם.

אפשר להניח שברשימה עליה עובדת השיטה יש לפחות צומת אחד. הערכים שבצמתים יכולים להיות כלשהם : חיוביים, שליליים ואפילו 0.

ושוב, קראו את ההערות הרשומות לעיל, הן בהתייחס לכל השאלות והן בהתייחס לשאלות הרקורסיה (3 ו-4) !

הגשה:

1. הגשת הממ"ן נעשית בצורה אלקטרונית בלבד, דרך מערכת שליחת המטלות.
2. הקפידו ששמות המחלקות והשיטות יהיו בדיוק כפי שמוגדר בממ"ן. **אחרת המחלקה לא תעבור קומפילציה והציון יהיה 0.**
3. עליכם להריץ את הטסטר שנמצא באתר הקורס על המחלקה שכתבתם. שימו לב שהטסטר לא מכסה את כל האפשרויות, ובפרט לא את מקרי הקצה. הוא רק בודק את השמות של השיטות במחלקה. מאד מומלץ להוסיף לו בדיקות. שימו לב שאם הטסטר לא יעבור קומפילציה מול המחלקה שכתבתם, הציון על המטלה יהיה אפס. אם יש שיטה שאתם מעוניינים לדלג עליה, עליכם לרשום את חתימת השיטה ולהחזיר ערך סתמי על מנת שהטסטר יעבור קומפילציה.
4. **גם במטלה זו – אם הוספתם הדפסות שלא ביקשנו בשיטות שכתבתם, כדי להיעזר בהן בפתרון השאלה, עליכם למחוק הדפסות אלו לפני ההגשה. הדפסות מיותרות כאלו יורידו בניקוד.**
5. את התשובות לכל השאלות יש להגיש בקובץ Java ששמו IntListTwo שאנחנו שמנו באתר. אין להוסיף אף קובץ אחר!
6. שימו לב שהקובץ שאתם שולחים חייב להיות אותו הקובץ שאנחנו שמנו באתר, רק עם התוספות שלכם.
7. שימו לב שהתכונות של ראש וזנב הרשימה בקובץ IntListTwo לא הוגדרו ב-private. זאת לא טעות אלא נועד להקל על בדיקת הממ"ן שלכם. כלומר, באופן יוצא דופן (ובניגוד למה שעליכם לעשות בכל הקשר אחר) בראש המחלקה שלכם צריך להופיע בדיוק כך :

```
IntNodeTwo _head, _tail;
```

ולא

```
private IntNodeTwo _head, _tail;
```

8. **אין להגיש את קובץ ה- API שכתבתם, וגם לא את הקובץ IntNodeTwo**
9. **ארזו את הקובץ IntListTwo.java בקובץ zip יחיד ושלחו אותו בלבד.**

ב ה צ ל ח ה