מטלת מנחה (ממ"ן) 13

הקורס: 20441 - מבוא למדעי המחשב ושפת Java

חומר הלימוד למטלה: יחידות 9-10 **נושא המטלה:** יעילות ורקורסיה

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

סמסטר: 2023א **מועד אחרון להגשה:** 21.1.2023

השאלות במטלה זו לקוחות מבחינות גמר שונות או דומות לשאלות של בחינות גמר. אנו ממליצים מאד לענות עליהן ללא הרצה במחשב (כפי שמקובל בבחינת הגמר) ולאחר מכן להריץ.

את התשובות לכל השאלות עליכם לכתוב במחלקה אחת בשם Ex13 (בדיוק). את התשובות לשאלות על הסיבוכיות כתבו (באנגלית בלבד) **כחלק מה- API של השאלה הרלוונטית.**

שאלה 1 – 25 נקודות

בהינתן סדרה כלשהי של **2n** סיביות (ביטים – bits), שיש בה **n אפסים ו- n אחדים**, אפשר לחשב את המספר המינימלי של החלפות בין שתי סיביות הדרושות כדי ליצור את הסדרה המסורגת (alternating sequence) של סיביות, כלומר 01…01010101 או 10…10101010.

לדוגמא, עבור הסדרה 00011011 מספר המינימלי של ההחלפות יהיה 2, כיון שמספיקות שתי החלפות בין סיביות כדי להגיע לסדרה 01010101: החלפה אחת היא בין הסיביות המודגשות 0001**10**11 וההחלפה השנייה היא בין הסיביות המודגשות 0**0**0110**1**1 (יכולות להיות גם שתי החלפות אחרות, למשל אחת היא 0**0**01**1**011 והשנייה היא 00011**01**1)

עליכם לכתוב שיטה סטטית המקבלת מחרוזת תווים שיש בה 2n תווים שהם אך ורק '0' או '1', **והיא מכילה n תווי '0' ו-n תווי '1' בדיוק. אפשר להניח זאת ואין צורך לבדוק זאת!**

השיטה צריכה להחזיר את המספר המינימלי של ההחלפות כמתואר לעיל.

חתימת השיטה היא:

```
public static int alternating (String s)
```

שימו לב, לא משנה אם ההחלפות יביאו את המחרוזת למצב "0101…01" או "1010…10". המספר המוחזר מהשיטה צריך להיות המינימלי מבין אלו שיביאו למצב אחד **או** למצב השני. לדוגמא, עבור המחרוזת "00101011", כדי להגיע למחרוזת "01010101" דרושות שלוש החלפות, בעוד שלהגיע למחרוזת "10101010" מספיקה החלפה אחת בלבד (התו הראשון והאחרון), ולכן השיטה צריכה להחזיר 1.

בפתרון הבעיה אסור להשתמש בשיטות מהמחלקה String, פרט לשיטות charAt ו- length , בהן מותר בשימוש.

שימו לב:
השיטה שתכתבו צריכה להיות יעילה ככל הניתן, גם מבחינת סיבוכיות הזמן וגם מבחינת סיבוכיות המקום. תשובה שאינה יעילה מספיק כלומר, שתהיה בסיבוכיות גדולה יותר מזו הנדרשת לפתרון הבעיה תקבל מעט נקודות בלבד.
ניתן להשתמש בשיטות עזר ככל הנדרש. בחישוב הסיבוכיות צריך לחשב גם את הזמן והמקום של שיטות העזר.

כתבו (באנגלית בלבד) כחלק מה- API של השאלה מה סיבוכיות הזמן (Time complexity) וסיבוכיות המקום (Space complexity) של השיטה שכתבתם. הסבירו תשובתכם.
אל תשכחו לתעד את מה שכתבתם!

שאלה 2 - 25 נקודות

נתונות השיטות הסטטיות הבאות:

```
private static int f (int[]a, int low, int high)
{
    int res = 0;
    for (int i=low; i<=high; i++)
        res += a[i];
    return res;
}

public static int what (int []a)
{
    int temp = 0;
    for (int i=0; i<a.length; i++)
    {
        for (int j=i; j<a.length; j++)
        {
            int c = f(a, i, j);
            if (c%2 == 0)
            {
                if (j-i+1 > temp)
                    temp = j-i+1;
            }

        }
    }
    return temp;
}
```

- המשך השאלה בעמוד הבא -

ענו על ארבעת הסעיפים הבאים(את התשובות לסעיפים א, ב ו-ד רשמו (באנגלית בלבד) ב- API של השיטה):

א. מה מבצעת השיטה what בהינתן לה מערך a מלא במספרים שלמים (חיוביים, שליליים ואפסים)? הסבירו בקצרה **מה** מבצעת השיטה ולא כיצד היא מבצעת זאת. כלומר, כתבו מה המשמעות של המספר המוחזר מהשיטה what.

ב. מהי סיבוכיות הזמן **וסיבוכיות המקום** של השיטה what ?

ג. כתבו את השיטה what כך שתבצע את מה שביצעה בסעיף א בסיבוכיות זמן ריצה קטנה יותר.

שימו לב:

השיטה שתכתבו צריכה להיות יעילה ככל הניתן, גם מבחינת סיבוכיות הזמן וגם מבחינת סיבוכיות המקום. תשובה שאינה יעילה מספיק כלומר, שתהיה בסיבוכיות גדולה יותר מזו הנדרשת לפתרון הבעיה תקבל מעט נקודות בלבד. ניתן להשתמש בשיטות עזר ככל הנדרש. בחישוב הסיבוכיות צריך לחשב גם את הזמן והמקום של שיטות העזר.

ד. מה סיבוכיות זמן הריצה **וסיבוכיות המקום** של השיטה שכתבתם בסעיף ג? הסבירו תשובתכם.

אל תשכחו לתעד את מה שכתבתם!

שאלה 3- 25 נקודות

נתון מערך שמלא במספרים **שלמים חיוביים ממש**.
נגדיר מסלול חוקי במערך כסדרה של אינדקסים במערך, המתחילה באינדקס 0 (התא הראשון במערך) ומתקדמת במערך מספר צעדים ימינה או שמאלה לפי הערך שבתא. שימו לב שאי אפשר להתקדם מעבר לגבולות המערך. המסלול צריך להסתיים בתא האחרון במערך.

כתבו שיטה **רקורסיבית** בוליאנית המקבלת מערך מלא **במספרים שלמים חיוביים ממש**, ומחזירה true אם ישנו מסלול חוקי במערך, ו- false אחרת. **אם יש תא אחד במערך, השיטה תחזיר true.**
חתימת השיטה היא:

```
public static boolean isWay(int[] a)
```

דוגמאות:

- עבור המערך:

0	1	2	3	4	5	6	7	8
2	4	1	6	4	2	4	3	5

התשובה שתוחזר תהיה true שכן ישנו מסלול שמתחיל בתא 0, הולך שני תאים ימינה לתא 2, משם תא אחד שמאלה לתא 1, ומשם ארבעה תאים ימינה לתא 5, משם שוב שני תאים ימינה לתא 7, משם שלושה תאים שמאלה לתא 4 ומשם ארבעה תאים ימינה לתא 8 שהוא האחרון במערך.

- עבור המערך:

0	1	2	3	4	5	6
1	4	3	1	2	4	3

התשובה שתוחזר תהיה false שכן אין אף מסלול שמתחיל בתא 0 ומגיע לתא 6 שהוא האחרון במערך, לפי הקפיצות ימינה או שמאלה. אם נתחיל בתא 0, נזוז תא אחד ימינה לתא אחד, משם חייבים לזוז ימינה ארבעה תאים, כי שמאלה אי אפשר בגלל גבולות המערך, וכך מגיעים לתא 5, משם חייבים לזוז שמאלה ארבעה תאים, כי ימינה אי אפשר, (שימו לב שמתא 5 אין אפשרות התקדם 4 צעדים ימינה, כיוון שתא 6 הוא כבר סוף המערך) ולכן מגיעים שוב לתא 1, כך שזהו תהליך אינסופי שלא מגיע לתא 6 לעולם.
שימו לב להימנע מרקורסיות אינסופיות כאלו.

השיטה צריכה להיות רקורסיבית ללא שימוש בלולאות כלל. כך גם כל שיטות העזר שתכתבו (אם תכתבו) לא יכולות להכיל לולאות.
מותר להשתמש במערך עזר.

מותר לשנות את המערך a במהלך השיטה, אבל חובה להחזיר אותו למצבו ההתחלתי בסופה.

אפשר להשתמש בהעמסת-יתר (overloading).
אסור להשתמש במשתנים סטטיים (גלובליים)!

שאלה 4- 25 נקודות

הנסיך מקפץ על גגות העיר. מטרתו למצוא את הרשע ולהילחם בו, על מנת להציל את הנסיכה.
מפת הגבהים של גגות העיר מיוצגת על-ידי מערך דו-ממדי **ריבועי** המכיל מספרים שלמים.
הניחו כי אם התא מכיל מספר אי-שלילי, הוא מייצג גובה של גג. המיקום של הרשע מסומן על-ידי המספר 1-.
בכל צעד הנסיך יכול להתקדם למשבצת סמוכה: צפונה, דרומה, מזרחה או מערבה (לא באלכסון).
אם המשבצת הסמוכה נמצאת באותו גובה – יכול הנסיך ללכת אליה.
בנוסף, יכול הנסיך לטפס על גג בגובה יחידה אחת, או לרדת מגג בגובה יחידה אחת או שתיים. אם הוא מנסה לטפס לגובה של יותר מיחידה אחת או לרדת לגובה של יותר משתי יחידות, הוא נפסל מיד. כשהנסיך נמצא על גג סמוך לרשע (אחד מארבעת שכניו) הוא יכול לקפוץ אליו ללא קשר להפרש הגבהים בינו לבין הרשע.

עליכם לכתוב שיטה סטטית רקורסיבית שתתכנן את המסלול שיביא את הנסיך לרשע במספר המשבצות הנמוך ביותר בלי להיפסל.
אפשר להניח כי המערך מייצג בצורה נכונה את הגבהים של גגות העיר. כל הערכים בתאים הם מספרים אי-שליליים ויש רק תא אחד שמחזיק מספר שלילי והוא 1-. **אין תאים נוספים במערך שיש בהם מספרים שליליים.**

חתימת השיטה היא:

```
public static int prince(int[][] drm, int i, int j)
```

כאשר מפת הגבהים של גגות העיר נתונה על-ידי הפרמטר drm (Digital Roof Map) ואילו i ו- j מציינים את אינדקס השורה והעמודה בהתאמה של התא בו מתחיל הנסיך.
על השיטה להחזיר את מספר התאים שעל הנסיך לעבור במסלול הקצר ביותר או 1- אם אין מסלול חוקי כזה. לאחר ריצת השיטה על המפה (המערך) להישאר ללא שינוי.

לדוגמא, עבור המפה הבאה:

	0	1	2	3	4
0	2	0	1	2	3
1	2	3	5	5	4
2	8	**-1**	6	8	7
3	3	4	7	2	4
4	2	4	3	1	2

הרשע נמצא בתא בשורה 2 בעמודה 1.

אם הנסיך נמצא בתא (0,0) קיימים שלושה מסלולים המובילים אותו לרשע. הם מסומנים במפות להלן:

2	0	1	2	3
2	**3**	5	5	4
8	**-1**	6	8	7
3	4	7	2	4
2	4	3	1	2

מסלול באורך 4

2	**0**	**1**	2	3
2	3	**5**	**5**	**4**
8	**-1**	**6**	8	7
3	4	7	2	4
2	4	3	1	2

מסלול באורך 10

2	**0**	**1**	**2**	**3**
2	**3**	**5**	**5**	**4**
8	**-1**	6	8	7
3	4	7	2	4
2	4	3	1	2

מסלול באורך 10

לכן השיטה תחזיר 4.

באותה מפה, אם הנסיך נמצא בתא (4,4) אין מסלול חוקי בו הוא יכול ללכת עד הרשע ולכן השיטה תחזיר 1-.

השיטה צריכה להיות רקורסיבית ללא שימוש בלולאות כלל. כך גם כל שיטות העזר שתכתבו (אם תכתבו) לא יכולות להכיל לולאות.

אפשר להשתמש בהעמסת-יתר (overloading).

אסור להשתמש במשתנים סטטיים (גלובליים)!

מותר לשנות את המערך drm במהלך השיטה, אבל חובה להחזיר אותו למצבו ההתחלתי בסופה.

שימו לב:

בשאלות 3 ו- 4 אין צורך לדאוג ליעילות השיטה שתכתבו! אבל כמובן שצריך לשים לב לא לעשות קריאות רקורסיביות מיותרות!

בכל השאלות - אל תשכחו לתעד (באנגלית בלבד) את מה שכתבתם!

שימו לב ששמנו טסטר באתר הקורס .חובה שטסטר ירוץ ללא שגיאות קומפילציה עם המחלקה שלכם .אם יש שיטה שלא כתבתם, כתבו חתימה והחזירו ערך סתמי כדי שהטסטרים ירוצו עם המחלקות ללא שגיאות קומפילציה.

אם הטסטר לא ירוץ ללא שגיאות קומפילציה הציון במטלה יהיה אפס ללא אפשרות ערעור.

הגשה

1. הגשת הממ״ן נעשית בצורה אלקטרונית בלבד, דרך מערכת שליחת המטלות.
2. הקפידו ששמות השיטות יהיו **בדיוק** כפי שמוגדר בממ״ן.
3. עליכם לתעד **(באנגלית בלבד)** את כל השיטות שאתם כותבים בתיעוד API ובתיעוד פנימי המסביר מה עשיתם בשיטה. בתיעוד זה כתבו גם מה הסיבוכיות של השיטות (בשאלות 1 ו-2).
4. את התשובות לכל השאלות עליכם לכתוב במחלקה אחת בשם Ex13 **(בדיוק)**. ארזו את הקובץ בתוך קובץ zip. אין לשלוח קבצים נוספים.

בהצלחה

שאלה לא להגשה

לפניכם שני קטעי הקוד (שאינם קשורים זה לזה):

```
int a =3;
while (a <= n)
   a = a*a;
```

```
public void foo (int n, int m)
{
   int i = m;
   while (i > 100)
         i = i/3;
   for (int k=i ; k>=0; k--)
   {
         for (int j=1; j<n; j*=2)
               System.out.print(k + "\t" + j);
         System.out.println();
   }
}
```

מה סיבוכיות זמן הריצה של קטעי הקוד האלו?

להזכירכם – חוקי הלוגריתמים:

$$\log_a m \times n = \log_a m + \log_a n$$

$$\log_a {}^{m}/_{n} = \log_a m - \log_a n$$

$$\log_a n^m = m \times \log_a n$$

שאלה לא להגשה

לפניכם קטע הקוד הבא:

```
public static int foo (int a, int b)
{
   if (a>3)
       return 2 + foo (b-1, a+1);
   if (b<=4)
       return 1 + foo (a-1, b+1);
   return 0;
}
```

לכל אחת מהקריאות הבאות לשיטה foo, ענו אם היא תעצור, ואם כן, מה היא תחזיר.

א. foo (3, 4)

ב. foo (4, 5)

שאלה לא להגשה

התבוננו בשיטות הבאות:

```
public static void f(int [][] a,
                        int a1, int b1, int a2, int b2)
{
     int temp = a[a1][b1] ;
     a[a1][b1] = a[a2][b2] ;
     a[a2][b2] = temp ;
     if (b1 < a[0].length-1)
         f(a, a1, b1+1, a2, b2-1) ;
     else if (a1+1 < a2-1)
         f(a, a1+1, 0, a2-1, a[0].length-1) ;
}

public static void printArray(int[][] a)
{
    for (int i= 0; i< a.length; i++)
    {
        for (int j= 0; j< a[i].length; j++)
         System.out.print (a[i][j] + "\t");
       System.out.println();
     }
 }
```

נניח שנתונה השיטה main הבאה:

```
public static void main (String [] args)
    {
        int[][] arr = {{1, 2, 3, 4}, {5, 6, 7, 8}} ;
        f(arr, 0, 0, arr.length-1, arr[0].length-1) ;
        printArray (arr);
    }
```

1. מה הפלט שתפיק השיטה main?
2. כמה קריאות רקורסיביות מתבצעות בזימון

```
f(arr, 0, 0, arr.length-1, arr[0].length-1) ;
```