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:** 3 נקודות

סמסטר: 2024ב **מועד אחרון להגשה:** 8.6.2024

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

נתון מערך מלא במספרים שלמים, שבו **כל מספר מופיע פעמיים ברצף** פרט למספר אחד שמופיע רק פעם אחת. **המערך אינו ממוין**.

לדוגמא, המערכים הבאים מקיימים את התנאי:

	0	1	2	3	4	5	6	7	8
a =	**6**	**6**	**18**	**18**	**-4**	**-4**	**12**	**9**	**9**

	0	1	2	3	4	5	6	7	8	9	10	11	12
b =	**8**	**8**	**-7**	**-7**	**3**	**3**	**0**	**0**	**10**	**10**	**5**	**5**	**4**

	0
c =	**5**

כתבו שיטה סטטית שמקבלת כפרמטר מערך שמקיים את התנאי הנ"ל, ומחזירה את המספר שמופיע במערך רק פעם אחת.

לדוגמא, במערכים לעיל:

- המספר הבודד במערך a הוא 12 שנמצא באינדקס 6
- המספר הבודד במערך b הוא 4 שנמצא באינדקס 12
- המספר הבודד במערך c הוא 5 שנמצא באינדקס 0

אתם יכולים להניח שהמערך אינו ריק ושהוא מקיים את התנאי, אין צורך לבדוק זאת.

חתימת השיטה היא:

```
public static int findSingle (int [] a)
```

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.

מה סיבוכיות זמן הריצה וסיבוכיות המקום של השיטה שכתבתם? הסבירו תשובתכם.

שאלה 2 - 25 נקודות

נתון כלי לצבירת מי גשם. חתך הכלי מתואר על ידי מערך הגבהים heights.

למשל, עבור מערך הגבהים {2, 1, 1, 4, 1, 1, 2, 3}

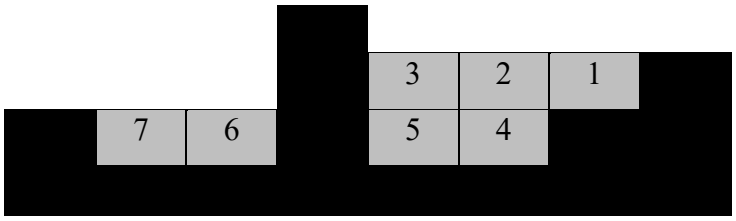
צורת הכלי היא:

כתבו שיטה סטטית המקבלת כפרמטר מערך גבהים ומחזירה את כמות הגשם שניתן לצבור בכלי עם מערך גבהים נתון.

ניתן להניח שכל הערכים במערך (הגבהים) הם מספרים שלמים חיוביים ממש.

למשל, עבור הכלי לעיל, ניתן לצבור 7 יחידות גשם (האיזורים האפורים שבאיור להלן).

רמז: יש לחשב בכל מקום בכלי מה גובה העמודה המקסימלית / המינימלית מימינו ומשמאלו.



חתימת השיטה היא:

```
public static int waterVolume (int [] heights)
```

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 מה סיבוכיות הזמן וסיבוכיות המקום של השיטה שכתבתם. הסבירו תשובתכם.

הערות לגבי שאלות 1 ו- 2:

- **ניתן להשתמש בשיטות עזר ככל הנדרש. בחישוב הסיבוכיות צריך לחשב גם את הזמן והמקום של שיטות העזר.**
- **אסור לשנות את המערכים בשתי השאלות, גם אם הם חוזרים לקדמותם לאחר השיטה!**
- **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 (באנגלית) את מה שכתבתם!**

שאלה 3 - 25 נקודות

נתונה המחלקה Password.

למחלקה Password יש בנאי שמקבל מספר שלם חיובי n. הבנאי מייצר סיסמא אקראית באורך n, המורכבת מאותיות הקטנות של האלפבית האנגלי (בלבד). (הבנאי שומר את הסיסמא הזו כתכונה של האוביקט).

חתימת הבנאי היא:

```
public Password (int length)
```

בנוסף במחלקה קיימות השיטות:

1. שיטה ציבורית בשם getPassword המחזירה את הסיסמא הסודית.

חתימת השיטה היא:

```
public  String getPassword ()
```

2. שיטה ציבורית בוליאנית בשם isPassword המקבלת מחרוזת תווים st, ומחזירה את הערך true אם המחרוזת st זהה בדיוק לסיסמא הסודית שיצר הבנאי, ו- false אחרת.

חתימת השיטה היא:

```
public  boolean isPassword (String  st)
```

עליכם לכתוב שיטה **סטטית רקורסיבית** בשם findPassword שמקבלת אובייקט מהמחלקה Password ואורך סיסמא length. השיטה צריכה לפצח ולהחזיר את הסיסמא.

חתימת השיטה שלכם צריכה להיות:

```
public static String findPassword (Password p, int length)
```

אם נכתוב שיטה main כדלהלן:

```
public static void main(String[] args)
{
  Password p = new Password(5);
  System.out.println (p.getPassword());
  System.out.println (Ex13.findPassword(p,5));
}
```

השיטה main תיצור אובייקט מהמחלקה Password ובו סיסמא סודית, ואז תדפיס את הסיסמא הסודית פעמיים. פעם אחת בעזרת השיטה getPassword ופעם שניה כפי שנמצאה לאחר החיפוש. כמובן שהשיטות יצטרכו להחזיר אותה מחרוזת.

שימו לב, המספר 5 כאן הוא רק דוגמא. אי אפשר להניח שאורך הסיסמא הוא 5. אורך הסיסמא אינו ידוע ואינו מוגבל!
שימו לב, האורך של המחרוזת אינו קבוע, ובכל זאת אנו ממליצים לא לנסות להריץ עם סיסמא שאורכה גדול מ- 6.

המחלקה Password נמצאת באתר הקורס בצורת class בלבד. עליכם להשתמש בה בהנחה שאתם יכולים להשתמש בבנאי ובשיטות שהוזכרו לעיל בלבד!

הערות חשובות:

- השיטה שתכתבו צריכה להיות רקורסיבית ללא לולאות בכלל!
- **רמז** - הדרך למצוא את הסיסמא היא על ידי בדיקת כל האפשרויות לסיסמא באורך n, ובדיקת כל אחת מהאפשרויות, אם היא הנכונה.
- אינכם יכולים לכתוב 26 קריאות רקורסיביות לשיטה. חשבו על דרך אחרת!
- בשיטה findPassword שאתם כותבים אסור לכם להשתמש בשיטה getPassword. השיטה getPassword ניתנת לכם רק כדי שתוכלו לבדוק את עצמכם.
- אם ברצונכם להשתמש בשיטות מהמחלקה String בכתיבת השיטה findPassword, מותר לכם להשתמש אך ורק בשיטות הבאות (ולא באף שיטה אחרת מהמחלקה!):
  charAt, equals, length, substring
  שימו לב שאין חובה להשתמש בשיטות אלו (בחלקן או בכולן), אבל אי אפשר להשתמש בשיטות אחרות מהמחלקה String.
- אסור להשתמש באף מחלקה אחרת!

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

ראו הערה חשובה לאחר שאלה 4.

שאלה 4 - 25 נקודות

נתון מערך דו-ממדי ריבועי (מספר השורות שווה למספר העמודות) שמכיל ערכים בוליאניים – true/false.

נגדיר: **איזור true** במערך (true region), כאוסף מקסימלי של תאים סמוכים שכולם בעלי ערך true. תאים הממוקמים **באלכסון** זה לזה **לא** נחשבים לסמוכים.

למשל, עבור המערך מימין (כאן אנחנו מסמנים את הערך true כ- 1 ואת הערך false כ-0), קיימים 3 איזורי true והם מסומנים במערך משמאל:

0	0	0	0	1
0	1	1	1	0
0	0	1	1	0
1	0	0	0	0
1	1	0	0	0

0	0	0	0	1
0	1	1	1	0
0	0	1	1	0
1	0	0	0	0
1	1	0	0	0

עליכם לכתוב שיטה **רקורסיבית** המקבלת כפרמטר מטריצה ריבועית בוליאנית ומחזירה כמה איזורי true שונים קיימים במטריצה. אם לא קיימים איזורי true יוחזר 0.

שימו לב שאיזור true מורכב לפחות מתא אחד.

חתימת השיטה היא:

```
public static int cntTrueReg (boolean[][]mat)
```

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

אפשר להשתמש בהעמסת יתר (overloading) וכן <u>מותר לשנות את המערך</u>.

הערות לגבי שאלות 3 ו- 4:

- **מותר להשתמש בהעמסת-יתר (Overloading)**
- **אסור להשתמש במשתנים סטטיים (גלובליים)!**
- **אין צורך לדאוג ליעילות השיטה! אבל כמובן שצריך לשים לב לא לעשות קריאות רקורסיביות מיותרות!**
- **אל תשכחו לתעד את מה שכתבתם!**

בשאלה 3 מותר להשתמש בשיטות length, equals, charAt ו- substring מהמחלקה String.

בשאלה 4 מותר לשנות את המערך במהלך השיטה.

שימו לב:

- **בכל השאלות - אל תשכחו לתעד (באנגלית בלבד) את מה שכתבתם!**
- **שמנו טסטר למחלקה Ex13 באתר הקורס. חובה שהטסטר 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.
- **אם הוספתם הדפסות שלא ביקשנו בשיטות שכתבתם, כדי להיעזר בהן בפתרון השאלה, עליכם למחוק הדפסות אלו לפני ההגשה. הדפסות מיותרות כאלו יורידו בניקוד.**

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) ;
```