מבוא למדעי המחשב
עבודה מספר 3

מתרגלים אחראיים: אור עמי, אפרת מילר
תאריך הגשה: לפי הרשום באתר

נושאים:

העבודה עוסקת ברקורסיה ובתכנות מונחה עצמים.

הנחיות:

- לתרגיל מצורפים מספר קבצי שלד (Map.java , Point.java, NumberPartition.java). בכל אחד מחלקי התרגיל עליכם להשלים את השיטות המתאימות בקובץ השלד של אותו חלק. יש להגיש קובץ zip יחיד (השם לא משנה) ובתוכו קבצי השלד (בלבד) המכילים את הקוד שכתבתם, ללא תת-תיקיות. **(הגשה שלא לפי ההוראות עלולה לפגוע בציון!)**

- אין לשנות את חתימות השיטות שבקבצי השלד, שכן הבדיקות האוטומטיות מסתמכות על חתימות אלו. ניתן ואף מומלץ להוסיף שיטות עזר ע"פ הצורך. כל שיטה בקובץ, בין אם סופקה על ידינו ובין אם אתם כתבתם אותה, חייבת להיות מתועדת ע"י הערה.

- שדות ושיטות עזר שהוספתם חייבים להיות מוגדרים כ-private, **ולכלול הערות קצרות מעליהן.**

- אין לכתוב הערות או תווים כלשהם בעברית (או כל שפה אחרת שאינה אנגלית).

- כל עוד לא נאמר אחרת, לא ניתן להניח שום דבר על נכונות הקלט (כלומר, מספר יכול להיות שלילי, מערך ריק בגודל 0, מערך null וכו'). אם שיטה מחזירה ערך מטיפוס boolean והקלט לא תקין, יש להחזיר false. אם שיטה מחזירה ערך מטיפוס שאינו פרימיטיבי והקלט לא תקין, יש להחזיר null. אם שיטה מחזירה ערך מטיפוס int והקלט לא תקין, יש להחזיר 1-. במידה ושיטה אינה מחזירה ערך (או שהיא בנאי) יש להדפיס הודעת שגיאה למסך.

- יש לקרוא כל שאלה עד סופה לפני כתיבת פתרון לשאלה.

- תרגיל אותו לא ניתן להדר (לקמפל) או להריץ יקבל ציון נכשל.

- כל איחור בהגשת התרגיל יגרום להורדת ציון, כפי שמוגדר בנהלים באתר הקורס (במידה ואתם מגישים את העבודה באיחור, לחצו על כפתור "Submit new delay" במערכת ההגשה והגישו את עבודתכם).

- ההגשות יבדקו להעתקות, אין לשתף קוד.

- ההגשה הינה בזוגות.

חלק 1 (70 נקודות)

בחלק זה תממשו אלגוריתם לצביעה רקוריסיבית של מפה, שבה כל אזור צבוע משפיע על האזורים השכנים.
זכרו להוסיף בקוד לפני כל מחלקה וכל שיטה הערות המסבירות במילים שלכם מה המחלקה/ השיטה עושה.

משימה 0 – המחלקה Point

המחלקה Point מייצגת נקודה במרחב, ומורכבת מהשיטות:

```
public Point(int x, int y)
public int getX()
public int getY()
```

כאשר השיטה הראשונה מייצרת נקודה על בסיס מספר השורה (X) ומספר העמודה (Y). ערכי x,y יכולים להיות **כל שני מספרים שלמים**. שתי השיטות הנוספות מחזירות את ערכי הx והy של הנקודה.
השלימו את המחלקה Point.

משימה 1

לצורך המשימה כל אות אנגלית מייצגת לנו צבע (אות גדולה ואות קטנה מייצגות את אותו הצבע).

מפה בגודל n הינה מערך דו-מימדי בגודל $n \times n$ של צבעים מטיפוס char. הגדר את הבנאי של

המחלקה Map היוצר Map באמצעות מערך דו מימדי של תווים. (במידה ואחד התווים במערך אינו אות אנגלית חוקית יש להשתמש באות z במקום).

```
public Map(char[][] map)
```

<u>הערה:</u> ניתן להניח כי map **אינו null**, וכי map הינה **מטריצה מרובעת** (כלומר אורך המימד הראשון שווה לאורך המימד השני, ו**אורך זה גדול ממש מ-0**).

השלם את הקוד של השיטה getMap אשר מחזירה את המפה כמערך דו מימדי:

```
public char[][] getMap()
```

בנוסף, השלם את הקוד של השיטה equals המקבלת אובייקט חדש (מפה חדשה) ובודקת האם היא זהה למפה הנוכחית. (היזהרו מהשוואה "שטחית" של כתובות בלבד! השוואה צריכה לכלול מעבר על כל איברי המפה).

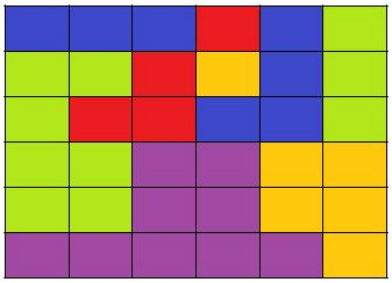
```
public boolean equals(Map map)
```

<u>הערה:</u> לאורך התרגיל נניח כי ראשית הצירים נמצא בפינה השמאלית העליונה במפה (כלומר התא במיקום 0,0).

משימה 2

ממשו את השיטה הבאה, **בתוך המחלקה Map,** אשר מחזירה את מספר הצבעים **השונים** במפה (זכרו את ההערה מעלה: צבעים שונים אם מדובר באותיות שונות, ואין הבדל בין אות גדולה לקטנה). אם המפה ריקה או שערכה null יש להחזיר 0.

```
public int numOfColors()
```



<u>דוגמא</u>: עבור המפה מעלה השיטה תחזיר 5.

משימה 3

אם M היא מפה בגודל n ו- $0 \le i,j < n$ נאמר שנקודה המוגדרת ע"י (i,j) הינה נקודה במפה M שנמצאת בשורה ה-i בעמודה ה-j.

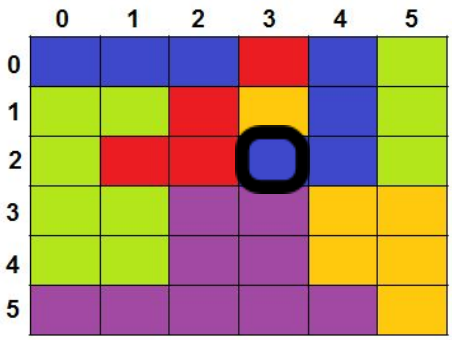
אם $0 < i,j < n-1$ אזי (i,j) הינה **נקודה פנימית** במפה.

ה**שכנים של נקודה** במפה הם שמונת התאים המקיפים אותה.

לכל נקודה שאינה פנימית (כלומר, על המסגרת) יש פחות משמונה שכנים. לדוגמא, לנקודה (0,0) יש שלושה שכנים בלבד: (0,1) (1,0) (1,1)..

ממשו את השיטה הבאה, בתוך המחלקה **Map**, אשר מחזירה את מספר הצבעים השונים של השכנים של הנקודה p כולל צבע הנקודה עצמה.

```
public int numOfColors(Point p)
```



<u>דוגמא</u>:
עבור המפה, הקריאה:
numOfColors(new Point(2,3))
תחזיר 4.

<u>הערה</u>: במקרה שהנקודה נמצאת **מחוץ למפה** יש להחזיר את הערך 0.

משימה 4

<u>הגדרה</u>: נקודה p' הינה **שכן חוקי** של נקודה p אם"ם שתי הנקודות ממוקמות במפה ולא מחוץ לה, $p \neq p'$ (לא מדובר באותו מיקום במפה), הנקודות ממוקמות בשכנות זו לזו, וצבען זהה.

ממשו את השיטה הבאה, בתוך המחלקה Map, אשר מחזירה true אם"ם $(i1,j1)$ הינה שכן חוקי של $(i2,j2)$ במפה map.

```
public boolean legalNeighbor(Point p1, Point p2)
```

דוגמא: עבור המפה שמוצגת באיור לעיל:

Point p1		Point p2		
i	j	i	j	legalNeighbor?
0	0	0	0	false
0	0	0	1	true
0	2	3	4	false
0	3	1	2	true

<u>הערה</u>: לפי ההגדרה מעלה לנקודה מחוץ למפה אין אף שכן חוקי!
<u>הערה 2</u>: זכרו - גם אלכסונים הם שכנים.

משימה 5

<u>הגדרה</u>: נקודה p' הינה **באזור** של נקודה p אם שתי הנקודות ממוקמות במפה ולא מחוץ לה ומתקיים אחד התנאים הבאים:

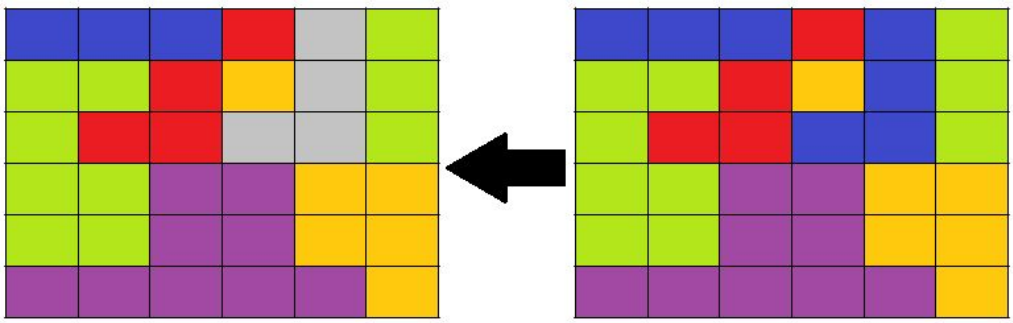
1. $p = p'$ (כלומר שתי הנקודות הן באותו מיקום);
2. p' הינה שכן חוקי של p (לפי ההגדרה במשימה קודמת);
3. p' הינה באזור של אחד השכנים החוקיים של p;

במילים אחרות, נגדיר את ה ***אזור*** של נקודה p להיות איחוד *האזורים* של השכנים החוקיים של p (נשים לב שזוהי הגדרה רקורסיבית).

ממשו את השיטה הבאה בתוך המחלקה Map המקבלת נקודה וצבע. על השיטה לשנות את המפה הנוכחית, ככה שה***אזור*** של הנקודה p יצבע בצבע color.

```
public void fill(Point p, char color)
```

<u>דוגמא:</u>
המפה השמאלית מתקבלת מהפעלת השיטה fill על המפה הימנית בנקודה (2,4) בה מופיע הצבע 'B' (כחול), כאשר הארגומנט color הוא הצבע אפור ('G').



חלק 2 (30 נקודות)

תרגיל זה עוסק בחלוקת סדרת המספרים $[1,2,3,...,n]$ לשתי קבוצות ועל כן שמה NumberPartition . חלוקה תוגדר לפי "**מחרוזות מציינות**" (characteristic strings).

מחרוזת s נקראת מחרוזת מציינת אם היא מכילה רק את התוים '0' ו'1'.

דוגמא: "s = "10010110 היא מחרוזת מציינת (באורך 8).

מחרוזת מציינת s באורך n > 0 מגדירה חלוקה של סדרת n המספרים הטבעיים הראשונים(1, 2, ... n) לשתי קבוצות:

1. $one(s)$
2. $zero(s)$

באופן הבא: עבור i מספר כלשהו ($1 \le i \le n$) האיבר ה-i בסדרה נמצא בקבוצה $zero(s)$ אם $s.charAt(i-1) == 0$ או בקבוצה $one(s)$ אם $s.charAt(i-1) == 1$.

דוגמא:

עבור $n = 5$ והמחרוזת המציינת $s = "10100"$ אזי הקבוצות יהיו:

1. $one(s) = \{1,3\}$
2. $zero(s) = \{2,4,5\}$

סעיף א

כיתבו פונקציה:

```
public static boolean isNumberPartition(int n, String s)
```

אשר מקבלת מספר שלם n ומחרוזת מציינת s באורך השווה ל-n ומחזירה ערך `true` אם ורק אם מתקיימים שלושת התנאים הבאים:

1. מספר האיברים ב- $one(s)$ שווה למספר האברים ב- $zero(s)$.
2. סכום האיברים ב- $one(s)$ שווה לסכום האיברים ב-$zero(s)$.
3. סכום ערכי הריבועים של אברי $one(s)$ שווה לסכום ערכי הריבועים של אברי $zero(s)$.

שימו לב: בשאלה זו אין להניח שהקלט תקין! עבור קלט שאינו תקין (למשל כאשר n שונה מאורך המחרוזת s), על הפונקציה להחזיר false. חשבו היטב על כל המקרים השונים בהם קלט לא יהיה תקין!

דוגמאות:

א. אם $n = 4$ ו-$s = "1010"$ אז קריאה לפונקציה תחזיר את הערך `false` מכיוון שתנאי 2 אינו מתקיים:

$one(s) = \{1,3\}, zero(s) = \{2,4\}$

$1+3 \neq 2+4$

ב. אם $n=8$ ו- $s=$ "01101001" אז קריאה לפונקציה תחזיר את הערך `true` מכיוון שמתקיים כי:

$one(s) = \{2,3,5,8\},\ zero(s) = \{1,4,6,7\}$

1. $|one(s)| = |zero(s)|$ (rule 1)
2. $2+3+5+8 = 1+4+6+7$ (rule 2)
3. $2^2+3^2+5^2+8^2 = 1^2+4^2+6^2+7^2$

ג. אם n=10 ו- s=`null` אז הקריאה תחזיר `false` מכיוון שהקלט אינו תקין.

סעיף ב

השלימו את הגדרת הפונקציה

```
public static void numberPartition(int n)
```

ואת פונקצית העזר הרקורסיבית שהיא קוראת לה. הפונקציה מקבלת מספר שלם n ומדפיסה את **כל** המחרוזות המציינות אשר עומדות בשלושת התנאים של הפונקציה isNumberPartition מסעיף א'.
סדר ההדפסה אינו משנה, כל מחרוזת מציינת צריכה להופיע בשורה חדשה. אם לא קיימת אף מחרוזת כזאת הפונקציה לא מדפיסה דבר.

```
public static void numberPartition(int n){
    if (n > 0) {
        numberPartition(/*השלימו*/);
    }
}

private static void numberPartition(int n, הוסיפו עוד ארגומנטים){
    //השלימו .פונקצית עזר.
}
```

דוגמאות:

א. אם $n=7$ אז קריאה לפונקציה לא תדפיס דבר מכיוון שלא קיימות מחרוזות מציינות עבור ערכים אלו של העומדת בתנאים של סעיף א'. (חשבו: למה?)
ב. אם $n=8$ אז קריאה לפונקציה תדפיס את שתי המחרוזות הבאות:

01101001
10010110

עבורן מתקיימים שלושת התנאים מסעיף א' (ראו בסעיף א' את דוגמא ב').
עבור קלט לא תקין הפונקציה לא תדפיס דבר.
זיכרו כי ביכולתכם לוודא כי הקוד שאתם כותבים נכון. כיתבו לעצמכם תוכנית בדיקה מתאימה.

בהצלחה!