מטלת מנחה (ממ"ן) 15

הקורס: 20441 - מבוא למדעי המחשב ושפת Java

חומר הלימוד למטלה: יחידות 17 - 18 **נושא המטלה:** רשימה מקושרת

מספר השאלות: 3 **משקל המטלה:** 5 נקודות

סמסטר: 2020ב **מועד אחרון להגשה:** 27.6.2020

(ת)

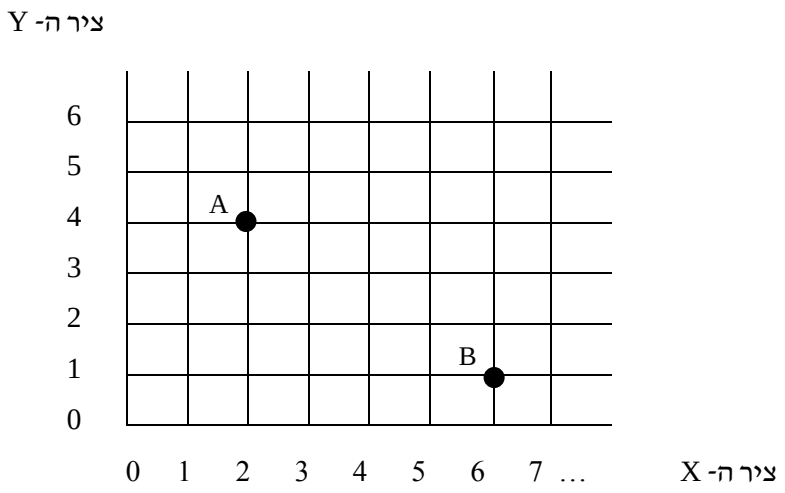
שאלה 1 - להרצה (10%)

המחלקה Point מייצגת נקודה במישור, לפי מערכת הצירים הקרטזית -

למחלקה Point יש את התכונות הפרטיות (instance variables) הבאות:

- double _x – שמייצגת את המיקום על פני ציר ה- X;
- double _y – שמייצגת את המיקום על פני ציר ה- Y.

לדוגמא, הנה מסומנות שתי הנקודות A = (2,4) ו- B = (6,1) במישור:



למחלקה Point הוגדרו שני **בנאים** (constructors):

- האחד - בנאי המקבל שני פרמטרים המהווים את ערכי התכונות שיהיו לנקודה.

public Point(double x, double y)

- השני - בנאי העתקה המקבל נקודה אחרת, ומעתיק את ערכיה.

public Point (Point other)

בנוסף הוגדרו במחלקה השיטות:

- שיטות **האחזור**:
  - double getX() המחזירה את ערכה של קואורדינטת ה- x.
  - double getY() המחזירה את ערכה של קואורדינטת ה- y.

- השיטות **הקובעות**:
  - void setX (double num) המשנה את ערכה של קואורדינטת ה- x להיות num.
  - void setY (double num) המשנה את ערכה של קואורדינטת ה- y להיות num.
- השיטה toString שמחזירה את תוכן האובייקט כמחרוזת תווים לפי הייצוג המתמטי המקובל - (x,y). כך, המחרוזת (3.0,4.0) מייצגת את הנקודה שקואורדינטת ה- x שלה היא 3.0 וקואורדינטת ה- y שלה היא 4.0. **שימו לב לדייק במחרוזת לפי הכתוב כאן**. ללא רווחים וללא תווים נוספים.
- boolean equals (Point other) – שיטה שמקבלת נקודה כפרמטר ומחזירה האם הנקודה שעליה הופעלה השיטה והנקודה שהתקבלה כפרמטר זהות.
- boolean isAbove (Point other) - שיטה שמקבלת נקודה כפרמטר ומחזירה האם הנקודה שעליה הופעלה השיטה נמצאת **מעל** לנקודה שהתקבלה כפרמטר. (באיור למעלה, הנקודה A נמצאת מעל לנקודה B)
- boolean isUnder (Point other) - שיטה שמקבלת נקודה כפרמטר ומחזירה האם הנקודה שעליה הופעלה השיטה נמצאת **מתחת** לנקודה שהתקבלה כפרמטר. **השיטה הזו משתמשת אך ורק בשיטה isAbove שהוגדרה לעיל.** אי אפשר להשתמש באף שיטה אחרת ובשום אופרטור נוסף. מותר להשתמש **אך ורק בקריאה לשיטה isAbove**.
- boolean isLeft (Point other) - שיטה שמקבלת נקודה כפרמטר ומחזירה האם הנקודה שעליה הופעלה השיטה נמצאת **משמאל** לנקודה שהתקבלה כפרמטר. (באיור למעלה, הנקודה A נמצאת משמאל לנקודה B)
- boolean isRight (Point other) - שיטה שמקבלת נקודה כפרמטר ומחזירה האם הנקודה שעליה הופעלה השיטה נמצאת **מימין** לנקודה שהתקבלה כפרמטר. **השיטה הזו משתמשת אך ורק בשיטה isLeft שהוגדרה לעיל.**
- double distance (Point p) –שיטה שמקבלת נקודה כפרמטר ומחזירה את המרחק בין הנקודה שעליה הופעלה והנקודה שהתקבלה כפרמטר. **לעזרתכם הנוסחה לחישוב מרחק בין הנקודה** (x1,y1) **לנקודה** (x2,y2) **הוא** $\sqrt{(y2-y1)^2+(x2-x1)^2}$.

  על מנת לחשב שורש ריבועי של מספר, ניתן להשתמש בשיטה (Math.sqrt(x , שהיא שיטה של Java שנמצאת במחלקה Math . כדי להשתמש בה אין צורך לייבא אף מחלקה, אלא לקרוא לה בשמה המלא (Math.sqrt(x כאשר במקום הפרמטר x כותבים את הביטוי שממנו רוצים להוציא שורש ריבועי.

  הפרמטר x של השיטה הזו יכול להיות מטיפוס שלם (int) או ממשי (double). השיטה מחזירה מספר ממשי (גם אם השורש הריבועי של x הוא מספר שלם).
- void move (double dx, double dy) – המזיזה את הנקודה ב- dx על ציר ה- X וב- dy על ציר ה- Y.

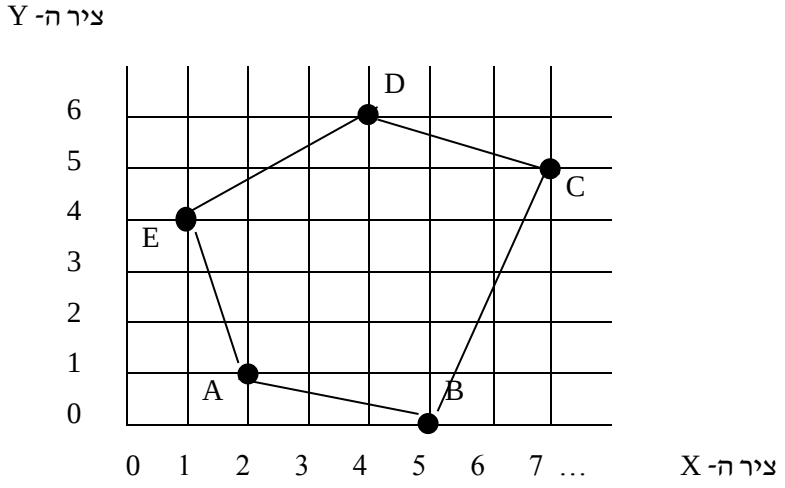
עליכם לכתוב את המחלקה Point לפי ההגדרות לעיל.

נגדיר: מצולע קמור הוא מצולע שכל זוויותיו הפנימיות קטנות מ-180 מעלות. במצולע קמור הקו המחבר כל שתי נקודות מתוך המצולע עובר רק בתוך המצולע.

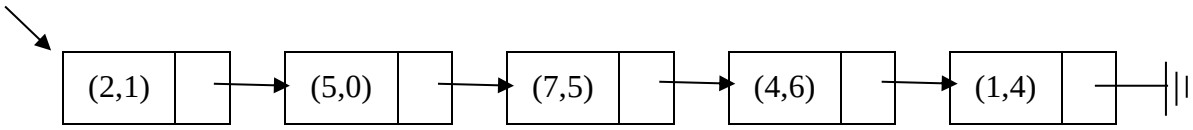
המחלקה Polygon מייצגת מצולע קמור במישור.

הייצוג נעשה על-ידי רשימה מקושרת ששומרת את רשימת הקדקודים (vertices) של המצולע לפי סדר הופעתם במצולע. אין חשיבות מי הקדקוד הראשון. כל קדקוד מיוצג על-ידי נקודה במישור.

הנה דוגמא למצולע קמור. השמות שהצמדנו לנקודות הם רק לשם התיחסות בהמשך.



הרשימה המייצגת את הקדקודים של המצולע היא:



כדי לעשות זאת, עליכם להגדיר שתי מחלקות PointNode ו- Polygon.

שאלה 2 - להרצה (10%)

המחלקה **PointNode** תייצג קדקוד אחד במצולע.

לכל אובייקט במחלקה יש שני שדות:

```
1. Point _point          //הנקודה במישור
2. PointNode  _next      //מצביע לאיבר הבא
```

למחלקה זו עליכם להגדיר שלושה בנאים:

```
1. public PointNode (Point p)
```

בנאי המקבל נקודה , שדה ה-next_ יאותחל ל-null.

```
2. public PointNode (Point p, PointNode n)
```

בנאי המקבל נקודה ואיבר נוסף מטיפוס PointNode, ומאתחל את התכונות לפי הפרמטרים.

```
3. public PointNode (PointNode p)
```

- בנאי העתקה. שימו לב שפה aliasing הוא לא טעות. יש להעתיק את המידע (next) עצמו ולא עותק של המצביע.

השיטות במחלקה PointNode הן:

- `public Point getPoint()` - שיטה המחזירה עותק של הנקודה שבקדקוד.
- `public PointNode getNext()` - שיטה המחזירה מצביע לאיבר הבא. שימו לב שפה aliasing הוא לא טעות. יש להחזיר את המצביע next ולא עותק של המצביע.
- `public void setPoint(Point p)` - שיטה המקבלת נקודה ומעדכנת את תכונת הנקודה שבקדקוד.
- `public void setNext( PointNode next)` - שיטה המקבלת מצביע ומעדכנת את תכונת המצביע לאיבר הבא. שימו לב שפה aliasing הוא לא טעות. יש לעדכן את המידע (next) עצמו ולא עותק.

שאלה 3 - להרצה (80%)

המחלקה Polygon מייצגת מצולע קמור במישור.

הייצוג נעשה על-ידי רשימה ששומרת את רשימת הקדקודים (vertices) של המצולע לפי סדר הופעתם במצולע. אין חשיבות מי הקדקוד הראשון. כל קדקוד מיוצג על-ידי נקודה במישור.

במחלקה זו מותר להגדיר אך ורק תכונה פרטית אחת, ראש הרשימה, שתצביע להתחלת הרשימה. אין להוסיף תכונות מעבר לתכונה זו.

עליכם לממש ב- Java את המחלקה Polygon לפי הסעיפים להלן:

1. הגדרת התכונה של המחלקה.
2. בנאי שיוצר מצולע ריק - מאתחל את ראש הרשימה להיות null.
3. שיטה בוליאנית addVertex שמוסיפה קדקוד למצולע. היא מקבלת כפרמטרים נקודה p, ומספר שלם pos המסמן לאיזה מקום ברשימה תיכנס הנקודה החדשה p.

אם הוספת הקדקוד הצליחה, השיטה תחזיר true, אם לא – השיטה תחזיר false.
עליכם לחשוב מתי השיטה עלולה שלא להצליח להוסיף קדקוד לרשימה.
אפשר להניח שהקדקוד החדש שנוסף לא מקלקל את היות המצולע מצולע קמור, ואין צורך לבדוק זאת. (זה לא יגרום להחזרת false).
כמו כן, ניתן להניח שכאשר מייצגים מצולע קמור ברשימה, מוסיפים את הקדקודים על ידי השיטה addVertex לפי סדר הופעתם במצולע.

הבהרה לגבי המקום ברשימה:

- מספרי המקומות מתחילים ב-1.
- אם המקום המבוקש הינו מקום שכבר תפוס (כלומר יש כבר צומת במקום הזה ברשימה), אזי הצומת החדש "יידחף" במקום החדש (לא יחליף אותו), והצומת שהיה קודם במקום זה יהפוך להיות זה שאחריו.
- אם המקום המבוקש גדול ב-1 מגודלה הנוכחי של הרשימה, הצומת החדש יוכנס בסופה.
- אם השיטה מתבקשת להכניס צומת במקום שהוא קטן מ-1, או במקום שגדול ביותר מ-1 ממספר הצמתים הנוכחי ברשימה, היא תחזיר false.

דוגמאות :

אם הרשימה היא זאת שהמשורטטת בממ"ן בשאלה מספר 1 (עם 5 צמתים), אזי :

- הקריאות ל-addVertex עם מקום שהוא קטן או שווה ל-0, או גדול או שווה ל-7 תחזיר false ולא תשנה את הרשימה.
- קריאה ל-addVertex עם מקום 1 תכניס את הצומת החדש למקום הראשון ברשימה, כך שהוא יצביע על הצומת (1, 2) שקודם לכן היה הראשון ברשימה.

- קריאה ל-addVertex עם מקום 6 תכניס את הצומת החדש למקום האחרון ברשימה, כך שהצומת (4, 1) שקודם לכן היה האחרון ברשימה יצביע עליו.
- קריאה ל-addVertex עם מקום 3 תכניס את הצומת החדש כך שהצומת שבו הנקודה (5,0) יצביע עליו, והצומת החדש יצביע על הצומת (7,5).

4. שיטה highestVertex המחזירה העתק של את הקדקוד שנמצא הכי גבוה במצולע. אם יש יותר מאחד בגובה הגבוה ביותר , היא מחזירה את הראשון בו נתקלה. אם אין קדקודים במצולע (כלומר הרשימה ריקה) השיטה תחזיר null.
5. השיטה toString המחזירה מחרוזת תווים המייצגת את המצולע. המחרוזת צריכה להיות בפורמט הבא: שימו לב שאין רווחים במחרוזת של הקדקודים.

```
The polygon has 5 vertices:
((2,1),(5,0),(7,5),(4,6),(1,4))
```

אם אין קדקודים השיטה תחזיר מחרוזת בפורמט הבא:

```
The polygon has 0 vertices.
```

6. שיטה calcPerimeter המחזירה מספר ממשי (double) המייצג את היקף המצולע.
אם מספר הקדקודים הוא 2 יוחזר אורך הקטע (לא הלוך וחזור).
אם מספר הקדקודים הוא 1 או 0 יוחזר 0.
7. שיטה calcArea המחזירה מספר ממשי (double) המייצג את שטח המצולע.
כדי לחשב את שטח המצולע, צריך לסכום את שטחי המשולשים המכסים את שטח המצולע. בדוגמא לעיל, למשל, צריך לסכם את שטחי המשולשים הבאים: A-B-C, A-C-D, A-D-E.
לשם חישוב שטח המשולש ,ניתן להשתמש בנוסחת Heron הקובעת כי שטח המשולש שווה לשורש הריבועי של s(s-a)(s-b)(s-c) כאשר a, b ו c -הם אורכי שלוש הצלעות של המשולש ,ו s -הוא מחצית היקפו.
אם מספר הקדקודים קטן מ-3 יוחזר 0.
שימו לב שהשיטה לחישוב שטח משולש צריכה להיות פרטית ולא ציבורית.
8. שיטה בוליאנית isBigger המקבלת מצולע אחר, ומחזירה true אם המצולע שעליו מופעלת השיטה גדול בשטחו מהמצולע המועבר כפרמטר ואחרת מחזירה false. ניתן להניח שהפרמטר אינו null.
9. שיטה findVertex המקבלת נקודה כפרמטר ומחזירה את המיקום שלה ברשימה, אם היא נמצאת. אם לא, יוחזר 1-. לדוגמא, אם השיטה תופעל עם הפרמטר (5,0) היא תחזיר את הערך 2. (כי זהו האיבר השני ברשימה.)
10. שיטה getNextVertex, המקבלת נקודה כפרמטר, ומחזירה העתק של **הנקודה** המייצגת את הקדקוד הבא במצולע. אם הנקודה שהתקבלה אינה קדקוד במצולע, השיטה תחזיר null. אם הנקודה היא האיבר האחרון ברשימה, יוחזר העתק של

הנקודה הראשונה. אם הנקודה היא הנקודה היחידה ברשימה יוחזר העתק של הנקודה עצמה.

11. שיטה getBoundingBox המחזירה את **המלבן** (כפוליגון) **(המקביל לצירים)** החוסם את המצולע. אם מספר הקדקודים קטן מ-3 יוחזר null.

שימו לב שאנו לא נסביר כאן כיצד עושים זאת ולא נאפשר לתת רמזים או פרטים בפורום.

לפניכם רשימת החתימות של הבנאי ושיטות המחלקה:

בנאי	public Polygon()
שיטה שמוסיפה קדקוד למצולע	public boolean addVertex(Point p, int pos)
שיטה שמחזירה העתק של את הקדקוד שנמצא הכי גבוה במצולע	public Point highestVertex()
שיטה שמחזירה מחרוזת תווים המייצגת את המצולע	public String toString()
שיטה שמחזירה את היקף המצולע	public double calcPerimeter ()
שיטה שמחזירה את שטח המצולע	public double calcArea()
שיטה שמקבלת מצולע אחר, ובודקת אם המצולע שעליו מופעלת השיטה גדול בשטחו מהמצולע המועבר כפרמטר	public boolean isBigger(Polygon other)
שיטה שמקבלת נקודה כפרמטר ומחזירה את המיקום שלה ברשימה	public int findVertex(Point p)
שיטה שמקבלת נקודה כפרמטר, ומחזירה העתק של הנקודה המייצגת את הקדקוד הבא במצולע	public Point getNextVertex(Point p)
שיטה שמחזירה את המלבן המקביל לצירים (כפוליגון) החוסם את המצולע	public Polygon getBoundingBox()

שימו לב לא לבצע aliasing במקומות המועדים (מלבד המקומות בהם במפורש נדרש אחרת).

מותר להוסיף שיטות נוספות (פרטיות), לפי ראות עיניכם.

אתם צריכים כמובן לכתוב API לשלוש המחלקות.

שימו לב לכל מקרי השגיאה האפשריים!

כתבו כהערה ב- API מה סיבוכיות הזמן וסיבוכיות המקום של כל שיטה שכתבתם.
הקפידו על יעילות השיטות שכתבתם!

שימו לב:

1. **אסור להשתמש במחלקות מוכנות כבר של Java.**
2. **מותר ורצוי להשתמש במחלקות שניתנו בהרצאה ונמצאות בחוברת השקפים, לדוגמא Math.sqrt.**

הגשה

1. הגשת הממ"ן נעשית בצורה אלקטרונית בלבד, דרך מערכת שליחת המטלות.
2. **הקפידו ששמות השיטות והמחלקות יהיו בדיוק לפי הוראות הממ"ן.**
3. את התשובות לשאלות יש להגיש בשלושה קובצי Java הבאים: Point.java, PointNode.java, Polygon.java ארוזים יחד בתוך קובץ zip יחיד. אין לשלוח קבצים נוספים.

ב ה צ ל ח ה