מטלת מנחה (ממ"ן) 12

הקורס: 20441 - מבוא למדעי המחשב ושפת Java

חומר הלימוד למטלה : יחידות 3 - 4 **נושאי המטלה:** שימוש במחלקות נתונות וכתיבת מחלקות

מספר השאלות: 2 **משקל המטלה:** 4 נקודות

סמסטר: **2021ב** **מועד אחרון להגשה:** 10.4.2021

(ת)

מטרת מטלה זו היא להקנות לכם את עיקרי התכנות מונחה-העצמים.

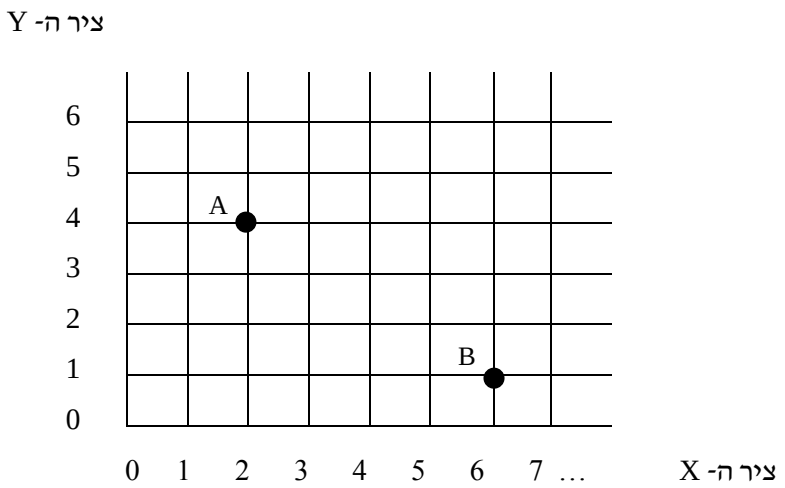
שאלה 1 - 20 נקודות

המחלקה Point מייצגת נקודה במישור, לפי מערכת הצירים הקרטזית (Cartesian system).

המחלקה Point מכילה 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.

לדוגמה, שתי הנקודות A = (2.0,4.0) ו- B = (6.0,1.0) מסומנות במרחב :



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)

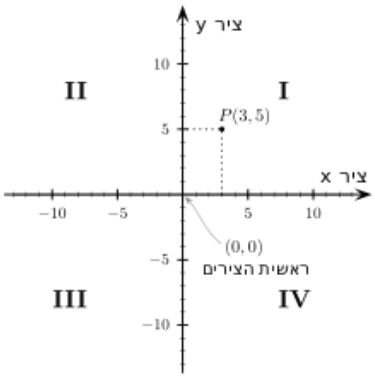
בנוסף, הוגדרו במחלקה השיטות:

- 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 נוספת בשם other כפרמטר ומחזירה האם הנקודה שעליה הופעלה השיטה והנקודה other שהתקבלה כפרמטר זהות. כלומר, מחזירה true אם ערכי הנקודה עליה השיטה מופעלת שווים לערכי הנקודה other.
- 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 ואין להשתמש בפעולות נוספות. אין לגשת לתכונות של הנקודות**.
- 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 ואין להשתמש בפעולות נוספות. אין לגשת לתכונות של הנקודות**.
- 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) , (x2,y2) היא

$$\sqrt[2]{(y2-y1)^2+(x2-x1)^2}$$

- int quadrant () – השיטה מחזירה את מספר הרביע (במישור) בו נמצאת הנקודה. מספרי הרביעים שהשיטה צריכה להחזיר הם לפי האיור הבא של מערכת הצירים (נלקח מויקיפדיה בערך "רביע"). אם הנקודה נמצאת על אחד מהצירים X או Y, השיטה תחזיר 0.



עליכם לכתוב את המחלקה Point לפי ההגדרות לעיל.

שאלה 2 - 80 נקודות

המחלקה **Triangle** מייצגת משולש במישור. משולש מאופיין על ידי שלושה קדקודים. **כל אחד מהם הוא מטיפוס Point.**

למחלקה **Triangle** יש את התכונות הפרטיות (instance variables) הבאות:

Point _point1 – מייצגת את הקודקוד הראשון של המשולש.

Point _point2 - מייצגת את הקודקוד השני של המשולש.

Point _point3 - מייצגת את הקודקוד השלישי של המשולש.

כמו כן, הוגדר במחלקה Triangle קבוע מסוג מספר ממשי בשם אפסילון כך:

```
static final double EPSILON = 0.001
```

יש להשתמש בקבוע זה בכל השוואה המתבצעת בין מספרים ממשיים לאחר ביצוע חישוב מתמטי. הסבר מפורט מופיע בסוף המטלה.

למחלקה Triangle הוגדרו ארבעה **בנאים** (constructors):

- בנאי ברירת מחדל המאתחל את קודקודי המשולש כך שערך קודקוד point1_ יהיה (1,0), ערך קודקוד point2_ יהיה (-1,0) וערך קודקוד point3_ יהיה (0,1).
- בנאי המקבל שלוש נקודות מטיפוס Point המהוות את שלושת קדקודי המשולש. אם לא ניתן ליצור משולש תקין מהנתונים שהתקבלו, יש לקבוע את ערכי הקודקודים לערכי ברירת מחדל. הגדרת תקינות משולש מפורטת בתאור פעולת isValid.
- בנאי המקבל שישה מספרים ממשיים המהווים את שלושת קודקודי המשולש. כל זוג מספרים מייצג את קואורדינטות ה- x וה- y של קדקוד אחד.

אם לא ניתן ליצור משולש תקין מהנתונים שהתקבלו, יש לקבוע את ערכי הקודקודים לערכי ברירת מחדל.

- בנאי העתקה המקבל משולש תקין אחר, ומעתיק את ערכיו.

public Triangle (Triangle other)

בנוסף הוגדרו במחלקה Triangle השיטות הבאות:

- שיטות **האחזור**:
  - getPoint1() - המחזירה את הקודקוד הראשון.
  - getPoint2() - המחזירה את הקודקוד השני.
  - getPoint3() - המחזירה את הקודקוד השלישי.
- השיטות **הקובעות**:
  - setPoint1(Point p) – השיטה מקבלת כפרמטר נקודה ומשנה את ערכי הנקודה point1_ לערכי נקודה זו.
  - setPoint2(Point p) – השיטה מקבלת כפרמטר נקודה ומשנה את ערכי הנקודה point2_ לערכי נקודה זו.
  - setPoint3(Point p) – השיטה מקבלת כפרמטר נקודה ומשנה את ערכי הנקודה point3_ לערכי נקודה זו.

עבור שלושת השיטות הקובעות, יש לבצע שינוי ערך קודקוד רק אם המשולש תקין לאחר השינוי.

- השיטה boolean isValid(Point p1, Point p2, Point p3) המחזירה true אם שלוש הנקודות שניתנו אכן יכולות להוות שלושה קדקודים של משולש.

בדיקת תקינות משולש - עליכם להשתמש באלגוריתם הבא: אם למשולש קיימת צלע שאורכה שווה לסכום שתי הצלעות האחרות – אזי המשולש אינו חוקי. אחרת, המשולש חוקי. לצורך ביצוע בדיקת השוויון יש להשתמש בקבוע אפסילון.

- השיטה String toString() המחזירה מחרוזת המייצגת את המשולש על ידי הדפסת קודקודיו. המחרוזת תהיה מהצורה: {p1,p2,p3}, כאשר כל נקודה תיוצג על ידי מחרוזת לפי המימוש במחלקה Point. **שימו לב לדייק במחרוזת לפי הכתוב כאן**. ללא רווחים וללא תווים נוספים.

לדוגמה: עבור שלושה קודקודים של משולש (2.5, -4.0) , (-12.3,0.5) ו- (3.0,0.0) תוחזר המחרוזת הבאה:

{(2.5, -4.0) , (-12.3,0.5) , (3.0,0.0)}

- השיטה double getPerimeter() המחזירה את היקף המשולש.
- השיטה double getArea() המחזירה את שטח המשולש.

חישוב שטח משולש: http://he.wikipedia.org/wiki/נוסחת_הרון

- השיטה boolean isIsosceles() המחזירה true אם המשולש הוא שווה שוקיים. אחרת, false.
- השיטה boolean isPythagorean() המחזירה true אם המשולש הוא ישר זווית. אחרת, false.
- השיטה boolean isContainedInCircle(double x , double y, double r) המחזירה true אם המשולש מוכל במעגל שמרכזו ב-(x,y) ורדיוסו r, ו- false אחרת. (אם קדקוד המשולש נוגע בשפת המעגל הקדקוד ייחשב כבתוך המעגל). אין צורך בשיטה זאת בנוסחה מתמטית מסוימת, רק בהיגיון בריא.
- השיטה Point lowestPoint() - פעולה המחזירה את הנקודה הנמוכה ביותר במשולש. אם יש שתי נקודות שהן "הנמוכות ביותר" צריך להחזיר את השמאלית ביניהן.
- השיטה Point highestPoint() - פעולה המחזירה את הנקודה הגבוהה ביותר במשולש. אם יש שתי נקודות שהן "הנמוכות ביותר" צריך להחזיר את השמאלית ביניהן.
- השיטה boolean isLocated() – המחזירה true אם המשולש כולו נמצא ברביע אחד, אחרת false. **תזכורת**: נקודה הממוקמת על אחד מהצירים אינה משויכת לרביע.
- השיטה boolean isAbove(Triangle other) - שיטה שמקבלת משולש כפרמטר ומחזירה האם המשולש שעליו הופעלה השיטה נמצא כולו **מעל** למשולש שהתקבל כפרמטר. אם כן הפעולה מחזירה true, ו-false אחרת.
- השיטה boolean isUnder (Triangle other) - שיטה שמקבלת משולש כפרמטר ומחזירה האם המשולש שעליו הופעלה השיטה נמצא כולו **מתחת** למשולש שהתקבל כפרמטר. **השיטה הזו משתמשת אך ורק בשיטה isAbove שהוגדרה לעיל. אין לגשת לתכונות של המשולשים.**
- השיטה boolean equals(Triangle other) - שיטה שמקבלת משולש כפרמטר ומחזירה האם המשולש שעליו הופעלה השיטה **זהה** למשולש שהתקבל כפרמטר.

שימו לב ששני המשולשים {(1,0),(0,0),(0,1)} ו- {(0,0),(0,1),(1,0)} לא זהים!

- boolean isCongruent(Triangle other) - שיטה שמקבלת משולש כפרמטר ומחזירה האם המשולש שעליו הופעלה השיטה הם משולשים חופפים (שווים). ניתן לבצע בדיקה זו על-ידי השוואת אורכי הצלעות בין המשולשים.

הסבר נוסף: https://he.wikipedia.org/wiki/חפיפת משולשים

תזכורת מתמטית:

בכדי לחשב מרחק בין שתי נקודות - (x1, y1), (x2, y2) - השתמשו בנוסחה הבאה:

$$d = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}$$

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).

הנחיות חשובות:

1. שימו לב, כיוון שמדובר במספרים ממשיים (לאו דווקא שלמים), השוויון הוא עד כדי אפסילון (EPSILON). כלומר, במחלקה מוגדר ערך קבוע קטן מאד ושמו אפסילון. אם ההפרש בין שני מספרים קטן מאפסילון זה, הם נקראים שווים.

   לדוגמה: אם a ו-b הם מסוג double אז כדי לבדוק אם הם שווים נכתוב:

   Math.abs(a-b) < EPSILON

   מומלץ לכתוב שיטה פרטית לביצוע השוואה זו.

2. **כל השיטות במטלה שמשתמשות בשיטת המרחק של הנקודה צריכות להשתמש ב-EPSILON.**
3. בכל השיטות במטלה שמקבלות אובייקט כפרמטר אפשר להניח שמתקבל אובייקט שאותחל ואינו שווה ל- **null**.
4. עליכם להימנע מביצוע **aliasing** בשיטות ובבנאים.
5. אם במהלך כתיבת השיטות המבוקשות אתם רוצים להשתמש בשיטות עזר נוספות, הן חייבות להיות **private**.
6. כדי להימנע משכפול קוד במחלקה Triangle, יש להשתמש לפי הצורך בשיטות הקיימות במחלקות Point ו- Triangle.

7. עליכם לתעד היטב את המחלקות Triangle ו- Point שכתבתם בשיטת ה- API, כפי שהודגמה בהרצאות. כדי ליצור את קובץ ה- html שמכיל את ה- API, עליכם לעבור למצב של documentation או interface (תלוי איזו גרסה של BlueJ מותקנת אצלכם) בכפתור העליון בצד ימין (ללחוץ על החץ), במסך של המחלקה. כשתעברו למצב התיעוד ייווצר בו בזמן קובץ ובו תיעוד ה-API של המחלקה, בשם Triangle.html ו- Point.html. הקובץ הזה נמצא בתוך התת-תיקיה doc שנמצאת בתוך התיקיה המכילה את הפרויקט שלכם. **עליכם לתעד את כל המחלקות שתכתבו ב- API וגם בתיעוד פנימי**. אפשר כמובן להשתמש בהערות ה-API שנמצאות באתר.

8. **שימו לב ששמנו טסטרים לשתי המחלקות באתר הקורס. חובה לוודא שטסטרים אילו ירוצו ללא שגיאות קומפילציה עם המחלקות שלכם. אם יש שיטה שלא כתבתם, כתבו חתימה לשיטה ובתוך גוף השיטה החזירו ערך סתמי כדי שהטסטרים ירוצו עם המחלקות ללא שגיאות קומפילציה. מי שיגיש מטלה שלא עוברת קומפילציה הציון במטלה שלו יהיה אפס!**

הגדרות מדויקות לבנאים ולשיטות הנדרשות לפי API תמצאו באתר הקורס.

הגשה

1. הגשת הממ"ן נעשית בצורה אלקטרונית בלבד, דרך מערכת שליחת המטלות.
2. זכרו כי הקפדה על שמות מחלקות ושיטות (ציבוריות), לפי הנדרש, היא הכרחית. כל חריגה מההגדרות (אפילו החלפה בודדת של אות גדולה בקטנה, למשל) תגרום לבדיקה האוטומטית שלנו להיכשל וכתוצאה מכך לנזק בלתי הפיך בציון.
לכן, הקפידו ששמות המחלקות והשיטות יהיו בדיוק כפי שמוגדר בממ"ן. **אחרת יורדו לכם הרבה נקודות!**
3. עליכם להריץ את הטסטרים שנמצאים באתר הקורס על המחלקות שכתבתם. שימו לב שהטסטרים לא מכסים את כל האפשרויות, ובפרט לא את מקרי הקצה. הם רק בודקים את השמות של השיטות במחלקות. מאד מומלץ להוסיף להם בדיקות.
4. עליכם להגיש את הקבצים Point.java ו- Triangle.java.
5. עטפו את שני הקבצים בקובץ zip יחיד ושלחו. אין לשלוח קבצים נוספים.

תוספת לשירותכם -

כדי שתוכלו לראות את ציורי המשולשים בצורה גרפית אנו נותנים לכם את המחלקה VisualTriangle. מחלקה זו יודעת לצייר מערכת צירים ואת המשולשים עליה. תוכלו למצוא את המחלקה בקבצים להורדה באתר הקורס. המחלקה מכילה שני בנאים ושתי השיטות הבאות:

שני הבנאים:

~~public VisualTriangle() – בנאי ריק שיוצר אובייקט מסוג VisualTriangle.~~

- public VisualTriangle(String textTitle) – בנאי שיוצר אובייקט מסוג VisualTriangle

  textTitle- טקסט שיוצג על כותרת החלון

- public VisualTriangle(String textTitle, boolean showVal) – בנאי שיוצר אובייקט מסוג VisualTriangle

  אם הפרמטר showVal=true יוצגו ערכי הקודקודים של המשולש

שתי השיטות:

- public add(Triangle t) – השיטה מקבלת אובייקט מסוג משולש ומוסיפה אותו לציור (ניתן להוסיף אינסוף משולשים לאותה מערכת צירים).
- public void clear() – השיטה מוחקת ממערכת הצירים את כל המשולשים שעליה.

שימו לב, אינכם חייבים להשתמש במחלקה זו. אך אם תעשו זאת, תוכלו לראות את המשולשים שהגדרתם, לפני ואחרי ביצוע שיטות שונות מהמחלקה Triangle. כך תוכלו לבדוק את השיטות שלכם ביתר קלות.

אם בחרתם להשתמש במחלקה זו, מאוד מומלץ לכתוב מחלקת בדיקה נפרדת (Tester) שכוללת main בלבד, יוצרת מופעים של Triangle, מפעילה עליהם שיטות ומציגה אותם באופן גרפי. כמובן שמחלקה זאת לא תהיה חלק מהתרגיל המוגש וכל אזכור שלה בתוך המחלקה Triangle הוא שגוי.

ב ה צ ל ח ה