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: 3 **משקל המטלה:** 4 נקודות

סמסטר: **2022א** **מועד אחרון להגשה:** 27.11.2021

(ת)

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

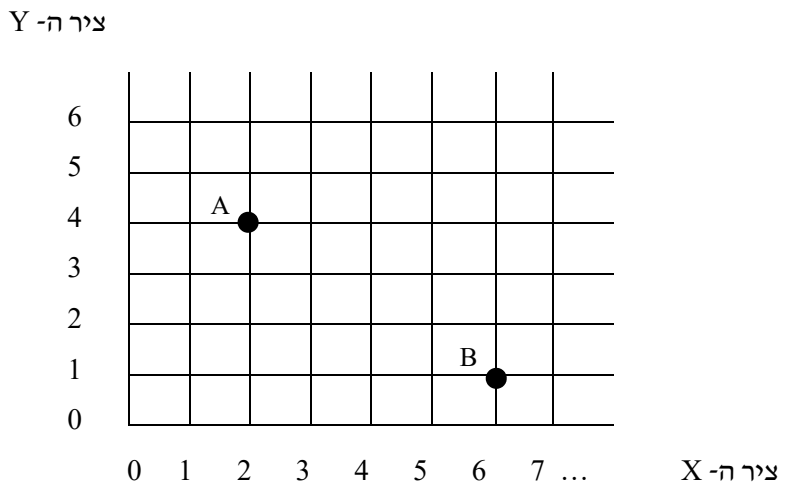
שאלה 1 - 20 נקודות

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

המחלקה Point מכילה את התכונות הפרטיות (instance variables) הבאות:

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

לדוגמה, שתי הנקודות A = (2,4) ו- B = (6,1) מסומנות במרחב:



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

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

public Point(int x, int y)

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

public Point (Point other)

בנוסף, הוגדרו במחלקה השיטות:

- שיטות **האחזור**:
  - int getX() המחזירה את ערכה של קואורדינטת ה- x.
  - int getY() המחזירה את ערכה של קואורדינטת ה- y.
- השיטות **הקובעות**:
  - void setX (int num) המשנה את ערכה של קואורדינטת ה- x להיות num.
  - void setY (int 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 (3,4) מייצגת את הנקודה שקואורדינטת ה- x שלה היא 3 וקואורדינטת ה- y שלה היא 4. **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**.
- void move (int deltaX, int deltaY) – המזיזה את הנקודה ב- deltaX על ציר ה- X וב- deltaY על ציר ה- Y.
- double distance (Point p) –שיטה שמקבלת נקודה כפרמטר ומחזירה את המרחק בין הנקודה שעליה הופעלה השיטה והנקודה שהתקבלה כפרמטר.

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).

עליכם לכתוב את המחלקה Point לפי ההגדרות לעיל.

שאלה 2 - 40 נקודות

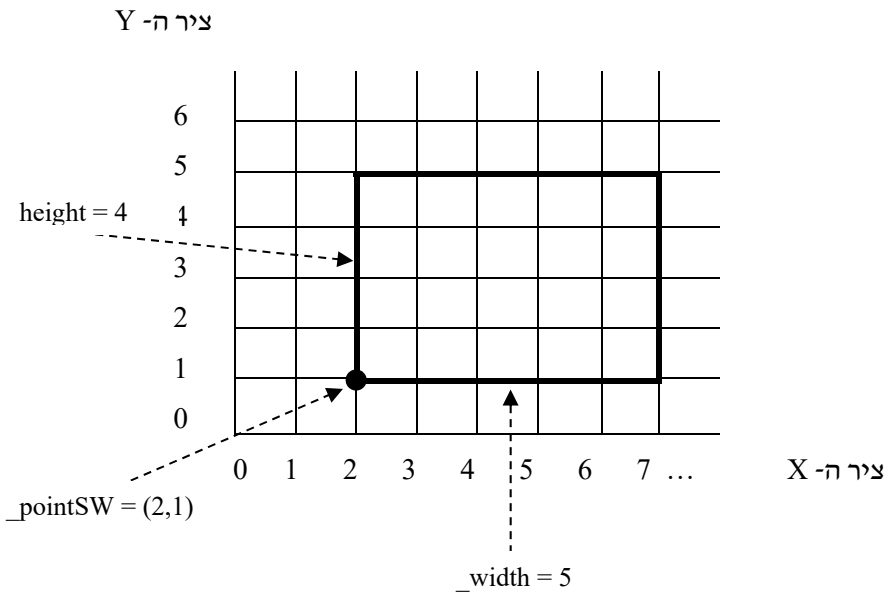
המחלקה RectangleA מייצגת מלבן שצלעותיו מקבילות לצירים.

למחלקה RectangleA התכונות הפרטיות (instance variables) הבאות:

- int _width – רוחב המלבן
- int _height – גובה המלבן
- Point _pointSW – מיקום הנקודה הדרום-מערבית של המלבן.

הנה דוגמא למלבן:

הנקודה הדרום-מערבית שלו היא במיקום (2, 1), גובהו 4 ורוחבו 5.



למחלקה RectangleA הוגדרו ארבעה בנאים:

- האחד - המקבל שני מספרים שלמים כפרמטרים, הראשון הוא הרוחב של המלבן והשני הוא הגובה. הנקודה הדרום-מערבית תהיה בראשית הצירים (0, 0). האורך והרוחב חייבים להיות שלמים חיוביים ממש. אם מישהו מהם אינו כזה, התכונה תאותחל להיות 1.

public RectangleA(int w, int h)

- השני - המקבל פרמטרים עם ערכים לתכונות המתאימות: הנקודה הדרום-מערבית תתקבל כפרמטר (כנקודה, שאינה null), והרוחב והגובה יתקבלו כפרמטרים. האורך והרוחב חייבים להיות שלמים חיוביים ממש. אם מישהו מהם אינו כזה, התכונה תאותחל להיות 1.

public RectangleA(Point p, int w, int h)

- השלישי - המקבל שתי נקודות כפרמטרים. הנקודה הראשונה היא הנקודה הדרום-מערבית (sw) והנקודה השניה היא הנקודה הצפון-מזרחית (ne). אתם יכולים להניח שאכן הנקודה sw נמצאת **ממש** דרומית-מערבית לנקודה ne. בפרט, לשתי הנקודות לא יכולות להיות אותו ערך x או אותו ערך y. אין צורך לבדוק זאת. כמו כן, אפשר להניח ששתי הנקודות המועברות כפרמטרים אינן null.

public RectangleA(Point sw, Point ne)

- הרביעי - בנאי העתקה, המקבל אובייקט מהמחלקה RectangleA ומעתיק את ערכיו.

public RectangleA(RectangleA r)

כמו כן, הוגדרו שיטות האחזור (get) והשיטות הקובעות (set) לפי השמות המקובלים לכל התכונות של המחלקה, והשיטה toString שמחזירה את נתוני המלבן כמחרוזת תווים.

- public int getWidth()
- public int getHeight()
- public Point getPointSW()
- public void setWidth(int w)
- public void setHeight(int h)
- public void setPointSW(Point p)
- public String toString()

הערות:

- בשיטות setWidth ו- setHeight, אם הפרמטר אינו חיובי ממש, לא ייעשה כלום, והתכונה תישאר כשהיתה.
- בשיטה toString המחרוזת המייצגת את המלבן שבאיור לעיל, תיראה כך בדיוק (ללא רווחים נוספים וללא סימנים נוספים) –

Width=5 Height=4 PointSW=(2,1)

למחלקה RectangleA נוסיף גם את השיטות הבאות:

- שיטה המחזירה את היקף המלבן

```
public int getPerimeter()
```

- שיטה המחזירה את שטח המלבן

```
public int getArea()
```

- שיטה המזיזה את המלבן למיקום אחר, השיטה מקבלת שני שלמים deltaX ו- deltaY המורים בכמה להזיז את המלבן בציר ה-X ובציר ה- Y בהתאמה.

```
public void move(int deltaX, int deltaY)
```

- שיטה בוליאנית שמקבלת מלבן כפרמטר ומחזירה האם המלבן שעליו הופעלה השיטה זהה למלבן שהתקבל כפרמטר (זהה מבחינת כל התכונות).

```
public boolean equals(RectangleA other)
```

- שיטה המחזירה את אורך האלכסון במלבן.

```
public double getDiagonalLength()
```

- שיטה בוליאנית שמקבלת מלבן כפרמטר ומחזירה האם שטחו של המלבן שעליו מופעלת השיטה גדול משטחו של המלבן שהתקבל כפרמטר.

```
public boolean isLarger(RectangleA other)
```

- שיטה שמחזירה את הנקודה הצפון-מזרחית של המלבן.

```
public Point getPointNE()
```

- שיטה המשנה את הצדדים של המלבן, כך שמה שהיה רוחב הופך להיות גובה, והגובה הופך להיות רוחב. הנקודה הדרום-מערבית אינה משתנה.

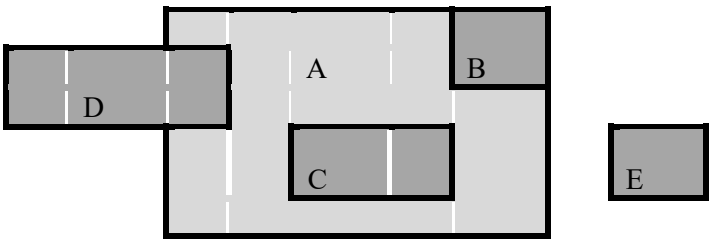
public void changeSides()

- שיטה בוליאנית המקבלת מלבן נוסף r ומחזירה true אם המלבן עליו מופעלת השיטה נמצא כולו בתוך המלבן r . (גם צלעות משותפות נחשבות "בתוך").

public boolean isIn (RectangleA r)

לדוגמא,

המלבן A המצויר להלן, מכיל את המלבנים B ו- C, אבל לא מכיל את המלבנים D ו-E.



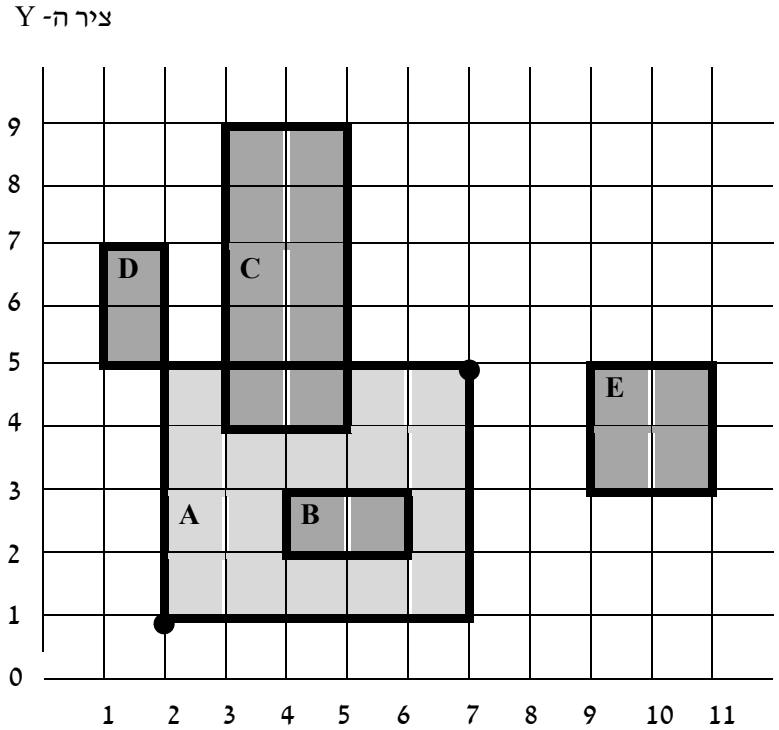
- שיטה בוליאנית המקבלת מלבן נוסף r ומחזירה true אם יש חפיפה בין המלבנים ו- false אחרת.

תזכורת –

נאמר שיש חפיפה בין שני מלבנים אם יש נקודות במישור שנמצאות בשני המלבנים.

public boolean overlap (RectangleA r)

לדוגמא, נסתכל על האיור שלהלן:



המלבן A המצויר לעיל, הנקודה הדרום-מערבית שלו היא (2,1) והנקודה הצפון-מזרחית שלו היא (7,5). הנקודות מסומנות בגרף בעיגולים בולטים.

לכל המלבנים הבאים יש חפיפה עם המלבן A:

- B שהנקודה הדרום-מערבית שלו היא (4,2) והנקודה הצפון-מזרחית שלו היא (6,3)
- C שהנקודה הדרום-מערבית שלו היא (3,4) והנקודה הצפון-מזרחית שלו היא (5,9)
- D שהנקודה הדרום-מערבית שלו היא (1,5) והנקודה הצפון-מזרחית שלו היא (2,7)

שימו לב שהמלבן A ו- D חופפים רק בנקודה אחת שהיא (2,5)

ואילו המלבן

- E שהנקודה הדרום-מערבית שלו היא (9,3) והנקודה הצפון-מזרחית שלו היא (11,5)

אינו חופף באף נקודה למלבן A.

עליכם לכתוב את המחלקה RectangleA לפי ההגדרות לעיל (ולפי הכתוב ב- API שבאתר).

שימו לב לא לבצע aliasing במקומות המועדים.

שאלה 3 - 40 נקודות

המחלקה RectangleB מייצגת מלבן שצלעותיו מקבילות לצירים.

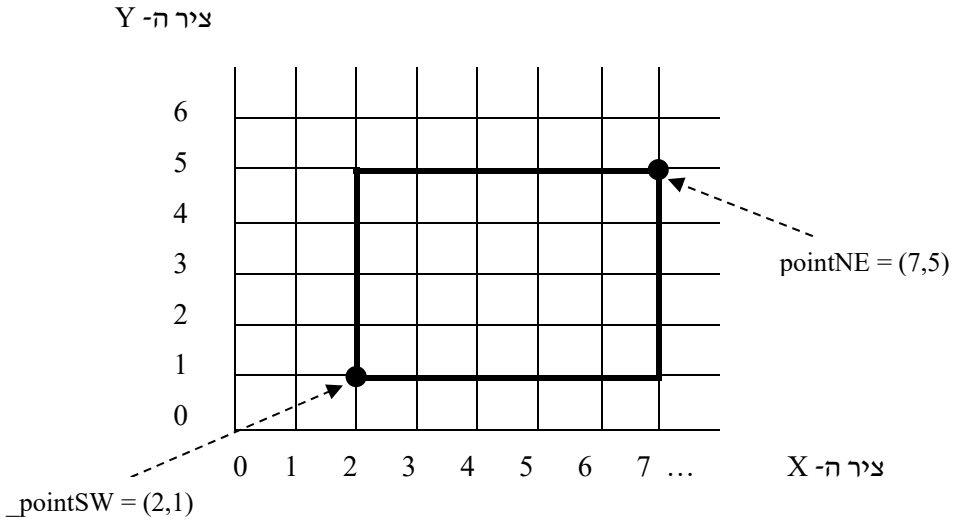
למחלקה RectangleB התכונות הפרטיות (instance variables) הבאות:

- Point _pointSW – מיקום הנקודה הדרום-מערבית של המלבן.
- Point _pointNE – מיקום הנקודה הצפון-מזרחית של המלבן.

הנה דוגמא למלבן שהוצג בשאלה 2:

הנקודה הדרום-מערבית שלו היא במיקום (2, 1),

הנקודה הצפון-מזרחית שלו היא במיקום (7, 5)



למחלקה RectangleB הוגדרו ארבעה בנאים:

- האחד - המקבל שני מספרים שלמים כפרמטרים, הראשון הוא הרוחב של המלבן והשני הוא הגובה. הנקודה הדרום-מערבית תהיה בראשית הצירים (0, 0). האורך והרוחב חייבים להיות שלמים חיוביים ממש. אם מישהו מהם אינו כזה, התכונה תאותחל להיות 1.

```
public RectangleB(int w, int h)
```

- השני - המקבל פרמטרים עם ערכים לתכונות המתאימות: הנקודה הדרום-מערבית תתקבל כפרמטר (כנקודה, שאינה null), והרוחב והגובה יתקבלו כפרמטרים. האורך והרוחב חייבים להיות שלמים חיוביים ממש. אם מישהו מהם אינו כזה, התכונה תאותחל להיות 1.

```
public RectangleB(Point p, int w, int h)
```

- השלישי - המקבל שתי נקודות כפרמטרים. הנקודה הראשונה היא הנקודה הדרום-מערבית (sw) והנקודה השניה היא הנקודה הצפון-מזרחית (ne). אתם יכולים להניח שאכן הנקודה sw נמצאת **ממש** דרומית-מערבית לנקודה ne. אין צורך לבדוק זאת. כמו כן, אפשר להניח ששתי הנקודות המועברות כפרמטרים אינן null.

```
public RectangleB(Point sw, Point ne)
```

- הרביעי - בנאי העתקה, המקבל אובייקט מהמחלקה RectangleB ומעתיק את ערכיו.

```
public RectangleB(RectangleB r)
```

השיטות של המחלקה RectangleB זהות לחלוטין לאלו של RectangleA.
לבד מהמקרים בהם מתקבל מלבן כפרמטר לשיטה, ואז במקום שכתוב RectangleA צריך להיות RectangleB.
במחלקה RectangleB אין להשתמש במחלקה RectangleA או בשיטות שנכתבו בה, ולהיפך! אלה שתי מחלקות נפרדות ובלתי תלויות לחלוטין!

שימו לב, גם השיטה toString של המחלקה RectangleB צריכה להיות לפי זו של המחלקה RectangleA, כלומר להדפיס את הנקודה הדרום-מערבית, את הרוחב והגובה.

הגדרות מדויקות לפי API תמצאו באתר הקורס בתת-ספריה "קובצי API".

הנחיות חשובות:

1. בכל השיטות במטלה שמקבלות אובייקט כפרמטר אפשר להניח שמתקבל אובייקט שאותחל ואינו שווה ל- **null**.
2. עליכם להימנע מביצוע **aliasing** בשיטות ובבנאים.
3. אם במהלך כתיבת השיטות המבוקשות אתם רוצים להשתמש בשיטות עזר נוספות, הן חייבות להיות **private**.
4. כדי להימנע משכפול קוד במחלקות RectangleA ו- RectangleB, יש להשתמש לפי הצורך בשיטות הקיימות במחלקות Point.
5. עליכם לתעד היטב את המחלקות Point , RectangleA ו- RectangleB 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 RectangleA.html, RectangleB.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. **שימו לב שאתם לא צריכים לשלוח את קובצי ה- html, אבל בהחלט עליכם להכין אותם.**

6. **שימו לב ששמנו טסטרים לשלוש 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!**
7. **הבהרה: התנהגות המחלקות בשאלות 2 ו-3 בכל השיטות צריכה להיות זהה. לכן, בשיטה setPointSW במחלקה RectangleB יש לדאוג שערכי הרוחב והגובה לא ישתנו, כלומר לעדכן גם את הנקודה הצפון-מזרחית בהתאם.**

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, RectangleA.java ו- RectangleB.java **בלבד. הגשה של קבצים נוספים תגרע מהציון. באחריותכם להגיש את הקבצים הנכונים.**
5. עטפו את שלושת הקבצים בקובץ zip יחיד ושלחו. **אין לשלוח קבצים נוספים**.

ב ה צ ל ח ה