# תכנות מונחה עצמים (שפת JAVA):

## תרגיל בית להגשה מס 1. (ארנון ברקת)

## סמסטר א , 2020.

## Exception Handling,

## Abstract classes and interfaces.

**תאריך חלוקת התרגיל בכיתה: יום ראשון, 02/12/2019.**

**התאריך להגשת התרגיל: עד וכולל יום ראשון, 23/12/2019. (עד השעה 23:45.)**

יש להגיש אך ורק דרך תפריט המטלות שבאתר הקורס, כפי שהוסבר בתרגול.

שם הקובץ המוגש יכיל את שם המגיש ויהיה אך ורק לפי הפורמט הבא: למשל:

עבור תרגיל בית מס 1: *HW1_AviLevi.java*. עבור תרגיל בית מס 2: *HW2_AviLevi.java*, וכך הלאה.

שם המחלקה יהיה אף הוא בהתאם. למשל: *public class HW1_AviLevi{*.

בנוסף, הקובץ יכלול בשורה הראשונה למעלה, הערה ובה השם המלא של המגיש/ה.

יש להמיר את קובץ ה- *java* לקובץ *zip*, ולהגיש באתר אך ורק קובץ אחד והוא קובץ ה- *zip*.

בכל מקרה אין להגיש באתר יותר מקובץ אחד של *java* שנמצא בתוך קובץ אחד של *zip*.

על הסטודנט חלה האחריות שהקובץ שהוא הגיש תקין, נשלח בצורה נכונה ומתאים לנדרש.

על התרגילים להיפתח בהצלחה בתוכנת ה- *Eclipse*, ללא שגיאות הידור או אזהרות.

יש לשלב בקוד הערות ותיעוד מתאים. יש לבצע הזחה של הקוד כנדרש.

**יש להקפיד על כללי הנדסת התוכנה ככל הניתן**: קוד קצר, לא מסורבל ויעיל הן מבחינת כתיבתו והן מבחינת ריצת התוכנית. לא להשתמש במשתנים סטטיים שלא לצורך, תוכנית כללית שניתנת בקלות לשינויים והרחבות בעתיד, שימוש בקבועים, חלוקה מתאימה לפונקציות, פונקציות עצמאיות (כלומר שאינן תלויות בקוד /משתנה חיצוני), וכיוצ"ב. בנוסף, פתרון התרגיל צריך להיות גם כללי לכל שינוי של הנתונים, שינוי הגדלים של המערכים באם קיימים, וכו.

כאשר הבדיקה תסתיים, תהיה על כך הודעה באתר. **עד שאין את ההודעה, יש עוד עבודות שלא נבדקו.**

**לא יבדקו תרגילים שמוגשים באיחור ו/או שאינם עומדים בדרישות הנ"ל**

**<u>המטלה הנדרשת</u>**

- כתוב מחלקה *Circle* המייצגת עיגול.

  לעיגול יש *int radius*, וגם *String color* עבור הצבע שלו.

- כתוב מחלקה *Triangle* המייצגת משולש.

  למשולש יש *int base* - עבור הבסיס, *int height* - עבור הגובה. גם עבורו שומרים את הצבע כמו מקודם.

- כתוב מחלקה *Rect* המייצגת מלבן.

  למלבן יש *int width*- עבור הרוחב, *int height* - עבור הגובה. גם עבורו שומרים את הצבע כמו מקודם.

לכל אחת מצורות אלו נרצה גם להציג את השטח כמתואר בהמשך.

- ב- *main*, יש לצור מערך אחד של הצורות השונות, לפי דוגמת ההרצה באופן מדויק.

  **בכל התוכנית יש לעבוד עם מערך 'רגיל' (כלומר עם סוגריים מרובעות [ ]), ולא עם *ArrayList* או משהו דומה.**

שים לב: רק עבור הצורה הראשונה, שהיא עיגול, קולטים מהמשתמש את הצבע והרדיוס. (ברצף, כלומר קודם הצבע נקלט, אז המשתמש לוחץ *enter* ומייד מזין את הרדיוס.)
שאר הצורות נקבעות בתוכנית בהתאמה לפלט.

- כעת: קיימת רשימה של צבעים אסורים: ירוק, סגול, חום ושחור. אם הקלט מהמשתמש הוא אחד מצבעים אלו (וללא תלות באותיות גדולות או קטנות), אז יש ללכוד את החריגה, להודיע לו, ולאפשר לו לחזור על כל תהליך הקליטה מחדש. (צבע ורדיוס.) באותו אופן, אם הוא מזין רדיוס שאינו מספר שלם, גם כאן יש לבצע לכידה, להודיע לו בהתאם, ולאפשר קליטה נוספת. עבור כל סוג לכידה יש ללכוד בנפרד עם הודעה שונה.

הערה: נתון וידוע, שהצורות האחרות במערך תקינות מכל הבחינות ואת זה אין צורך לוודא או לבדוק.

- כעת: כאשר הקלט תקין, יש לכתוב מתודה *show* אחת, שתדפיס את כל המערך, וגם תציג את השטח של כל אחת מהצורות. (כמו בדוגמת ההרצה.)
  לאחר מכן, יש לבצע מיון של כל הצורות במערך לפי השטח, ולהדפיס שוב אותן לאחר המיון.

- בשלב הזה, יש לכתוב ממשק *(interface)* שיאפשר את היכולת לבדוק תקינות של רדיוס העיגול. יש לממש את הממשק בתוך המחלקה של העיגול. בממשק תהיה מתודה אחת בשם *radiusCheck*, שמטרתה **לזרוק חריגה** ברגע שהרדיוס גדול מ- 500 או שהוא מספר שלילי.
  (ניתן להפעיל את הבדיקה מתי שרוצים, כלומר לאו דווקא בעת יצירת העיגול, כלומר בבנאי שלו.)
  החריגה שתיזרק הוא מסוג מחלקה בשם ***RadiusException* שאותה עליך לכתוב**, והיא תהיה חריגה מהסוג של ***<u>checked</u>***. בעת לכידת החריגה הזו, יש להדפיס את הפלט לפי הדוגמא שבהמשך, רק בעזרת המתודה *getMessage*

(שמתקבלת בירושה מהמחלקה *Throwable* ). כלומר בעזרתה לשחזר את ההודעה שנשמרה באובייקט של *RadiusException* בעת שיצרנו אותו.

יש להפעיל את בדיקת הרדיוס לאחר שכבר נוצר העיגול ע"י המשתמש, ואם הרדיוס לא תקין יש להציג כאמור את ההודעה (למשל: *800 is illegal radius* ראה דוגמת הפלט) ולאפשר למשתמש לחזור על כל התהליך כ מו קודם וכו.

**<u>אילוצים:</u>**

- יש לקבל פלט מדויק בהתאם לדוגמת ההרצה שבהמשך. (ובהתאם לקלט כל פעם.)
- המילים *Object* ו- *instanceof* לא יופיעו באף מקום בתוכנית.
- את המשתנה *color* אסור להכריז במחלקות, וגם לא תהיה אליו גישה כלשהי ממחלקות אלו. (לא ישירה ולא ע"י מתודה כגון *get*.)
- אין להעתיק את המערך למבנה עזר אחר כלשהו כגון *ArrayList* וכו.
- בדיקת תאימות הצבעים תבוצע **בעת יצירת הצורה**. (כלומר, **<u>ניסיון לצור צורה</u>** מ- צבע אסור תזרוק חריגה.)

**<u>הדוגמא להרצת התוכנית:</u>**

```
Please enter color and then radius for the first circle:
black
600

Illegal color!
Please enter color and then radius for the first circle:
black
100

Illegal color!
Please enter color and then radius for the first circle:
RED
800

800 is illegal radius
Please enter color and then radius for the first circle:
reD
-1

-1 is illegal radius
Please enter color and then radius for the first circle:
reD
1.5

Only integer is allowed here..
Please enter color and then radius for the first circle:
yellow
10

Original array and areas:
==========================
Circle, yellow, 10, area = 314.1592653589793
Triangle, red, 10, 20, area = 100.0
Rect, blue, 3, 5, area = 15.0
Circle, yellow, 100, area = 31415.926535897932
```

```
Triangle, red, 1, 2, area = 1.0
Rect, white, 6, 2, area = 12.0

after sorting:
==========================
Triangle, red, 1, 2, area = 1.0
Rect, white, 6, 2, area = 12.0
Rect, blue, 3, 5, area = 15.0
Triangle, red, 10, 20, area = 100.0
Circle, yellow, 10, area = 314.1592653589793
Circle, yellow, 100, area = 31415.926535897932
```

***בהצלחה!***