תרגיל 4 - AutoCAD צעצוע

הקדמה ומנהלות

מפאת קוצר בזמן, התרגיל הוא תרגיל רשות. עבור סטודנטים שיבחרו להגיש את התרגיל, הוא ישתקלל לממוצע בתור ציון מגן.

הגשה בתיקיית הבית בנובה, בתיקיה בשם ex4 תחת התיקיה c_lab. יש להעתיק אליה את קובץ ה-README הקבוע, ולתת הרשאות 755 לכל הקבצים בתוכה.
את התרגיל יש לכתוב, כמובן, בשפת Java בלבד.
מועד הגשה: 29/6

תזכורת: **התרגילים נבדקים אוטומטית בעזרת מערכת לזיהוי העתקות**, כולל מול הגשות מסמסטרים קודמים.

קובץ ההרצה של התרגיל צריך להיקרא ToyCAD.class. שימו לב לאותיות גדולות וקטנות.
הטסט האוטומטי של התרגיל נגיש במיקום:
~nimrodav/java_ex/run_test.sh
שימו לב שהטסט בודק רק חלק מדרישות התרגיל, אך כמובן שכל הדרישות מחייבות אתכם, גם דרישות שהטסט לא בודק.

מומלץ מאוד להוריד את סביבת הפיתוח Eclipse למחשב האישי שלכם, לפתח את כל התרגיל עליה, ורק לאחר מכן להעביר את הקוד לשרתי הלינוקס ולבדוק שהוא עובד שם.

Is-A

כפי שאמרנו פעמים רבות בכיתה, בתכנות מונחה עצמים קלאס A צריך לרשת מקלאס B, אם ורק אם המשפט
"A is a B"
הוא נכון בחיים האמיתיים. לפני הגשת התרגיל, יש לוודא שהקוד שהגשתם מקיים הנחיה זו.
לנוחותכם, ועל מנת להקל על שיתוף הפעולה בין שותפים, בתיקיית ההגשה חייב להיות עותק של קובץ ההרצה
~nimrodav/java_ex/is_a.sh
והטסט האוטומטי אוכף זאת. קובץ ההרצה מדפיס למסך את כל יחסי ה-is-a שאתם טוענים שמתקיימים בין אובייקטים בעולם האמיתי. כך תוכלו להיות בטוחים שלפחות אחד מהשותפים נתן את דעתו על ההנחיה הזו.

קונבנציית קוד

הטסט האוטומטי של התרגיל דורש שהקבצים יהיו מפורמטים לפי הקונבנציה של הכלי google-java-format. ניתן לפרמט את הקבצים בעזרת הפקודה

```
java -jar ~nimrodav/google-java-format/google-java-format-1.5-all-deps.jar --replace *.java
```

התרגיל עצמו

בתרגיל זה נבנה גירסת צעצוע של AutoCAD, תוכנה ידועה לעריכת שרטוטים.
התוכנה מקבלת קלט מ-stdin, שורה שורה, כאשר כל שורה מכילה פקודה יחידה. פקודות יכולות להורות על הוספת צורות לשרטוט, שינויים בצורות קיימות, מחיקת צורות, ציור השרטוט על המסך וכו'.
לדוגמה, הפקודה

```
new Circle color x y radius
```

מוסיפה לשרטוט צורה חדשה מסוג מעגל, שמרכזו בקואורדינטות X, Y צבעו נתון, והרדיוס שלו נתון.
התכנית היא case-insensitive, כלומר היא מקבלת פקודות ללא תלות באותיות קטנות או גדולות.
השרטוט מורכב בכל רגע נתון מאוסף צורות. לכל צורה, בנוסף לצבע ולמאפיינים של המקום הגאומטרי שלה, יש ID, מספר טבעי שמזהה אותה באופן חד ערכי. הצורה הראשונה מקבל את ה-ID שערכו 0, השניה את ה-ID שערכו 1, וכו'. ID לעולם לא יכול לחזור על עצמו, כלומר אם צורה קיבלה את ה-ID 7 ולאחר מכן נמחקה, אף צורה אחרת לא יכולה לקבל את ה-ID 7.
הצורות שצריך לתמוך בהן הן: מעגל, אליפסה, מקבילית, מלבן, ריבוע, ומשולש.
ניתן להניח שצורות אינן חופפות.
פקודות מתחלקות לכמה תת סוגים:

1. פקודת new – יוצרת צורה חדשה, מהסוג הנתון, לפי הדוגמאות הבאות:

```
new Circle color x y radius

new Ellipse color x1 y1 x2 y2 D
```

אנו נעבוד עם ההגדרה הבאה של אליפסה: יהיו שתי נקודות "מוקד" במישור הממשי, ויהי מספר ממשי D. אליפסה היא אוסף הנקודות במישור שסכום מרחקיהן משתי נקודות המוקד הוא D. אלו מכם שהחלידו מעט בגיאומטריה יכולים להשלים ידע למשל מהמאמר בוויקיפדיה, שגם מסתמך על ההגדרה הזאת:

https://en.wikipedia.org/wiki/Ellipse

```
new Parallelogram color x1 y1 x2 y2 x3 y3
```

פקודה זו יוצרת מקבילית חדשה (Parallelogram בלעז) ששלושה מקדקודיה נמצאים בקואורדינטות הנתונות, והקדקודים הראשון והשלישי הם מנוגדים. במצב זה הקדקוד הרביעי נגזר באופן חד ערכי ממיקומי שלושת הקדקודים האחרים. דרך פשוטה לחשב את מיקום הקדקוד הרביעי היא בעזרת חיבור וקטורים, שקיבל בצדק את השם "כלל המקבילית".
תזכורת שימושית: שטח מקבילית ניתן לחישוב בתור הערך המוחלט של דטרמיננטה. ניתן לבדוק האם נק' נמצאת בתוך מקבילית ע"י חלוקה של המקבילית לשני משולשים.

```
new Rectangle color x1 y1 x2 y2
```

בדומה למקבילית, יוצר מלבן ששניים מקדקודיו במיקומים הנתונים (ומכאן נגזרים שני הקדקודים הנותרים – ניתן להניח שהקדקודים הנתונים הם מנוגדים). צלעות המלבן מקבילות לצירים.

new Square color x1 y1 length

יוצר ריבוע שמרכזו במיקום הנתון, ואורך צלעו נתון. צלעות הריבוע מקבילות לצירים.

new Triangle color x1 y1 x2 y2 x3 y3

יוצר משולש שקדקודיו במיקומים הנתונים. ניתן להניח ששלושת המיקומים אינם על אותו ישר.
תזכורת שימושית: ניתן להשלים משולש למקבילית, ואז שטח המשולש הוא חצי משטח המקבילית. קל לבדוק האם נק' נמצאת בתוך משולש בעזרת חישובי שטחים. החישוב הנ"ל מצריך השוואה בין שני מספרים שבריים, והדרך לעשות זאת היא לבדוק האם הם קרובים עד כדי אפסילון כלשהו; עקרונית, לעולם לא נשווה בקוד floats או doubles באמצעות אופרטור == .

פקודת new מדפיסה למסך את ה-ID של הצורה החדשה שנוצרה. ה-ID חייב להיות ה-ID המינימלי שטרם נעשה בו שימוש, כלומר ה-ID's מחולקים 0, 1, 2....

1.1. צבעים: יש לתמוך ב-4 צבעים: BLUE, RED, YELLOW, GREEN.
1.2. כל הקואורדינטות והגדלים הגאומטריים בתרגיל נתונים כמספרים ממשיים.

2. פקודת delete מקבלת ID יחיד של צורה שטרם נמחקה, ומוחקת את הצורה מהשרטוט. ניתן להניח שה-ID יהיה תקין. היא אינה מדפיסה דבר למסך.

3. פקודת move מקבלת ID של צורה שטרם נמחקה, ומזיזה את הצורה באופן הבא:

Move 7 -3.14 4.6

פקודה זו מזיזה את הצורה בעלת ה-ID 7. היא מזיזה אותה 3.14 "יחידות" אחורה בציר ה-X, ו-4.6 יחידות "קדימה" בציר ה-Y. היא אינה מדפיסה למסך דבר.

4. פקודת copy זהה לפקודת move, אך היא יוצרת עותק חדש של הצורה ש"מוזז" בקואורדינטות הנתונות, ובנוסף משאירה את העותק הקיים במקום. היא מדפיסה למסך את ה-ID של הצורה החדשה שנוצרה.

5. פקודת area מחשבת את השטח המצטבר מצבע מסוים בשרטוט, באופן הבא:

area BLUE

מחשבת ומדפיסה את כלל השטח בשרטוט שמכוסה בכחול. (וכמובן באופן זהה לצבעים אחרים).
השטח צריך להיות מחושב בעזרת משתנים מסוג double, ולהיות מודפס עם שתי ספרות (בדיוק) אחרי הנקודה העשרונית.

6. פקודת color משנה את הצבע של צורה באופן הבא:

Color BLUE ID

משנה את הצבע של הצורה בעלת ה-ID הנתון לכחול. ניתן להניח שהצבע החדש שונה מהצבע הקודם של הצורה.

7. פקודת circumference עובדת באופן שקול לחלוטין לפקודת AREA, אך מחשבת ומדפיסה את ה<u>היקף</u> המצטבר של הצורות בעלות הצבע הנתון. גם כאן, ההיקף צריך להיות מחושב בעזרת משתנים מסוג double, ולהיות מודפס עם שתי ספרות (בדיוק) אחרי הנקודה העשרונית.
היקף אליפסה לא ניתן להבעה במקרה הכללי כנוסחה פשוטה. על מנת לחשב אותו, יש לסכום מספר רב של איברים ראשונים מהטור האינסופי הבא (ניתן שוב להשלים ידע מהמאמר בוויקיפדיה במידת הצורך):

$$C = \pi(a+b)\left[1 + \sum_{n=1}^{\infty}\left(\frac{(2n-1)!!}{2^n n!}\right)^2 \frac{h^n}{(2n-1)^2}\right].$$

לעומת זאת, היקף מעגל ניתן להבעה במדויק, ויש להשתמש בנוסחה המדויקת עבורו.
שימו לב שעצרת כפולה מוגדרת בעזרת הנוסחה הבאה:

1!! = 1
2!! = 2
For all n > 2: n!! = n * (n-2)!!

8. פקודת is_inside מקבלת ID וקואורדינטה, ומדפיסה 1 או 0 בהתאם להאם הקואורדינטה נמצאת בתוך הצורה עם ה-ID הנתון.

10. התוכנית מסתיימת כשהיא מקבלת את הפקודה EXIT.

<u>הלכה למעשה - Polymorphism</u>

שימו לב שהקוד שמטפל בפקודות area ו-circumference <u>לא כולל התייחסות לצורות ספציפיות</u>. הקוד צריך לא לכלול בכלל שמות של מחלקות של צורות ספציפיות.

בהצלחה!