תכנות מונחה עצמים מתקדם

עבודת הגשה מס' 2

להגשה עד ה 11/06/20 ב-23:55

דגשים להגשה

- לכל שאלה לפנות לציון במייל. **על כל פניה** להכיל את פרטי הסטודנט המלאים כולל ת.ז.
- חובה לתעד כל קובץ, מחלקה ופונקציה ע"י javaDoc
  ניתן להיעזר בתיעוד באתר oracle או בקבצים הרלוונטיים במודל
- העבודה מתבסת על עבודת הגשה 1 – עליכם לעדכן/להרחיב את המחלקות הקיימות במידת הצורך ולהשתמש בהן. כל המחלקות החדשות ניתן לארגן בחבילה בשם-graphics

1. מבוא

זהו התרגיל השני בקורס ,בתרגיל זה נתרגל יישום של ממשקים עם שימוש ב-GUI.

מטרת התרגיל:

- בניית סביבת **GUI**.
- יישום ממשקים.
- שימוש ב-Exceptions.

יש לבצע את העבודה לפי הדרישות המצוינות במסמך הנתון. בעבודה זאת יש לממש את ממשק GUI לניהול **טורניר בע"ח** ע"י שימוש בפעולות המוגדרות עליו.

2. הגדרות כלליות:

אתם מייבאים את כל המחלקות ישר מן התרגיל הקודם.

- מחלקת נחש ותנין שתיהן מממשות את ממשק זוחל.
- מחלקת נחש – רמת ארסיות יש שלוש רמות: נמוך בינוני וגבוה
- בנוסף מחלקת תנין תהיה גם ע"ח יבשתי וגם ימי. יש להתאים את המחלקות ולבצע פה "הורשה מרובה" תוך שימוש התבנית העיצוב delegator.

בנוסף יש להגדיר את ממשקים ומחלקות (ב- **graphics** package):

- public class **CompetitionFrame** extends JFrame – המחלקה תכלול פונקציית ***main***
- public class **CompetitionPanel** extends JPanel
- public class **AddAnimalDialog** extends JDialog
- public interface **IDrawable**
- public interface **IAnimal**
- public interface **IClonable**

1. ממשק **IDrawable** צריך להגדיר בצורה הבאה:

```
public interface IDrawable {
        public final static String PICTURE_PATH = "…";
        public void loadImages(String nm);
        public void drawObject (Graphics g);
}
```

2. ממשק **IAnimal** צריך להגדיר בצורה הבאה

```
public interface IAnimal {
        public boolean eat(int energy);
}
```

3. ממשק **IClonable** : ניתן לעשות שימוש במה שקיים ב-Java.

   מחלקות Point יממשו את הממשק הנ"ל(בנוסף למחלקות שיצוינו בהמשך).

4. בנוסף להגדרה של **Animal** מתרגיל הקודם: בתרגיל הזה **Animal** מממש גם את

```
public abstract class Animal extends Mobile implements ILocatable,
IDrawable, IClonable, IAnimal {  … }
```

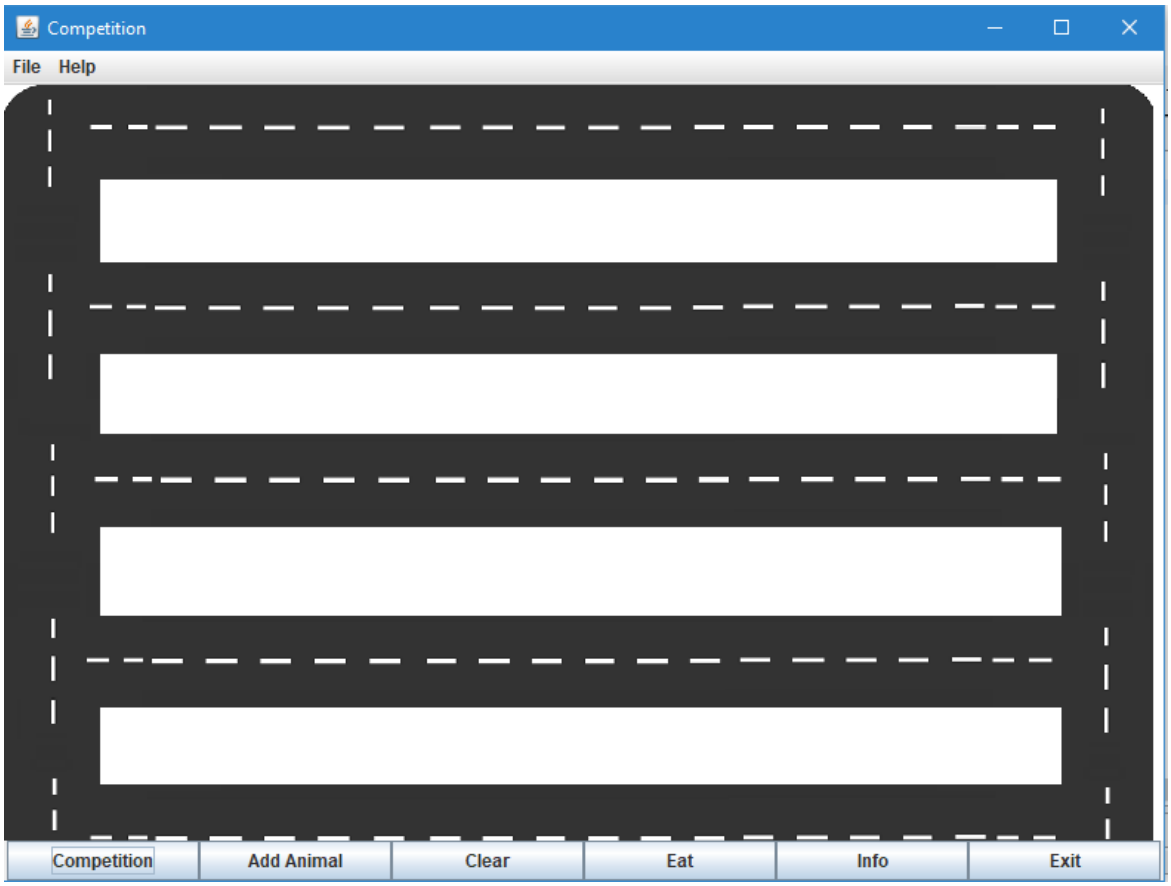
5. במחלקה **Animal** יש להגדיר את השדות הנוספים הבאים:
   - רמה מקסימלית קבועה של אנרגיה **(maxEnergy)**
   - כמות האנרגיה שהחיה צורכת למטר **(energyPerMeter)**

   (עבור כל מטר שהחיה זזה רמת האנרגיה שלה יורדת בהתאם).

```
        protected int size;
        protected Orientation orien;
        protected int maxEnergy;
        protected int energyPerMeter;
        protected CompetitionPanel pan;
        protected BufferedImage img1, img2, img3, img4;
```

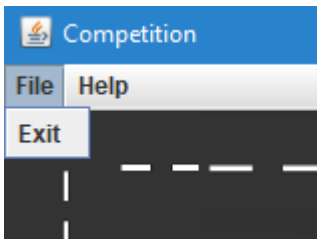
במחלקת Animal ולכל מחלקות של חיות יש להגדיר בנאים נוספים לאתחול שדות וגם להוסיף פונקציות לפי צורך. **ניתן להרחיב גם מחלקות וממשקים אחרים.**

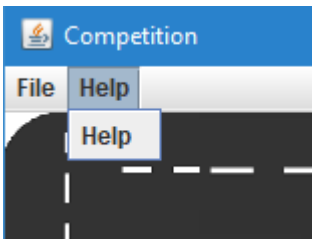
3. הגדרות ה- GUI עבור התרגיל :



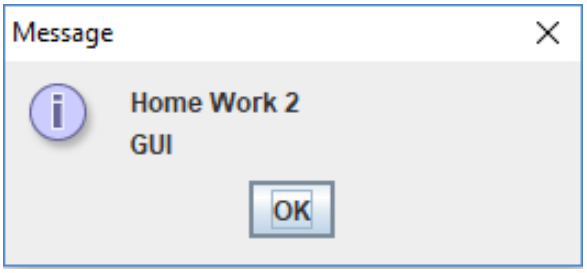
- ה- **CompetitionFrame** כולל menu ו **CompetitionPanel** עם מספר כפתורים הנמצאים בתחתית הפאנל.
- ה-Background מעודכן להיות תמונת המסלולים בתחילת התכנית.
- כל השטח ה-**CompetitionFrame** מיועד לציורי בע"ח.
- קיימים 3 סוגי מסלולים:
  - 5 מסלולים מקוקווים לבע"ח אווירים.
  - 4 בריכות לבע"ח ימיים
  - מסלול מעגלי לבע"ח יבשתיים

ב-**menu** יש להגדיר את האפשרויות הבאות:





- **Exit** – סגירה של התכנית
- **Help** – לפתיחת *showMessageDialog* בצורה הבאה:



- **הכפתורים** שנמצאים בתחתית ה- **CompetitionPanel** מבצעים את הפעולות הבאות:
  - **Competition** פותח חלון של **CompetitionDialog**, שמאפשר להגדיר ,תחרות חדשה (אוויר, מיים, יבשה)
-
  - **AddAnimal** – פותח חלון של **AddAnimalDialog**, שמאפשר להגדיר בע"ח חדש עם פרמטרים הבאים:

1. בחירת סוג של **חייה**: יבשתית, אווירית או מיים עם בדיקת התאמה לסוג התחרות. במידה וסוג התחרות לא מתאים עבור בעה"ה שנמצא על המסך יש לזרוק חריגה שתוביל להצגת *showMessageDialog עם הודעה מתאימה. אחרת, במידה וסוג התחרות מתאים יש למקם את החייה לתחילת המסלול שלה.*
2. המיקום ההתחלתי:
   - עבור תחרות יבשתית מיקום כל החיות הוא (0,0).
   - עבור תחרות אווירית/ מיים יש לבחור מסלול אווירית מ-1 ל-5, מיים מ-1 ל-4. מיקום בתחילת המסלול (x, 0) הערך x תלוי במסלול.
3. כל החיות שיצירתם מתחילים את תנועתם לכיוון מזרח.
4. את כל החיות מציירים על המסך בגודל של 65 פיקסלים.

  - **Clear** – מחיקת חיה מהפאנל.
  - **Eat** – אוכל. מבקש מספר שלם ומעלה את רמת האנרגיה של אותו בע"ח בהתאם לערך שקיבל עד למקסימום שלו.
  - **Info** – מופיע כל המידע על בע"ח בצורת ה-JTable (בחלון נפרד או ב-) **CompetitionPanel** :

Animal	Category	Type	Speed	Energy Amount	Distance	Energy Comsumpt.

Animal: שם החייה (לדומה שחורי ולא הסוג שלה)

Category: Air, Water, Terrest

Type: Dog , Cat, ...

Speed: מהירות החייה

Energy Amount: כמות האנרגיה הנוכחית.(תתעדכן לאחר כל תזוזה)

Distance: סה"כ המרחק שכעה"ח עבר

Energy Consumption: סה"כ כמות האנרגיה שהחיה איגרה (לאחר כל אכילה כמות האנרגיה שקיבלה מתווספים לסכום).

כאשר בע"ח נמחק מהפאנל, המידע השמור עליו יישאר טבלה כפי שהיה לפני שנמחק.

  - **Exit** –סגירה מסודרת של התוכנית.

4. הגדרות הגרפיקה עבור התרגיל:

- לכל בע"ח יש לממש פונקציית **drawObject()** (המוכרזת בממשק **IDrawable**) ולקרוא לה ב- **CompetitionPanel.paintComponent()**.

בשביל ציורי בע"ח יש להשתמש ב- **BufferedImage**. דוגמא של שימושו:

1) **הגדרת image**:

```
private BufferedImage img=null;
```

2) **טעינה של תמונה**:

```
  try { img = ImageIO.read(new File(BACKGROUND_PATH)); }
  catch (IOException e) { System.out.println("Cannot load image"); }
```

פה BACKGROUND_PATH זה full path של קובץ, למשל "*D://image.png*" .
באופן דומה טוענים תמונות לרכבים.

3) **ציור של תמונה כרקע**

```
  public void paintComponent(Graphics g)   {
    super.paintComponent(g) ;
    g.drawImage(img,0,0,getWidth(),getHeight(), this);
    .....
}
```

4) **לכל בע"ח יבשתי יש 4 שדות מסוג BufferedImage** :

img1 לצורך תנועה לכיוון מזרח

Img2 לצורך תנועה לכיוון דרום

Img3 לצורך תנועה לכיוון מערב

Img4 לצורך תנועה לכיוון צפון

לשאר בעלי חיים יש רק שדה אחד img1

דוגמא של פונקציה **drawObject (Graphics g)** במחלקה **Animal**:

```
public void drawObject (Graphics g)
{
   if(orientation==EAST) // animal move to the east side
      g.drawImage(img1, location.x, location.y-size/10, size*2, size,
pan);
      else if(orientation==SOUTH) // animal move to the south side
      g.drawImage(img2, location.x, location.y-size/10, size, size,
pan);
      else if(orientation==WEST) // animal move to the west side
      g.drawImage(img3, location.x, location.y-size/10, size*2, size,
pan);
   else if(orientation==NORTH) // animal move to the north side
      g.drawImage(img4, location.x-size/2, location.y-size/10, size,
      size*2, pan);
}
```

בדוגמא הזו משתמשים במשתנים הבאים:

- **size** – גודל של בעה"ח (protected data member של מחלקת **Animal**)
- **location.x** ו-**location.y** - קואורדינטות של "פה" בעה"ח (**protected Point location** הוא שדה של מחלקת **Animal** , הקואורדינטות האלה משתנים בפונקציית **move.**

5. הגדרות תנועת בע"ח על המסלולים:

- כל בע"ח ינוע באופן הבא: התזוזה תתבצע בהתאם למהירות בע"ח - כלומר שינוי הקואורדינטות יהיה לפי המהירות המתאימה.
- <u>עבור תחרות יבשתית בלבד:</u> כאשר קדמתו של בע"ח נמצא באחת הקצוות של המסך(דורש חישוב מתאים), על בעה"ח לשנות את כיוונו לפי כיוון המסלול וכמו כן תמונת בעה"ח צריכה להשתנות בהתאם.
- ברגע שלבע"ח נגמרת האנרגיה – מפסיק תנועתו על המסלול והוא עומד במקום.

התרגיל הזה ישמש כבסיס לתרגיל הבא, לכן יש להשקיע בכתיבת קוד איכותי.
בתרגיל הבא תתבקשו להרחיב את המימוש ולהוסיף תבניות עיצוב שונות.

עבודה נעימה