**תכנות מונחה עצמים מתקדם**

**עבודת הגשה מס 2'**

**להגשה עד ה- 18/04/2017 ב- 23:55**

**נא לקרוא את כל המסמך עד סופו לפני תחילת העבודה!**

**דגשים להגשה**

* ניתן להגיש עבודה זו בזוגות – רק אחד מהסטודנטים יגיש את העבודה במודל. בתיעוד של קובץ יש לציין שם, ת.ז וקמפוס של מגיש\ים, בתוך תיעוד הjavadoc
* על העבודה לעבוד בצורה מדויקת עם קובץ הprogram
* שימו לב כי מאחר ויש בעבודה שימוש בrandom ובsystem.err.println – הפלט לא יהיה זהה לקובץ המצורף! כמו כן, העבודה מורכבת יותר מהקודמת ולכן חלק מההדפסות תלויות בצורת המימוש שלכם. עליכם עדיין להיצמד לפלט זה ככל הניתן.
* לכל שאלה אנא עברו על מסמך ה[FAQ](https://docs.google.com/document/d/1rq4r5ogEOLsVdRQ3l1WSAR236tPd2hD4btRqvRBDaOk/edit?usp=sharing) ובדקו האם היא נענתה כבר. במידה ולא, ניתן להוסיף הצעת עריכה בקובץ או לפנות למתרגל האחראי – בר אוחיון במייל baroh@ac.sce.ac.il  
  **על כל פניה** להכיל את פרטי הסטודנט המלאים כולל ת.ז, קמפוס, ושם המתרגל.
* חובה לתעד כל קובץ, מחלקה ופונקציה ע"י javaDoc  
  ניתן להיעזר [בתיעוד באתר oracle](http://www.oracle.com/technetwork/java/javase/documentation/index-137868.html) או בקבצים הרלוונטיים במודל

**דגשים לעבודה זו**

* העבודה מתבססת על עבודת הגשה 1 – עליכם לעדכן את הקבצים הקיימים ולהשתמש במחלקות שכבר בניתם.
* עליכם לבנות היררכיית מחלקות בצורה הטובה ביותר, ולהימנע משכפול קוד.
* ניתן להוסיף מחלקות עזר – תחת package privateutil.
* על כל העבודה להיות פרויקט יחיד המחולק לpackages לפי המטלות.
* על הקבצים המצורפים MessageUtility, Program להיכלל תחת package utilities.  
  על הקבצים Plant,Cabbage,Lettuce להיכלל תחת package plants
* על כל השדות בכל המחלקות להיות פרטיים בלבד – למעט שדה השם, שניתן להגדיר כprotected ע"מ לשלוח אותו לloggers השונים ללא שימוש בthis.getName.
* על כל הSettres להיות בוליאניים, לבדוק תקינות נתונים (אם הוגדרה) ולהחזיר האם בוצעה השמה.
* יש להשתמש במחלקת MessageUtility המצורפת בכל הפונקציות הבאות:
  + Constructors (עבור חיות)
  + Getters & Setters (עבור חיות)
* Boolean Functions (גם עבור מחלקות תזונה)  
  עליכם להיעזר בClass Diagram והקבצים המצורפים ע"מ להבין כיצד לממש את ההיררכיה בתוכנית – לא הכל מפורט במסמך!
* עליכם להימנע משכפול קוד ככל שניתן.

1. **מזון: package food**
   1. **IEdible** – ממשק המתאר אובייקטים שניתן לאכול.
      1. Methods
         * getFoodtype():EFoodType
   2. **Enum EFoodType** – סוגי המזון הקיימים:
      1. ***MEAT*** - כל החיות למעט אריות. (אף חיה בגן החיות לא אוכלת אריות.)
      2. ***NOTFOOD*** – אריות.
      3. ***VEGETABLE*** - כל הצמחים.
2. **תזונה: package diet** [בחבילה זו אין צורך בlogConstractor]
   1. **IDiet** – ממשק המתאר פונקציונליות של אכילה.
      1. Methods
         * canEat(EFoodType food): boolean
         * eat(Animal animal, IEdible food): boolean
   2. **Carnivore** - מחלקה המגדירה אוכלי בשר. מממשת את הממשק IDiet.
      1. מידע נוסף:
         * אכילה גורמת לעלייה במשקל של 10% ממשקל הגוף הנוכחי.
   3. **Omnivore** - מחלקה המגדירה אוכלי כל. מממשת את הממשק IDiet. (יש להימנע משכפול קוד משאר המחלקות בקטגוריה זו!)
      1. מידע נוסף:
         * אכילת בשר גורמת לעלייה במשקל של 10% ממשקל הגוף הנוכחי, ואכילת ירקות גורמת לעלייה ב7% ממשקל הגוף הנוכחי.
   4. **Herbivore** - מחלקה המגדירה אוכלי עשב. מממשת את הממשק IDiet.
      1. מידע נוסף:
         * אכילת צמחים גורמת לעלייה במשקל של 7% ממשקל הגוף הנוכחי
3. **צמחים: package plants** – [אין צורך לממש, כל החבילה בקבצים המצורפים]
   1. **Plant** – מחלקה המגדירה את המאפיינים המשותפים לכל הצמחים. מממשת את ILocatable וIEdible.
   2. **Cabbage** - כרוב
   3. **Lettuce** - חסה
4. **תנועה: package mobility** [ אין צורך בLogger כלל. ]
   1. **ILocatable** – ממשק המתאר פונקציונליות של מיקום.
      1. Methods
         * getLocation(): Point
         * setLocation(Point): boolean
   2. **Mobile** – מחלקה מופשטת המגדירה תנועה במרחב ומממשת את הממשק ILocatable.
      1. Attributes
         * location : Point // Current location
         * totalDistance : double // Distance the object traveled, [>0]
      2. Methods
         * Mobile(Point) - Constructor(location)
         * addTotalDistance(double): void – increases td after movement
         * calcDistance(Point):double – same as HW1
         * move(Point): double – returns distance traveled (0 if non)
      3. מידע נוסף:
         * שינוי במיקום בהכרח גורר שינוי ב totalDistance
   3. **Point** - [מעבודה קודמת \ בקבצים המצורפים].
      1. מידע נוסף:
         * ציר X נע בין 0 ל - 800
         * ציר Y נע בין 0 ל – 600
5. **חיות: package animals**
   1. **Animal** – מחלקה מופשטת המגדירה את המאפיינים המשותפים לכל החיות. המחלקה מרחיבה את Mobile ומממשת את IEdible
      1. Attributes:
         * name: String
         * weight: double
         * diet: IDiet - used for Eating appropriate food
      2. Methods
         * Animal(String, Point) //Constructor(name, location)
         * makeSound():void
         * eat(IEdible):boolean – eat some food
      3. מידע נוסף:
         * כל החיות (למעט אריות) הן אכילות וסוגן הוא בשר

**Lion** – אוכל בשר, משקל התחלתי 408.2 ק"ג מיקומו התחלתי x: 20, y: 0

* + 1. Attributes:
       - scarCount: int – (starts at 0)
    2. Methods
       - Lion(String) // Constructor(name)
       - roar() : void – see output
    3. מידע נוסף:
       - אינו ניתן לאכילה ע"י חיות אחרות
       - בכל פעם שהאריה אוכל, יש הסתברות של 50% שתתווסף לו צלקת כתוצאה מהקרב עם הטרף. (השתמשו במחלקה Random)
  1. **Bear** – אוכל-כל, משקל התחלתי 308.2 ק"ג מיקומו התחלתי x: 100, y: 5
     1. Attributes
        + furColor : String
     2. Methods
        + roar() : void – see output
     3. מידע נוסף:
        + צבע הפרווה של דובים יכול להיות רק שחור, אפור או לבן. (ברירת מחדל אפור)  
          {"BLACK", "WHITE", "GRAY"}
  2. **Elephant** - אוכל עשב, משקל התחלתי 500 ק"ג מיקום התחלתי x: 50, y: 90
     1. Attributes
        + trunkLength: double
     2. Methods
        + chew() : void – see output
     3. מידע נוסף:
        + אורך החדק של פיל יכול לנוע בין 0.5 ל 3 מטר. (ברירת מחדל 1)
  3. **Giraffe** - אוכל עשב, משקל התחלתי 450 ק"ג מיקום התחלתי x: 50, y: 0
     1. Attributes
        + neckLength: double
     2. Methods
        + chew(): void – see output
     3. מידע נוסף:
        + אורך הצוואר של ג'ירף נע בין 1 ל 2.5 מטר (ברירת מחדל 1.5)
  4. **Turtle**– אוכל עשב, משקל התחלתי 1 ק"ג מיקום התחלתי x:80, y: 0
     1. Attributes
        + Age: int
     2. Methods
        + chew(): void – see output
     3. מידע נוסף:
        + גילו של צב יכול לנוע בין 0 ל500 שנה. (ברירת מחדל 1)
  5. **התנהגות כללית של החיות בספארי**
     1. לאחר אכילה כל חיה משמיעה את הקול שלה – על makesound להפעיל את הפונקציות roar/chew בהתאם. אין לממש את הפונקציה makeSound במחלקות של החיות עצמן!.
     2. על הפונקציות roar/shew להשתמש ב MessageUtility.logSound
     3. לאחר אכילה משקל החיה משתנה בהתאם לפירוט בסעיף תזונה
     4. לאחר תנועה משקל החיה משתנה בהתאם לנוסחהweight-(distance\*weight\*0.00025)

**עבודה נעימה!!!**