הנחיות לפתרון תרגיל הבית

- על הקוד המוגש להיות מתועד היטב ועליו לכלול:
  - מפרט, כפי שהודגם בתרגול.
  - תיעוד של כל מחלקה ומתודה ושל קטעי קוד רלוונטיים.
  - במידת הצורך, יש להוסיף תיעוד חיצוני.
- יש להפעיל את הכלי Javadoc כדי ליצור קבצי תיעוד בפורמט HTML ולצרף אותם לפתרון הממוחשב המוגש. כדי לגרום לקובצי ה-HTML להכיל את פסקאות המפרט שבהן אנו משתמשים, יש לציין זאת במפורש. ב-Eclipse, ניתן לבצע פעולה זו באופן הבא: 1. לבחור Export מתפריט File, לבחור Java->Javadoc וללחוץ על כפתור Next, 2. לבחור עבור Javadoc command את הקובץ javadoc.exe מתוך התיקייה bin הנמצאת בתיקייה שבה מותקן ה-Java SDK, 3. לבחור את הקבצים שלהם מעוניינים ליצור תיעוד וללחוץ פעמיים על כפתור Next, 4. להקיש ב-Extra Javadoc options את השורה הבאה וללחוץ על כפתור Finish:

-tag requires:a:"Requires:" -tag modifies:a:"Modifies:" -tag effects:a:"Effects:"

- התנהגות ברירת המחדל של פעולות assert היא disabled (הבדיקות לא מתבצעות). כדי לאפשר את הידור וביצוע פעולות assert, יש לבצע ב-Eclipse את הפעולות הבאות: 1. מתפריט Run לבחור Run Configurations, 2. בחלון שנפתח, לעבור ללשונית Arguments, 3. בתיבת הטקסט VM arguments לכתוב ea–, 4. ללחוץ על כפתור Debug.
- במהלך תרגיל בית זה אתם נדרשים לשרטט תרשימי UML. קיימים שני סוגי כלי עזר לשרטוט תרשימים כאלה:
  - תוכנות לשרטוט תרשימים כמו Visio, SmartDraw, Dia או draw.io.
  - כלים המצטרפים לסביבת פיתוח התוכנה. כלי חינמי אחד לדוגמה שיודע להצטרף בתור plugin ל-Eclipse הוא UMLet. לאחר ההתקנה של plugin זה, ניתן לשרטט תרשימי UML ב-Eclipse ע״י בחירה בתפריט ב: File->New->Other->UMLet Diagram. כלי זה מותקן בחוות המחשבים בפקולטה.

הנחיות להגשת תרגילי בית

- תרגילי הבית הם חובה.
- ההגשה בזוגות בלבד.
- עם סיום פתירת התרגיל, יש ליצור קובץ דחוס להגשה המכיל את:
  - כל קבצי הקוד והתיעוד.
  - פתרון לשאלות ה״יבשות״ בקובץ Word או PDF. על הקובץ להכיל את שמות ומספרי תעודות הזהות **של שני הסטודנטים** המגישים.
- הגשת התרגיל היא **אלקטרונית בלבד**, דרך אתר הקורס **ע״י אחד מבני הזוג בלבד**.

הקובץ המוגש יקרא hw4_<id1>_<id2>.zip כאשר <id1> ו-<id2> הם מספרי הזהות של הסטודנטים המגישים. לדוגמא hw4_12345678_9876541.zip (כמובן יש להשתמש במספרי הזהות שלכם).

- תרגיל שיוגש באיחור וללא אישור מתאים (כגון, אישור מילואים), יורד ממנו ציון באופן אוטומטי לפי חישוב של 5 נקודות לכל יום איחור ועד 2 ימי איחור שלאחריהם לא תתאפשר ההגשה כלל.
- על הקוד המוגש לעבור הידור (קומפילציה). על קוד שלא עובר הידור יורדו 30 נקודות.

מועד ההגשה:
יום ה', 25/1/18

המטרות של תרגיל בית זה הן להתנסות בתחומים הבאים:
- תכן של תוכנה תוך שימוש בעקרונות שנלמדו בקורס.
- הבנה ושימוש ב-design patterns.

שאלה 1 (10 נקודות)

נתון חלק מהמפרט והמימוש של המחלקות הבאות המאפשרות לברך אנשים:

```
abstract class Greeting {
	public void greet(String name) {
		System.out.println(name);
	}
}
```

```
class MaleGreeting extends Greeting {
	@Override
	public void greet(String name) {
		System.out.print("Hello Mr. ");
		super.greet(name);
	}
}
```

```
class FemaleGreeting extends Greeting {
	@Override
	public void greet(String name) {
		System.out.print("Hello Ms. ");
		super.greet(name);
	}
}
```

```
abstract class Person {
	private String name;

	public Person(String name) {
		this.name = name;
```

```
	}

	public String getName() {
		return name;
	}

	public abstract void greet();
}
```

```
class Male extends Person {
	public Male(String name) {
		super(name);
	}

	@Override
	public void greet() {
		new MaleGreeting().greet(getName());
	}
}
```

```
class Female extends Person {
	public Female(String name) {
		super(name);
	}

	@Override
	public void greet() {
		new FemaleGreeting().greet(getName());
	}
}
```

כמו כן נתונה מחלקה המשתמשת במחלקות לעיל:

```
public class Greetings {
	public static void main(String[] args) {
		Male male = new Male("Danny");
		Female female = new Female("Danna");
		male.greet();
		female.greet();
	}
}
```

א.

מה התוכנית תדפיס לאחר הרצתה?

ב.
איזה design pattern ממומש כאן? איזו בעיית תכן בא design pattern זה לפתור?

ג.
ציירו class diagram המכיל את המחלקות הנ"ל ואת הקשרים ביניהן והסבירו בעזרת התרשים את אופן פעולת ה-design pattern הממומש כאן. נדרש לתת הסבר עקרוני וממצה במספר שורות.

להגשה "יבשה": תשובות לשאלות הנ"ל.

שאלה 2 (35 נקודות)

א.
סטודנט תכנן מחלקה המייצגת עובד בשם Employee. המחלקה צריכה להיות מסוגלת לכתוב את נתוני העובד לבסיס נתונים או לקובץ XML. לכן, הוסיף הסטודנט למחלקה Employee מתודות בשם toDB() ו-toXML() לביצוע הפעולות הנ"ל. איזה עקרון SOLID מופר פה? הסבירו והציעו **במילים ובתרשים UML מתאים** פתרון חלופי בעל תכן מוצלח יותר.

ב.
סטודנט תכנן מחלקה המייצגת ריבוע כמחלקה יורשת ממחלקה המייצגת מלבן. איזה עקרון SOLID מופר פה? הסבירו.

ג.
הסבירו כיצד עקרונות הפתיחות/סגירות (open/closed principle) מ-SOLID ועיקרון היוצר מ-GRASP מתבטאים ב-Factory method design pattern.

ד.
בתרגיל בית מס' 3 ביצעתם תכן עבור תוכנה לניהול מסעדה. תנו דוגמה לביטוי של עקרון ה-Creator מ-GRASP בתכן שלכם ותנו דוגמה לביטוי של עקרון ה-Information Expert בתכן שלכם. בתשובתכם **עליכם לפרט ולצרף תרשימים** מתשובתכם לתרגיל 3.

ה.
לאובייקטים של ממשק משתמש מחבילת ה-Swing ניתן להוסיף מסגרת בעזרת המתודה setBorder(). למשל, הנה דוגמה של הוספת מסגרת שחורה לאובייקט כפתור מטיפוס JButton:

```
JButton button = new JButton();
Border border = BorderFactory.createLineBorder(java.awt.Color.black);
button.setBorder(border);
```

איזה design pattern ממומש בקטע זה? הסבירו.

ו.

המחלקה java.lang.System היא מחלקה בספריות הסטנדרטיות של Java המאפשרת שירותים שונים הנוגעים לקלט, פלט ועוד. אחת מהפונקציות שם היא getSecurityManager() שזהו המפרט שלה:

getSecurityManager

```
public static SecurityManager getSecurityManager()
```
Gets the system security interface.

Returns:
if a security manager has already been established for the current application, then that security manager is returned; otherwise, null is returned.

וריאציה של איזה design pattern ממומש כאן? הסבירו.

להגשה "יבשה": תשובות לשאלות הנ"ל.

שאלה 3 (55 נקודות)

בשאלה זו נתכנן ונממש מערכת של לוח מודעות אלקטרוני. לוח מודעות אלקטרוני מורכב מ-25 לוחות המסודרים ברשת של 5 על 5. כל לוח מאזין למצב של אובייקט בשם ColorGenerator המשנה את צבעו בכל שתי שניות. קיים רק אובייקט ColorGenerator יחיד במערכת. בכל פעם שהצבע משתנה, המופע היחיד של ColorGenerator מודיע ללוחות על הצבע החדש והם בתגובה משנים את צבעם לצבע זה. התוצאה היא שבכל שתי שניות לוח המודעות כלו משנה את צבעו לצבע הנקבע ע"י המופע של ColorGenerator.

בגרסה המקורית של התוכנה, ColorGenerator מודיע ללוחות על שינוי הצבע בסדר עולה: לוח1, לוח2, לוח3, ..., לוח25. שינוי הצבע בסדר זה משעמם ולכן הוחלט לאפשר סדרים שונים להודעה ללוחות. יאופשרו הסדרים הבאים:

- סדר עולה (1-25).
- שינוי בעמודות (1,6,11,16…2,7…)
- עדכון בשני מעברים (1,3,5,… 2,4,6…).
- עדכון בסדר אקראי.

עליכם לתכנן ולממש בקוד את לוח המודעות האלקטרוני (Billboard), הלוחות (Panels) ו-ColorGenerator תוך שימוש ב-design patterns הבאים: Singleton, Observer, Strategy.
יש להשתמש ב-Swing כדי להציג את לוח המודעות המורכב מלוחות המשנים את צבעם.

הנחיה: ניתן לגרום לפעולה מסוימת להתבצע בכל מספר קבוע של שניות באמצעות המחלקה javax.swing.Timer, כפי שנעשה בתרגיל בית 1. יש להשתמש במחלקה זו לצורך

תזמון שינוי המצב של ColorGenerator. כמו כן, יש להוסיף השהייה של 40 msec בין ההודעות של ColorGenerator ללוחות כך שניתן יהיה לראות את סדר עדכון הלוחות בעין.

עבור כל design pattern ברשימה לעיל הסבירו כיצד הוא ממומש בקוד שלכם, ואיזה בעייה הוא פותר.

הקפידו כי כל קבצים ה-java יהיו ב-package בשם: homework4 (יורדו נקודות למי שלא ימלא אחר הנחיה זו).

להגשה ממוחשבת: מימוש לוח המודעות האלקטרוני על כל רכיביו כולל תיעוד כפי שנלמד בקורס ועם abstraction functions, representation invariants וקריאות למתודה checkRep() במקומות המתאימים.
להגשה יבשה: הסבירו כיצד השתמשתם ב design patterns במימוש הקוד שלכם.