

תוכנה 1 – אביב 2019/20

תרגיל מספר 6

חוזים, מחלקות וממשקים

הנחיות כלליות:

קראו בעיון את קובץ נהלי הגשת התרגילים אשר נמצא באתר הקורס.

- הגשת התרגיל תיעשה במערכת ה-moodle בלבד (<http://moodle.tau.ac.il/>).
- יש להגיש קובץ zip יחיד הנושא את שם המשתמש ומספר התרגיל (לדוגמא, עבור המשתמש aviv1 יקרא הקובץ aviv1_hw6.zip). קובץ ה-zip יכיל:
 - א. קובץ פרטים אישיים בשם details.txt המכיל את שמכם ומספר ת.ז.
 - תיקיה בשם partsA-C ובתוכה היררכיית תיקיות שמכילה את קבצי הג'אווה Polynomial, SectionA, SectionB
 - תיקיה בשם dates ובה היררכיית תיקיות שמכילה 6 קבצי ג'אווה.
 - תיקיה בשם riddles ובתוכה היררכיית תיקיות: 4 התיקיות a,b,c,d, שבכל אחת מהן קבצי הג'אווה הרלוונטיים (שימו לב שאלה תיקיות שקיימות כבר בשלד שקיבלתם).
 - היררכיית התיקיות של 3 החלקים צריכה להיות בדיוק באותו אופן שקיבלתם אותה. כלומר, ה-zip שתגישו יראה בדיוק כמו ה-zip של השלד, רק שיכיל בתור קבצי ה-java את הקוד שלכם.
 - ב. קובץ התשובות המילוליות בשם answers.txt
- נא לא להשתמש בפקודה System.exit()! היא מחבלת בבדיקות אוטומטיות. אין כל צורך לעשות בה שימוש, כאשר תוכניות יכולות להסתיים ע"י הגעה לסוף מתודת ה-main.

ייבוא הפרויקט:

מומלץ ליצור 3 פרוייקטים שונים: אחד עבור החלק partsA-C, אחד עבור dates ואחד עבור riddles. בקובץ הזיפ שקיבלתם יש 3 תיקיות מתאימות עם השמות האלו, ובכל אחת מהן תיקיית src. עבור כל אחד מהפרויקטים קחו את תיקיית ה-src המתאימה, כאשר הנחיות יבוא הפרוייקט הן אותן ההנחיות כמו בתרגילים הקודמים. המחלקות לסעיפים א-ג נמצאות תחת partsA-C, המחלקות לחלק ד נמצאות תחת dates, והמחלקות לחלק ה נמצאות תחת riddles. יש להגיש את הקבצים בקי zip בדיוק באותה היררכיית תיקיות שקיבלתם בקי zip המקורי. כלומר תוכן zip שאתם מגישים צריך להיראות בדיוק כמו zip המקורי שקיבלתם בשלד, רק שקבצי ה-java צריכים להכיל את הקוד שלכם.

חלק א' (5%) – חוזים

נתונה מחלקה בקובץ SectionA עם משתמר מחלקה וחוזה עבור כל אחד משירותי המחלקה (מלבד מתודת העזר size שאתם יכולים להניח שהינה תקינה). עבור כל אחת מן השירותי מחלקה (מלבד size()) ציינו בקובץ answers.txt האם המימוש תואם את החוזה ואת משתמר המחלקה. אם לא, הראו דוגמא נגדית

והסבירו מה התיקון הנדרש בחוזה (שימו לב שהתיקון נעשה בחוזה ולא במימוש – בעקבות התיקון בחוזה, המימוש של המתודה כעת תואם את החוזה ומשתמר המחלקה מתקיים). כאשר אתם בודקים את תקינותה של מתודה מסוימת, הניחו ששאר המתודות מקיימות את החוזה, ושלפני הקריאה משתמר המחלקה מתקיים.

חלק ב' (5%) – מימוש מתודות על פי חוזה

עבור כל אחד מהמתודות המופיעות בקובץ SectionB ממשו את המתודות הנתונות בקובץ, על פי החוזה המפורט. שימו לב, כי כל מימוש אשר תואם את החוזה ואת החתימה נחשב תקין (כלומר אין מגבלות נוספות).

חלק ג' (20%) – מימוש מחלקה על פי מפרט

בחלק זה של התרגיל נממש מחלקה בשם Polynomial המייצגת פולינום עם מקדמים ממשיים. המחלקה תתמוך בשירותים הציבוריים אשר מופיעים בקובץ Polynomial.

המשימה:

ממשו את המחלקה Polynomial בצורה יעילה תוך שימוש במערכים בלבד. אין להשתמש במבנה נתונים נוסף. (התבססו על השלד הנתון). שימו לב שניתן להגדיר שירותי עזר.

חשוב: במידה והמתודה מחזירה פולינום, יש ליצור פולינום חדש ולא לשנות את הקיים. לדוגמא עבור המתודה adds, יש לחבר את הפולינום הקיים עם הפולינום שניתן כארגומנט ולהחזיר פולינום חדש, מבלי לשנות את הפולינום הקיים או את הפולינום שניתן כארגומנט.

שימו לב כי אנו מחשיבים את הדרגה של פולינום האפס בתור אפס (כמו גם הדרגה של כל פולינום שיש לו מקדם חופשי בלבד).

ניתן לבדוק את עצמכם בעזרת פונקציית ה-main הנתונה במחלקה Test. שימו לב כי בדיקה זו הינה חלקית בלבד. ניתן לשנות ולהוסיף לה בדיקות כרצונכם.

חלק ד' (50%) – Dates

המנשק Date המופיע למטה (עם מימוש חסר אותו תתבקשו להשלים) מייצג תאריך המורכב משלושה חלקים: יום, חודש ושנה. בתרגיל זה אנו מניחים לשם פשטות כי בכל שנה יש 365 ימים, ובפרט בחודש פברואר של כל שנה ישנם 28 ימים.

אנו נניח כי התאריך החוקי המוקדם ביותר הוא הראשון לינואר בשנה 1.

היום בחודש מיוצג על ידי מספר בין 1 ל-31. והחודשים מיוצגים על ידי מספר בין 1 (ינואר) ל-12 (דצמבר).

להזכירכם החודשים בהם יש 31 ימים הם: ינואר, מרץ, מאי, יולי, אוגוסט, אוקטובר ודצמבר. ואילו החודשים בהם יש 30 ימים הם: אפריל, יוני, ספטמבר ונובמבר. בפברואר כאמור יש 28 ימים.

ניתן להניח שבחישובים הקשורים לתאריכים הנעשים על משתנים מהטיפוס int לא יוצרו חריגות מטווח הערכים החוקי.

```
public interface Date {
```

```

public static int getDaysInMonth(int month) {
    return 0;
}

public String toString();

public int getDay();

public int getMonth();

public int getYear();

public void addDays(int days);

public default int DifferenceInDays(Date other) {
    return 0;
}

public default boolean isBetweenDates(Date date1, Date date2) {
    return false;
}
}

```

עליכם לממש את המנשק Date על ידי שלושה ייצוגים שונים: הראשון עושה שימוש במחרוזת, השני במערך של מספרים מסוג int ואילו השלישי משתמש ב-int יחיד המייצג את התאריך בתור מספר הימים שחלפו מאז הראשון לינואר בשנה 1 (הסבר מפורט בהמשך). לכל אחת מהמחלקות יהיה בנאי המתאים לייצוג הפנימי שלה וכמובן כל אחת מהן מממשת את המנשק. ניתן להניח שהבנאים מקבלים קלט תקין ליצירת Date (בהתאם לייצוג הפנימי של כל מחלקה). באופן כללי, ניתן להניח קלט חוקי לכל מתודה, אלא אם נאמר אחרת.

1. השלימו את המימוש של המתודה הסטטית getDaysInMonth במנשק Date. מתודה זו מקבלת חודש בתור int בין 1 ל-12, ומחזירה את מספר הימים בחודש הנתון.
2. כתבו מימוש למתודה הדיפולטית differenceInDays במנשק Date. מתודה זו מקבלת תאריך אחר בשם other ומחזירה int שערכו הוא מספר הימים בין התאריך של המופע עליו מופעלת המתודה לבין other. כאשר other הוא התאריך המאוחר יותר, הערך שיוחזר הוא חיובי, וכאשר הוא התאריך המוקדם מבין השניים יוחזר ערך שלילי. כמובן, שכאשר התאריכים זהים יוחזר 0 (כלומר קונספטואלית מחסרים מ-other את התאריך הנוכחי ואז ממירים את הפרש לימים).
3. כתבו מימוש למתודה הדיפולטית isBetweenDates במנשק Date. מתודה זו מקבלת שני תאריכים date1 ו-date2 ומחזירה true אם ורק אם התאריך של המופע הנוכחי נמצא בין 2 התאריכים האלה, כולל הקצוות (כלומר גם אם הוא זהה לאחד משני התאריכים). שימו לב, כי לא מובטח איזה משני הארגומנטים מייצג את התאריך המאוחר יותר, לכן יש להתמודד עם שתי האפשרויות.
4. השלימו את המימוש של המתודות במחלקה DateString, המממשת את המנשק בעזרת ייצוג פנימי של String. ספציפית, הפורמט שנניח הוא שלושה מספרים המופרדים על ידי התו "/" ללא רווחים או אפסים מובילים, כך שהמספר השמאלי ביותר הוא היום בחודש, המספר האמצעי הוא

החודש והמספר הימני הוא השנה. למשל, השני לנובמבר 2019 ייוצג ע"י המחרוזת "2/11/2019". כמו כן, המתודה toString במימוש זה וגם בשני המימושים האחרים תחזיר מחרוזת המייצגת את התאריך בדיוק בפורמט הזה. המתודות getDate, getMonth ו- getYear יחזירו את המספר המייצג את היום, החודש והשנה בהתאמה. המתודה addDays מקבלת פרמטר days מטיפוס int, ומשנה את התאריך של המופע הנוכחי בכך שהיא מוסיפה לו days ימים. אם days הוא שלילי אז התאריך החדש יהיה מוקדם יותר, אך בכל מקרה של חריגה שאמורה ליצור תאריך מוקדם יותר מה-1/1/1 התאריך החדש שיקבע במקום זאת יהיה בדיוק 1/1/1. כמובן כאשר days הוא אפס התאריך לא משתנה.

5. השלימו את המימוש של המתודות במחלקה DateArray, המממשת את המנשק בעזרת ייצוג פנימי של מערך בגודל 3 של int. התא הראשון מכיל את השנה, התא השני את החודש והתא השלישי את היום בחודש. ההנחיות מקבילות לאלו שניתנו במימוש הקודם.
6. השלימו את המימוש של המתודות במחלקה DateInt, המממשת את המנשק בעזרת int יחיד שמציין את מספר הימים שחלף מאז ה-1/1/1 (התאריך החוקי המוקדם ביותר לצורכי תרגיל זה). למשל התאריך 1/1/1 עצמו ייוצג על ידי המספר 0, ואילו התאריך 1/2/2 ייוצג על ידי המספר $31+365=396$. ההנחיות מקבילות לשני המימושים הקודמים.
7. ממשו את המחלקה DateFactory המגדירה את המתודות הבאות:

```
public class DateFactory {  
  
    public static Date createDate(String date) {  
        return null;  
    }  
  
    public static Date createDate(int[] date) {  
        return null;  
    }  
  
    public static Date createDate(int date) {  
        return null;  
    }  
}
```

כל אחת מהמתודות הסטטיות יוצרת אובייקט מטיפוס Date, כשהאובייקט הקונקרטי נקבע על סמך טיפוס הקלט.

הערה: מחלקה שתפקידה היחיד הוא יצור אובייקטים של מחלקות אחרות נקראת *factory class*. מחלקות אלו מסתירות את פרטי יצור האובייקטים מלקוחות של אובייקטים אלו. השימוש בטכניקה זו נועד להסתיר את המחלקות הקונקרטיות שמממשות מנשק.

מותר כרגיל להוסיף מתודות עזר, ומומלץ כמו תמיד שיהיו private.

הערה חשובה: עליכם לממש את המתודות באופן שונה בכל מחלקה בהתאם לייצוג הפנימי. אין להמיר את הייצוג הפנימי לייצוג אחר לצורך מימוש פעולה (רק לצורך פלט). כמובן שניתן למשל כחלק מחישובי הביניים להמיר מחרוזת למספרים, אך הייצוג שנשמר בשדות יהיה רק הייצוג המצוין בתרגיל. המחלקות השונות לא "יעזרו" זו בזו. **לא משתפים קוד בין מימושים.** אם זה יוצר שכפול קוד, זה בסדר לצרכי התרגיל. ובאופן כללי, אם יש ספק אם משהו נחשב כשיתוף בין מימושים (למשל מתודות עזר),

ההנחיה הכללית היא להעדיף ליצור שכפול קוד מאשר כל צורה של שיתוף. וזה בסדר גמור אם יהיו חלקים דומים או זהים בין המימושים. החלקים המשותפים היחידים הם המימושים של מתודות ה-default במנשק Date עצמו.

להלן התוכנית המצורפת בשם **TestDate** המדגימה את השימוש במחלקה DateFactory ובמנשק.

```
public class TestDate {

    public static void main(String[] args) {
        String d1 = "2/11/2019";
        int d2 = 390;
        int[] d3 = {1986, 9, 24};

        Date date1 = DateFactory.createDate(d1);
        Date date2 = DateFactory.createDate(d2);
        Date date3 = DateFactory.createDate(d3);

        System.out.println("date2 day: " + date2.getDay()); // 26
        System.out.println("date2 month: " + date2.getMonth()); // 1
        System.out.println("date2 year: " + date2.getYear()); // 2

        System.out.println("date3: " + date3); // 24/9/1986

        System.out.println("April: " + Date.getDaysInMonth(4)); // 30
        System.out.println("February: " + Date.getDaysInMonth(2)); // 28
        System.out.println("July: " + Date.getDaysInMonth(7)); // 31

        System.out.println("is date1 between date2 and date3: " +
            date1.isBetweenDates(date2, date3)); // false
        System.out.println("is date2 between date1 and date3: " +
            date2.isBetweenDates(date1, date3)); // false
        System.out.println("is date3 between date2 and date1: " +
            date3.isBetweenDates(date2, date1)); // true

        System.out.println("Difference in days between date1 and date3: "
            + date1.differenceInDays(date3)); //-12084

        date2.addDays(-400);
        System.out.println("date2 after subtracting 400 days: " + date2);
// 1/1/1

        date3.addDays(12084);
        System.out.println("date1 after adding 12084 days: " + date3); //
2/11/2019

        date1.addDays(-12084);
        System.out.println("date1 after subtracting 12084 days: " + date1);
// 24/9/1986
```

```
}  
  
}
```

הקובץ TestDate נועד לבדיקה עצמית בלבד, אותו אתם יכולים להרחיב ולשנות ולבדוק את עצמכם (ניתן להגיש גם אותו, אך הוא לא יבדק).

חלק ה' (20%) – חידות ג'אוה (תיקון מחלקות קיימות)

בכל סעיף של חלק זה תקבלו חבילה ובה מספר מחלקות. חבילות אלה מצורפות לתרגיל באתר הקורס. עליכם לשנות את הקוד בהתאם להנחיות, כדי לקבל את התוצאה הנדרשת. יש להגיש את כל המחלקות (כולל אלה שלא שיניתם בהם דבר, וכמובן, הקוד המתוקן). אין רווחים בין השורות של ההדפסות.

1. החבילה il.ac.tau.cs.sw1.riddle.a מכילה שתי מחלקות, A ו-B.

B היא תכנית המקבלת כארגומנט מס' שלם. עליכם לשנות את הקוד בתוך printA() ב-A בלבד כך שבהרצת פונקציית ה-main ב-B יודפס המספר שניתן כקלט בין A1 ל-A2. לדוגמא, אם הקלט הוא 15, יודפס:

```
B  
A1  
15  
A2
```

- **מותר:** לשנות את הקוד בתוך printA().
 - **אסור:** לשנות את B, את חתימת printA(), וקוד ב-A שנמצא מחוץ ל-printA().
2. החבילה il.ac.tau.cs.sw1.riddle.b מכילה שלוש מחלקות, A, B ו-C.

C היא תכנית המקבלת כארגומנטים שלוש מחרוזות. עליכם להשלים את מימוש המתודות printA, printA2 ו-printA3 ב-A כך שבהרצת פונקציית ה-main ב-C יודפסו 3 המחרוזות בין הכוכביות. לדוגמא, אם הקלט לתכנית הוא hello world bye, יודפס:

```
hello  
  
***  
  
world  
  
***  
  
bye
```

- **מותר:** לשנות את הקוד בתוך `printA`, `printA2` ו-`printA3`.
- **אסור:** לשנות את B ו- C, את חתימות המתודות `printA`, `printA2` ו-`printA3`, וקוד ב- A שנמצא מחוץ למתודות הנ"ל.

3. החבילה `il.ac.tau.cs.sw1.riddle.c` מכילה שתי מחלקות, A ו- B.

B היא תכנית. עליכם לשנות את חתימות המתודות והשדות ב- A כך ש-('א') הקוד יתקמפל ללא שגיאות, ו- ('ב') B תדפיס, כשורה אחרונה, `success!`. ייתכן שיודפסו שורות נוספות לפני שורה זו, המשמשות לבקרה בלבד (כל עוד מודפס `success!` הפתרון נכון).

- **מותר:** לשנות את חתימות המתודות ואת השדות, כולל: שינוי נראות (`public`-ל-`private` ולהפך), הוספת והורדת `static`, שינוי טיפוס ההחזרה של מתודה, ושינוי הארגומנטים למתודה.
- **אסור:** לשנות את B, לשנות קוד בתוך מתודות A, לשנות את שמות המתודות ב- A ולשנות את ערך השדה k.

4. החבילה `il.ac.tau.cs.sw1.riddle.d` מכילה שתי מחלקות, A ו- B.

A היא תכנית המדפיסה מס' שלם. בתוך קוד A מופיעות 4 קריאות לפונקציה `setI` של מחלקה B, עם הארגומנטים `k`, `B.I`, `j` ו- `l` בהתאמה. עליכם לשנות את הארגומנטים של `setI` כך שהקוד יתקמפל ללא שגיאות, ובסופו של דבר יודפס המס' 210. עדיין, הארגומנט של כל קריאה חייב להיות `I`, `j`, `k` או `l`.

- **מותר:** לשנות את הארגומנטים של `setI`, למשל במקום בו הופיע `setI(j)` אפשר לשנות ל-`setI(k)`. אם יש צורך ניתן להוסיף שם מחלקה או מופע לפני שדה, למשל `B.I` או `this.l`.
- **אסור:** להשתמש במשתנים\שדות מלבד הארבעה הנ"ל, או במס' טבעיים. אין לשנות את B או כל קוד ב- A מלבד הארגומנטים המועברים ל- `setI`.

בהצלחה!