

תוכנה 1 – חורף תשע"ו

תרגיל מספר 5

הנחיות כלליות:

- קראו בעיון את קובץ נהלי הגשת התרגילים אשר נמצא באתר הקורס.
- הגשת התרגיל תעשה במערכת ה-moodle בלבד (<http://moodle.tau.ac.il/>).
- יש להגיש קובץ zip יחיד הנושא את שם המשתמש ומספר התרגיל (לדוגמא, עבור המשתמש aviv יקרא הקובץ aviv_hw5.zip). קובץ ה-zip יכיל:
 - קובץ פרטים אישיים בשם details.txt המכיל את שמכם ומספר ת.ז.
 - קבצי ה-java של התוכניות אותם התבקשתם לממש.

הגשת מחלקה עם חבילות: יש לכווץ בתוך קובץ ה-zip שאתם מגישים את כל היררכיית התיקיות מתחת ל-src באקליפס. למשל, כדי להגיש את המחלקה sw1.pac.MyClass העתיקו את התיקיה sw1 שמתחת ל-src כולל כל מה שבתוכה לתוך קובץ ה-zip.

בתרגיל זה נעסוק במערכת של המלצות על ספרים ונממש שיטה מאוד טריוויאלית למתן המלצה על ספר על סמך של דירוגים של משתמשים שונים במערכת. Amzone ו Netflix הן שתי מערכות גדולות אשר משתמשות בדירוגי משתמשים למתן המלצות על התכנים שהן מציעות. כמובן שהאלגוריתמים שבהם מערכות אלא משתמשות הם יותר מתוחכמים ממה שאנחנו ניראה.

מידע של ספרים, משתמשים ודירוגים

באתר הקורס ניתן למצוא שלושה קבצי csv אשר מכילים מידע על ספרים, משתמשים והדירוגים שלהם. קבצי מידע אלה נוצרו מתוך בסיס הנתונים Book Crossing DataSet שנוצר על ידי קבוצת מחקר מאוניברסיטת פרייבורג (Institut für Informatik, Universität Freiburg). בסיס הנתונים הזה זמין בחינם למטרות מחקר, וניתן הוריד אותו בשלמותו מהקישור הבא:

<http://www2.informatik.uni-freiburg.de/~chiegler/BX>

ניתן להתרשם מתוכן הקבצים על ידי פתיחתם ב Excel. כמו כן, ניתן לצפות בתוכנו של קובץ באמצעות כל עורך טקסט כמו notepad++.

בתרגיל זה נשתמש בשלוש מחלקות: Book, User, BookRecommendations. מחלקות אלה נמצאות ב il.ac.tau.cs.sw1.ex4. שימו לב – כמו בתרגיל 4, אין להשתמש במבני נתונים גנריים כמו Set, Map, List וכו'.

המימוש של המחלקה Book נתון בחלקו, עליכם להשלים אותו וכן להשלים את מימושו של שתי המחלקות האחרות.

מערכת המלצה על ספרים

(1) [10 נק'] השלימו את המימוש של המחלקה User על פי המפרט הבא:

שדה	משמעות
<code>int userID</code>	מזהה יחודי של המשתמש
<code>String Locations</code>	המיקום המשוויך למשתמש זה
<code>int age</code>	גיל המשתמש

הנכם רשאים להוסיף שדות נוספים לפי ראות עיניכם.

הגדירו שני בנאים למחלקה על פי החתימות הבאות:

```
public User(int userID, String location, int age)
```

```
public User(int userID, String location)
```

הבנאי השני אינו מקבל ארגומנט עבור הגיל של המשתמש, ולכן עליו לאתחל אותו באופן דיפולטי ל 1- (השתמשו בקבוע NO_AGE אשר מאותחל במחלקה)

ממשו גם getter-ים (מתודות get) עבור שלושת השדות של המחלקה.

ניתן להעזר ב Source->Generate Getters and Setter על מנת לייצר את הקוד של ה getter-ים.

לנוחותכם, ממשו את המתודה toString עבור המחלקה User. מימוש זה אינו חובה ואנחנו לא מגדירים פורמט אחיד לייצוג האובייקט באמצעות מחרוזת. במחלקה Book מופיע מימוש ל toString ואתם יכולים להעזר בו.

בנוסף, ממשו את המתודה hasAge אשר תחזיר true אם השדה age מכיל ערך חוקי (כלומר, ערך שאינו NO_RATING), ו false אחרת. הסיבה להוספת מתודה זו היא עיצוב נכון יותר של המחלקה. המשתמש באובייקט מסוג User לא אמור לדעת כיצד בחרנו לייצג את גילו של משתמש שלא קיים עבורו ערך של גיל. מתודה זו תאפשר לו לדעת אם גילו של המשתמש נקבע מבלי להיחשף לפרטי המימוש שלנו (שימוש בערך 1-).

חתימת המתודה:

```
public boolean hasAge()
```

(2) [10 נקודות] ממשו את המתודה הסטטית loadUserData אשר מקבלת כפרמטר שם של קובץ csv ומחזירה מערך של Users אשר נוצרו על סמך הנתונים מהקובץ. גודל המערך אמור להיות כמספר המשתמשים במערכת. ניתן להניח כי מספר המשתמשים קטן מ 20,000 (השתמשו בקבוע MAX_USERS_IN_FILE) כמו כן, ניתן להניח שפורמט הקובץ הוא תקין, כלומר, מכיל שורות בפורמט המתואר בהמשך. ניתן להניח כי שם הקובץ שמועבר כפרמטר הוא תקין.

כל שורה בקובץ ה csv מייצגת משתמש יחיד והיא בעלת המבנה הבא:

```
userID;location;age
```

למשל, המשתמש הראשון שמופיע בקובץ מיוצר ע"י השורה:

```
"2";"stockton, california, usa";"18"
```

שימו לב:

א. השורה הראשונה בקובץ מכילה את כותרות העמודות ואינה מכילה נתוני משתמשים.

- ב. עבור חלק מהמשתמשים, השדה של age מכיל את הערך NULL, כלומר, הגיל של המשתמש אינו נתון. במקרה זה, יש להשתמש בבנאי שמקבל רק שני משתנים ובאמצעותו לאתחל את גיל המשתמש לערך ברירת המחדל.
- ג. המרכאות שמופיעות בכל הקבצים אינן חלק מהמחרוזת. כך לדוגמא, השדה location של האובייקט מסוג User שיבנה על פי השורה הראשונה לא יכיל את המרכאות. הנחיה זו תקפה גם לשאר הקבצים.

חתימת המתודה:

```
public static User[] loadUsersData(String fileName) throws Exception
```

- (3) [10 נקודות] באופן דומה, ממשו את המתודה הסטטית loadUsersData של המחלקה Book. מתודה זו קוראת קובץ csv אשר מכיל מידע על ספרים ומחזירה מערך. הניחו כי מספר הספרים במערך לא יעלה על 20,000 (MAX_BOOK_IN_FILE). כל ספר מיוצג ע"י שורה אחת בקובץ בפורמט הבא:

ISBN; Book-Title; Book-Author; Year-Of-Publication; Publisher

בנאי המחלקה, ה-getters שלה וכן המתודה toString אשר מחזירה את היצוג של האובייקט במחרוזת כבר ממומשים במחלקה.

חתימת המתודה:

```
public static Book[] loadBooksData(String fileName) throws Exception
```

- (4) [20 נקודות] כעת, נעבור למחלקה BookRecommendation.

ממשו את המתודה הסטטית loadRatings אשר מקבלת שלושה פרמטרים: שם קובץ ה-csv (fileName), מערך של משתמשים (usersArray) ומערך של ספרים (booksArray). הפונקציה מחזירה מערך דו מימדי אשר מכיל את הדירוגים של ספרים על ידי משתמשים כפי שמופיעים בקובץ ה-csv.

האיבר במקום ה-[i][j] במערך הדו מימדי, יהיה הדירוג של המשתמש usersArray[i] לספר booksArray[j].

אם מספר המשתמשים במערך הוא n ומספר הספרים הוא m – גודל המערך הדו מימדי שיתקבל יהיה nXm.

דירוג הספרים מיוצג בקובץ ה-csv באופן הבא: כל שורה מייצגת דירוג יחיד של משתמש לספר. פורמט השורות יהיה:

User-ID; ISBN; Book-Rating

כאשר USER-ID הוא המזהה החד ערכי של משתמש (השדה userID במחלקה User), ISBN הוא המזהה החד ערכי של ספר (השדה ISBN במחלקה Book), ו Book-Rating הוא מספר שלם בין 0 ל 10 (כולל שני הקצוות).

משתמש יחיד יכול לדרג מספר ספרים ולכן יופיעו מספר שורות עם אותו userID. באופן דומה, יותר ממשתמש אחד יכול לדרג כל ספר, ולכן אותו ISBN יכול להופיע בכמה שורות.

לא כל משתמש נתן דירוג לכל ספר, לכן, עליכם לאתחל כל תא שבו חסר דרוג בערך 1- (הערך 0 הוא דירוג חוקי כך שלא ניתן להשתמש בו). השתמשו בקבוע NO_RATING אשר מוגדר במחלקה.

חתימת המתודה:

```
public static int[][] loadRatingsData(String fileName, User[] usersArray, Book[] booksArray)
```

יתכן ותצטרכו להוסיף מתודות עזר על מנת להתאים בין User-ID לאינדקס שלו ברשימה usersList, וכן בין ה ISBN לבין האינדקס ב booksList.

(4) [5 נקודות] השלימו את הבנאי של המחלקה BookRecommendation כך שיאתחל את השדות הבאים:

שדה	משמעות
books [] Book	מערך הספרים במערכת
users [] User	מערך המשתמשים במערכת
ratings [][] int	טבלת דירוגים של משתמשים לספרים. האיבר ratings[i][j] מכיל את הדירוג של היוזר ה i ברשימת המשתמשים לספר ה j ברשימת הספרים.

חתימת הבנאי:

```
public BookRecommendations(Book[] books, User[] users, int[][] ratings)
```

(5) [10 נקודות] ממשו את המתודה getAverageRatingForUser אשר מקבלת אינדקס של משתמש ומחשבת את הדירוג הממוצע שלו עבור הספרים שקרא.

חתימת המתודה:

```
public double getAverageRatingForUser(int userIndex)
```

באופן דומה, ממשו את המתודה getAverageRatingForBook אשר מקבלת אינדקס של ספר ומחשבת את הדירוג הממוצע שלו בקרב כל המשתמשים שדירגו אותו. במידה ואף משתמש לא דירג ספר זה, היא תחזיר את הדירוג NO_RATING

חתימת המתודה:

```
public double getAverageRatingForBook(int bookIndex)
```

(6) [10 נקודות] נרצה לספק המלצה למשתמש על סמך קבוצת הגיל של המשתמש. לצורך כך, נבחן רק המלצות של משתמשים בקבוצת הגיל שלו.

תחילה נממש את המתודה getUsersInAgeGroup אשר מקבלת אובייקט מטיפוס User ומחזירה מערך של boolean-ים שגודלו כמספר המשתמשים במערכת. בתא ה i יופיע הערך true אם המשתמש שייך לקבוצת הגיל של המשתמש, ו false אחרת. קבוצת גיל של משתמש שגילו X מוגדרת בתור קבוצת כל המשתמשים שגילם בטווח [x-3, x+3], כולל הקצוות (השתמשו בקבוע AGE_GROUP_MARGIN_SIZE).

במתודות אשר מתייחסות לקבוצות גיל של משתמש, ניתן להניח שגילו של המשתמש מוגדר.

חתימת המתודה:

```
public boolean[] getUsersInAgeGroup(User user)
```

(7) [5 נקודות] כעת נממש את המתודה getAverageRatingForBookInAgeGroup אשר תחזיר את הדירוג של הממוצע של הספר רק בקרב קבוצת גיל מסויימת. במידה ולספר כלשהו לא קיים דירוג בקרב קבוצת גיל זו, החזירו את הדירוג NO_RATING (-1) בדומה לסעיף 6.

חתימת המתודה:

```
public double getAverageRatingForBookInAgeGroup(int bookIndex, boolean[]  
ageGroup)
```

(8) **[10 נקודות]** כעת, ממשו את המתודה `getHighestRatedBookInAgeGroup` אשר מקבלת כפרמטר משתמש (מטיפוס `User`), ומחזירה את הספר עם הדירוג הממוצע הגבוה ביותר בקרב קבוצת הגיל שלו. החזירו את הספר עם הדירוג הממוצע הכי גבוה, גם אם המשתמש קרא (ודירג) ספר זה. במידה ויש מספר ספרים עם הדירוג הכי גבוה, ביחרו שרירותית אחד מהם. ניתן ואף כדאי להשתמש במתודות שמימשותם בסעיפים הקודמים.

חתימת המתודה:

```
public Book getHighestRatedBookInAgeGroup(User user)
```

(9) **[10 נקודות]** לבסוף, נדפיס את ההמלצה לקובץ. ממשו את המתודה `printRecommendationToFile` אשר מקבלת משתמש (`User`) ומחזרת שהיא נתיב לקובץ ומדפיסה את ההמלצה בפורמט הבא:

```
The recommended Book for you is: 0140361219,Winnie-The-Pooh,A.A. Milne  
The book's average rating among its age group is: 9.75  
The book's average rating among all the users is: 7.80
```

(בסוף המשפט האחרון ניתן להוסיף ירידת שורה, לבחירתכם).

ניתן להניח שהפורמט של המחרוזת שמייצגת את שם הקובץ הוא חוקי, כלומר, המחרוזת אינה ריקה ומכילה שם חוקי לקובץ או נתיב תקין.

חתימת המתודה:

```
public void printRecommendationToFile(String fileName) throws Exception
```

השתמשו בשלוש המתודות המופיעות בסוף המחלקה בשביל לייצר את המחרוזות אותן צריך להדפיס.

טסטר

לתרגיל זה מצורפת מחלקת `טסטר`. על מנת להריץ את הטסטר עליהם לעדכן את מיקום שלושת קבצי ה-`csv` בהתאם למיקומם על המחשב שלכם. הטסטר לא יתקמפל עד שלא תממשו את המתודה `getAge` במחלקת `User`. בנוסף, עדכנו את שלושת הקבועים שמכילים את הנתיבים הנכונים לקבצים. הוסיפו בדיקות נוספות מלבד הבדיקות המופיעות בטסטר. לצורך הבדיקות החדשות כדאי לעבורך את קבצי הנתונים כך שתוכלו לבדוק את עצמכם בקלות (למשל – במקום לבדוק מה ממוצע הדירוגים של ספר כלשהו, ואז לבדוק לאילו קבוצות גיל שייכים המדרגים, הוסיפו נתונים משלכם – הוסיפו/עדכנו משתמשים לפי קבוצת הגיל המבוקשת, ספרים חדשים ואת הדירוגים שתרצו לבצע עליהם את החישובים).

בהצלחה!