תוכנה 1 – אביב תשע"ה

תרגיל מספר 8

collection framework -אוספים גנריים

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 aviv יכיל: zip יכיל:
 - א. קובץ פרטים אישיים בשם details.txt המכיל את שמכם ומספר ת.ז.
 - ב. קבצי ה- java של התוכניות אותם התבקשתם לממש.

חלק א' (50 נק')

בתרגיל זה עליכם לממש מבנה נתונים של היסטוגרמה באמצעות אוספים גנריים. נגדיר היסטוגרמה בתור מבנה נתונים אשר סופר מופעים של עצמים שונים. מבנה הנתונים הוא גנרי, כלומר, ניתן לקבוע מה יהיה טיפוס העצמים שהוא יחזיק.

לדוגמא, עבור אוסף האיברים הבא: 1, 2, 3, 1, 2, ההיסטוגרמה תכיל את האיברים 1, 2, 3 ואת מספר המופעים שלהם.

יחד עם קבצי התרגיל מסופק לכם הממשק Histogram המכיל שישה שירותים:

```
public interface IHistogram<T> extends Iterable<T> {
    public void addItem(T item);
    public void addItemKTimes(T item, int k) throws IllegalKValue;
    public void addAll(Collection<T> items);
    public int getCountForItem(T item);
    public void clear();
    public Set<T> getItemsSet();
}
```

- א. השירות addItem מוסיף את הפריט item פעם אחת להיסטוגרמה.
- ב. השירות addItemKTimes מוסיף ספירה של k עבור הפריט addItemKTimes ב. השירות וא הוא שלילי או 0, יזרק מוגדר בתוך הpackage המצורף לתרגיל).
- ג. השירות addAll מעדכן את ההיסטוגרמה עם האיברים ב items (אוסף כלשהו). מספר המופעים של כל איבר בהיסטוגרמה יעודכן לפי מספר המופעים שלו ב item (למשל, אם איבר item מופיע פעמיים ב item, הרי שיתווסף 2 לספירה של מופעי איבר זה).
 - שלא ראינו item יחזיר את מספר הפעמים שהאיבר getCountForItem ד. השירות כלל יוחזר הערך 9.
 - ה. השירות (כלומר, לאחר clear ירוקן את ההיסטוגרמה מכל האיברים והספירות (כלומר, לאחר clear, השירות (פובר). getCountForItem

ו. השירות getItemsSet יחזיר אוסף מטיפוס Set אשר מכיל את כל האיברים בהיסטוגרמה ללא השירות שלהם.

<u>סעיף 1 (25 נק'):</u>

ממשו את המחלקה HashMapHistogram אשר ממששת את המנשק HashMapHistogram. המימוש יעשה באמצעות הכלה (aggragation) של HashMap , כלומר, כל מופע של HashMapHistogram יכיל שדה מטיפוס HashMap. שדה זה יהיה אחראי על שמירת הספירות עבור כל אובייקט. טיפוס זה הוא גנרי, כלומר, ניתן לקבוע ביצירת האובייקט מהוא טיפוס האיברים אותם יחזיק.

(לב נק') 2סעיף

המנשק HashMapHistogram יורש מהמנשק, Iterable, מה שמחייב את HashMapHistogram לממש את השירות ().iterator

נרצה לעבור על תוכן ההיסטוגרמה באופן הבא: נעבור על כל האיברים, החל מהאיבר עם מספר המופעים הגדול ביותר ועד לאיבר עם מספר המופעים הקטן ביותר (עבור שני איברים עם אותו מספר המופעים אין חשיבות לסדר המעבר עליהם). אפשרות אחת היא להחזיק סוג של מבנה נתונים ממויין ולתחזק את הסדר שלו בכל הכנסת איבר, אך אנחנו מעוניינים במימוש שיאפשר הכנסת איברים מהירה ולכן לא נבצע שום מיון/סידור בעת הכנסת איברים. את המחיר בביצועים נשלם כאשר נרצה לעבור על כל האיברים באמצעות האיטרטור, ואז כן יתבצע סידור.

לצורך כך עליכם לממש:

- א. מחלקה חדשה המממשת את המנשק Iterator. ההיסטוגרמה שלנו ממומשת ע"י מיפוי (Map) מאיבר למספר המופעים שלו (ספירות), והאיטרטור צריך לעבור על האיברים בסדר יורד של הספירות.
 - אין צורך לתמוך בפעולת remove. מימוש דיפולטי של remove כבר מופיע בשלד המחלקה.
- ב. מחלקה חדשה המממשת Comparator. האיברים והספירות שלהם נמצאים במפה, כך שמיון ב. האיברים ע"פ מספר המופעים יבוצע למעשה ע"י מיון מפתחות המפה על פי הערכים בסדר יורד. לצורך כך נשתמש במיון (sort) ובאמצעות Comparator שישווה שני איברים ע"פ ערכם במפה.

שימו לב, ניתן ואף כדאי להעביר אוספים בין המופעים של המחלקות השונות וכן להשתמש בהכלה של אוספים לפי הצורך. למשל, על מנת להשוות בין הערכים של שני מפתחות במפה, ה Comparator יצטרך גישה למפה עצמה. ה iterator בתורו יצטרך לייצר את האוסף הממויין עליו יעבור במהלך האיטרציות.

שלד כל המחלקות אותן אתם נדרשים לממש נתון לכם בחבילה il.ac.tau.cs.sw1.ex8.histogram המופיעה בקבצי התרגיל.

העזרו ב HashMapHistogramTester בשביל לבדוק את עצמכם.

חלק ב' (50 נק')

בחלק זה נתרגל עבודה עם אוספים (Collections) ע"י מימוש מנוע פשוט להשוואה בין קבצי טקסט. בין קבצי העזר של התרגיל תוכלו למצוא את המחלקות FileSimilarityTest ו- FileUtils, ואת שלד המחלקה FileIndex (כולן בחבילה fileIndex). יש להגיש רק את (וווברוםחלקות עזר נוספות, אם כתבתם כאלה, בתוך תיקיות החבילה.

מנוע ההשוואה שלנו יקבל כקלט תיקיה במערכת הקבצים, יקרא את כל הקבצים בה, ו<u>ישמור ב"אינדקס"</u> עבור כל אחד מהקבצים את <u>מספר המופעים של כל מילה (token) בו</u>. לאחר מכן, נוכל להשוות בין הקבצים השמורים באינדקס.

<u>(20 נק'): סעיף 1</u>

המתודה ()index במחלקה FileIndex קוראת את הקבצים ומוסיפה אותם לאינדקס, והקוד שלה כבר נתונים index () בהן היא משתמשת: המתודה ()clearPreviousIndex מנקה נתונים נתון. השלימו את מתודות העזר בהן היא משתמשת: המתודה (addFileToIndex(File file מוסיפה את שנשמרו בקריאה הקודמת ל- ()index מחלקה. המחלקה. קריאת המילים מן הקובץ תתבצע בעזרת הנתונים הרלוונטיים לגבי קובץ טקסט יחיד לשדות המחלקה. קריאת המילים מן הקובץ תתבצע בעזרת readAllTokens(File file)

עבור קבצים שאינם מכילים מילים תקינות או שלא ניתן לקרוא מהם, תודפס הודעה על שגיאה והם <u>לא</u> יתווספו לאינדקס.

שימו לב, עליכם לבחור את מבני הנתונים המתאימים לייצוג המידע הדרוש (לשם כך, קראו גם את הסעיפים שימו לב, עליכם לבחור את מבני הנתונים מתוך Java collection framework. בפרט, <u>השתמשו</u> בבל שימוש יעיל באוספים גנריים מתוך HashMapHistogram אשר מומש בחלק א' על מנת לשמור את ספירות ה לובץ.

על מנת שלא לקשור בין המימוש הספציפי של IHistogram לבין המחלקה FileIndex, השתמשו במפעל במפעל אשר מומש עבורכם ומופיע ב package של חלק זה. שימוש פרקטי של במפעל – הוא יאפשר לנו לבדוק את הקוד של חלק ב' שלכם עם מימוש תקין על IHistogram, במידה ויש טעויות במימוש של חלק א', כל זאת מבלי לשנות את המימוש ל FileIndex.

(טעיף 2 (10 נק') 0

השלימו את מימוש המתודה getCommonTokensNum(File file1, File file2). מתודה זו מקבלת כקלט שני קבצים השמורים באינדקס ומחזירה את מספר המילים המשותפות לשני הקבצים. מילה תחשב משותפת לשני הקבצים אם היא מופיעה לפחות פעם אחת בשניהם.

: ('סעיף 3 (15 נק

המתודה getCosineSimilarity(File file1, File file2) המתודה (cosine similarity, את ציון הדמיון שלהם השמורים באינדקס, את ציון הדמיון שלהם

$$\frac{\sum_{w \in \mathsf{file1} \cap \mathsf{file2}} A_w \cdot B_w}{\sqrt{\sum_{w \in \mathsf{file1}} A_w^2} \cdot \sqrt{\sum_{w \in \mathsf{file2}} B_w^2}}$$

דוגמא לשימוש בנוסחא: נתונים לנו 2 קבצים עם ההיסטוגרמות הבאות:

מופעים	מילה
1	love
3	my
5	guitar

מופעים	מילה
4	while
2	my
3	guitar

הדמיון ביניהם יחושב באופן הבא:

$$\frac{3*2+3*5}{\sqrt{4^2+2^2+3^2}*\sqrt{1^2+3^2+5^2}}$$

במונה נסכום את מכפלות המופעים של האיברים המשותפים: המילים guitar ו my. מכיוון ש love ו while מכיוון ש while

הסבר לנוסחא: אנו מסתכלים על קבצי טקסט כעל וקטורים של מילים, ומחשבים את <u>קוסינוס הזווית</u> ביניהם כמדד לדמיון בין הקבצים. בהינתן מילה M_w , נסמן ב- M_w , את מספר המופעים שלה ב-file1 (כלומר, כמה פעמים היא חוזרת בתוצאת readAllTokens) וב- M_w , את מספר המופעים שלה ב-file2. במונה אנחנו מחשבים את מכפלת מספרי המופעים עבור כל מילה הנמצאת בשני הקבצים, וסוכמים. לכן, אם יש הרבה מילים משותפות שחוזרות על עצמן, המונה יהיה גדול. במכנה אנחנו מחלקים ב<u>נורמות</u> של כל אחד מהקבצים, כדי "לבטל" את השפעת גודל הקובץ ומספר החזרות של מילים בו.

בפרט, תוצאת הנוסחא היא 1 אם משווים קובץ מסוים לעצמו, ו- 0 אם משווים שני קבצים שאין ביניהם מילים משותפות כלל.

השלימו את מימוש המתודה. שימו לב כי לשם כך עליכם גם להשלים את מימוש מתודת העזר verifyFile(File file) אשר מקבלת כקלט קובץ ומחזירה true אם"ם זהו קובץ ששמור באינדקס. אחרת, עליה להדפיס הודעה למשתמש המתחילה ב- [ERROR] (בדומה להודעות השגיאה שמודפסות מהקוד הנתון לכם) ולהחזיר false. המתודה getCosineSimilarity משתמשת במתודה זו לבדיקה התחלתית של הקובץ.

(ל נק') 4 (סעיף

השלימו את מימוש המתודה () getNumIndexedFiles המחזירה את מס' הקבצים השמורים <u>כרגע</u> באינדקס.

<u>בדקו את עצמכם:</u> המחלקה FileSimilarityTest מכילה תכנית בדיקה קצרה עבור המחלקה שכתבתם. כדי להריץ אותה, שימרו את התיקיה testFiles תחת תיקיית הפרויקט ב- Eclipse או תחת התיקייה ממנה אתם מריצים את קוד הג'אווה. תיקיה זו מכילה 3 קבצים. בהרצה אמור להתקבל הפלט הבא (המסלול המלא לקבצים יכול להשתנות כתלות במקום בו שמרתם את התיקייה):

```
Indexing C:\testFiles\file1.txt
Indexing C:\testFiles\file2.txt
Indexing C:\testFiles\file3.txt
Indexed 3 files.
C:\testFiles\file2.txt: cosine similarity: 0.324, number of common words: 17
C:\testFiles\file3.txt: cosine similarity: 0.503, number of common words: 20
```

התוצאה, אגב, מראה כי לפי המדד שלנו, שיר של רוברט פרוסט (ב-file1) דומה יותר לשיר אחר של אותו משורר, מאשר לשיר של שייקספיר, כפי שהיינו מצפים.

מומלץ לשנות ולהוסיף קבצים לצרכי הבדיקה.

בהצלחה!