

# תוכנה 1 – אביב תשע"ו

## תרגיל מספר 4

### 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\_hw4.zip). קובץ ה-zip יכול:
  - א. קובץ פרטים אישיים בשם details.txt המכיל את שמכם ומספר ת.ז.
  - ב. 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.

### הקדמה: בדיקות קלט

- החל מתרגיל זה ואילך, יהא עליכם לבצע בדיקות של תקינות הקלט המתקבל לתכנית. למשל, האם מס' הארגומנטים תקין, האם ערכי הארגומנטים תקינים וכו'. עם זאת:
- אם מצוין בסעיף כלשהו כי ניתן להניח משהו על הקלט, אין צורך לבדוק זאת בקוד של אותו סעיף (הנחת קדם).
  - בפונקציות בהן מתקבל כקלט מסלול לקובץ, אין צורך לבדוק את חוקיות המסלול או אם הוא מצביע לקובץ קיים.
  - אין צורך לבדוק את מה שנאסף כבר ע"י הקומפיילר, למשל את טיפוס הארגומנטים למתודה.
- כדי להודיע למשתמש על מספר ארגומנטים לא תקין בשורת הפקודה נשתמש בפקודת throw באופן הבא:

```
throw new Exception("[ERROR] " + message);
```

כאשר המחרוזת [Error] תשמש אותנו להבדיל בין שגיאות שנזרקו ע"י הקוד שלכם לבין שגיאות שנזרקו ע"י מחלקה מהספרייה הסטנדרטית. המחרוזת message תכיל מסר קריא המסביר את הבעיה. השגיאה תיזרק ותגרום לסיים התכנית.

כדי להודיע למשתמש על קלט לא תקין מה-console, נדפיס הודעת אזהרה למשתמש באופן הבא:

```
System.out.println("[WARNING] " + message);
```

הודעה זו תסביר למשתמש כיצד עליו לתקן את הקלט בפעם הבאה.

### תוכנית לתיקון שגיאות בטקסט

בתרגיל זה נממש תוכנית אינטרקטיבית המתקנת שגיאות כתיב בטקסט, על סמך אוצר מילים בשפה – רשימת מילים חוקיות בשפה ורשימת טעויות נפוצות. שתי הרשימות יטענו מתוך קבצים.

שימו לב, בתוכנית זאת עליכם להשתמש **במערכים בלבד**, ולא במבני נתונים גנריים כמו Lists/Maps/Set וכו'. שימוש ב Arrays.asList גם כן אסור מכיוון שהוא מייצר אובייקט מטיפוס List.

בתרגיל זה נממש את המחלקה il.ac.tau.cs.sw1.ex4.SpellingCorrector (שלד המחלקה מופיע בקבצים המצורפים לתרגיל).

**חשוב:** בתרגיל זה תבצעו פעולות השוואה על מחרוזות. הקפידו להשתמש ב equals על מנת להשוות בין שתי מחרוזות, ולא ב ==.

(1) **[10 נק']** בשלב הראשון, נוסיף למחלקה שירות לטעינת אוצר המילים שלנו מקובץ ע"י מימוש מתודה לקריאת מילים באמצעות Scanner. בתוך המחלקה הגדירו את השירות:

```
public static String[] loadVocabulary(Scanner scanner)
```

השירות יקבל כקלט עצם מסוג java.util.Scanner ויניח שהוא כבר מאותחל לקריאה ממקור כלשהו (למשל, קובץ). עליכם לקרוא את המילים בעזרת ה- Scanner ולהחזיר מערך עם כל המילים שנקראו **ממוינות לקסיקוגרפית וללא חזרות**. לצורך המיון, העזרו בשירות Arrays.sort.

- גגדיר "מילה" כרצף של תווים שמופרד ע"י רווחים לבנים (whitespaces) מהרצפים האחרים. ניתן להניח שמפריד ברירת המחדל של Scanner הוא לפי רווחים לבנים.

- עליכם להחזיר רק מילים חוקיות. גגדיר מילה חוקית באופן הבא:

a. מילה אשר אורכה גדול מ 0 (כלומר, לא מחרוזת ריקה).

b. מילה המכילה רק אותיות חוקיות בשפה האנגלית (lowercase או upper). לדוגמא, המילים !abc , a-4 או ipv6 אינן מילים חוקיות ויש להתעלם מהן.

את סינון המילים הלא חוקיות ניתן לבצע בכל מני שיטות. שיטה מומלצת – שימוש בביטויים רגולריים. חומר קריאה מהתיעוד הרשמי של Java:

<https://docs.oracle.com/javase/7/docs/api/java/util/regex/Pattern.html>

ניתן כמובן להעזר גם בתוצאות חיפוש רגיל באינטרנט.

- את כל המילים יש להמיר ל lowercase.

- יש להחזיר רק את 2500 המילים ה"חוקיות" השונות הראשונות. בפרט, גודל המערך המוחזר לא יעלה על 2500.

- אם בסיום הקריאה מצאתם פחות מ- 2500 מילים שונות, גודל המערך המוחזר צריך להיות בהתאם. למשל, אם קראתם 2463 מילים, החזירו מערך בגודל 2463 שמכיל אותן.

- **אין לסגור את ה-Scanner בתוך המתודה.**

- ניתן להיעזר בשירותים של המחלקה java.util.Arrays (למעט ב Arrays.asList כפי שצויין)

(2) **[20 נק']** נוסיף למחלקה SpellingCorrector שירות אשר טוען את רשימת הטעויות הנפוצות. כמו בסעיף הקודם, השירות יקבל כקלט Scanner המאותחל לקריאה מקובץ הטעויות הנפוצות.

הטעויות הנפוצות מוגדרות ברמת כל אות. למשל, כאשר מקלידים את האות "a", לפעמים מתכוונים להקליד אחת מהאותיות "s", "i", "e", "o". כלומר, האותיות s, e, i, o מהוות תיקונים אפשריים עבור האות a.

קובץ הטעויות הנפוצות יכיל לכל היותר 26 שורות. האות הראשונה בכל שורה היא אות אותה נרצה לתקן, אחריה יופיע הסימן ":" ואחריו כל התיקונים האפשריים (לפחות תיקון אפשרי אחד). ניתן להניח שברשימת התיקונים לא תופיע אותה האות פעמיים וכן שלא יהיו שתי רשימות תיקונים עבור אותה האות.

עליכם לטעון את הקובץ לתוך מבנה הנתונים הבא: מערך שמכיל 26 מערכים מטיפוס char. המערך הראשון מכיל את כל התיקונים עבור האות "a", המערך השני מכיל את כל התיקונים עבור האות "b" וכן הלאה. במידה ולא סופקו תיקונים עבור אות מסויימת, המערך המייצג אותה יהיה מערך ריק (לא null).

לדוגמא: עבור קובץ התיקונים הבא:

a: e o i s

m: n

n: m

נייצר מערך דו מימדי בגודל 26.

התא באינדקס 0 יכיל מערך באורך 4 אשר מכיל את התוים: e, o, i, s

התא באינדקס 15 (מתאים ל m) יכיל מערך באורך 1 עם התו: n

התא באינדקס 14 (מתאים ל n) יכיל מערך באורך 1 עם התו: m.

שאר התאים יכילו מערכים ריקים.

סדר האותיות במערכים אמור להיות לפי סדר הקריאה מתוך הקובץ ואין למיין מערכים אלה (ראו דוגמא בתוכנית הבדיקה)

הוסיפו ל- SpellingCorrector את המתודה loadCorrections בעלת החתימה הבאה:

```
public static char[][] loadCorrections(Scanner scanner)
```

המתודה תקבל שתי מילים ותחזיר את המרחק ביניהן.

(3) [15 נק'] נוסף שירות, אשר עבור מילה מסויימת word, אוצר מילים מסויים, המיוצג ע"י המערך vocabulary, ורשימת תיקונים אפשריים המיוצגת ע"י המערך corrections מייצרת את רשימת כל התיקונים החוקיים ל word.

המילה word\_1 היא תיקון חוקי ל word אם:

1. word\_1 נוצרה ע"י החלפת אות יחידה ב word באחד התיקונים האפשריים שמוצעים ב corrections.

2. המילה word\_1 מופיעה ב vocabulary.

אורך המערך המוחזר יהיה כמספר התיקונים האפשריים, וניתן להניח שאורכו לא יעלה על 20. יש למיין את המערך על פי סדר לקסיקוגרפי עולה (בדומה למיין של vocabulary).

ממשו את המתודה getPossibleCorrections על פי החתימה הבאה:

```
public static String[] getPossibleCorrections(String word, String[] vocabulary, char[][] corrections)
```

(4) **[10 נק']** נוסף שירות המקבל משפט, חותך אותו למילים ע"פ הרווחים במשפט ומחזיר את רשימת המילים במשפט (לפי סדר הופעתן) בתוך מערך. במידה ויש יותר מרווח אחד בין שתי מילים, שימוש ב split ייצר מחרוזות ריקות, ובמקרה הזה עליכם להתעלם מהן ולא להחזיר אותן במערך, על מנת שלא לנסות ולתקן אותן. את המשפט יש להמיר ל lowercase. אורך המערך שיחזור יהיה כמספר המילים במשפט.

ממשו את המתודה המתודה splitSentence על פי החתימה הבאה:

```
public static String[] splitSentence(String sentence)
```

לדוגמא, אם נריץ את המתודה splitSentence על המשפט "love love me do" נקבל מערך באורך 4 עם המחרוזות הבאות: [love, love, me, do] עבור המשפט "love love me do" נקבל את אותו המערך, כיוון שאנחנו מתעלמים מרווחים.

(5) **[5 נקודות]** נוסף את השירות buildSentence אשר מקבל מערך של מילים, בונה מהן משפט לפי הסדר ומחזיר את המחרוזת המתקבלת. כל שתי מילים במשפט יופרדו על ידי רווח בודד. ניתן להניח שברשימת המילים words אין מילה ריקה. העזרו ב trim() במידה ואתם רוצים להוריד רווחים בסוף המילה שנוצרת.

```
public static String buildSentence(String[] words)
```

לדוגמא, עבור המערך [love, love, me, do] נקבל את המחרוזת "love love me do"

(6) **[5 נק']** נוסף את השירות isInVocabulary אשר מקבל מערך מחרוזות המייצגת את האוצר המילים וכן מחרוזת, ובודק אם המילה קיימת באוצר המילים. השירות יחזיר ערך בוליאני. ממשו את המתודה isInVocabulary על פי החתימה הבאה:

```
public static boolean isInVocabulary(String[] vocabulary, String word)
```

(7) **[35 נק']** כעת, נוסף למחלקה SpellingCorrector מתודת main המבצעת תיקון אוטומטי של משפטים הנקראים מה-console (כלומר, System.in). התכנית תקבל כארגומנטים משורת הפקודה את הנתיב לקובץ אוצר המילים ונתיב לקובץ התיקונים האפשריים. עם תחילת התוכנית היא תטען את אוצר המילים ורשימת התיקונים ע"י הפעלת השירותים loadVocabulary ו loadCorrections. לאחר טעינת הקבצים התוכנית תדפיס את השורה: "Loaded vocabulary from SSS", ואחרי את השורה "Loaded corrections from DDD" כאשר SSS ו DDD הם הנתיבים לקבצים כפי שהועברו בשורת הפקודה.

במידה וחסר ארגומנט או שאחד הקבצים אינו תקין, יש להדפיס הודעת שגיאה לבחירתכם (בפורמט המצוין בעמוד הראשון) ולצאת מהתוכנית.

לאחר מכן תתנהל אינטראקציה עם המשתמש (ראו מצגת תרגול 4 לגבי יצירת Scanner וחיבורו ל-System.in) באופן הבא:

- i. התוכנית תבקש מהמשתמש להזין משפט קלט (המשפט מסתיים בירידת שורה).
- ii. התוכנית תחלק את המשפט למילים ע"י שימוש ב splitSentence
- iii. התוכנית תבצע בדיקת איות לכל מילה ע"י שימוש בשירות findSimilarWords באופן הבא:
  - a. במידה והמילה חוקית, כלומר, קיימת ב vocabulary, התוכנית ממשיכה למילה הבאה.
  - b. במידה והמילה היא לא חוקית, התוכנית את ההודעה הבאה:  
The word (word) is incorrect  
כאשר (word) מייצג את המילה.
  - c. לאחר מכן, התוכנית תדפיס את כל ההצעות האפשריות לתיקון לפי סדר הופעתן ברשימת התיקונים האפשריים המתקבלת מהשירות getPossibleCorrections התיקונים ימוספרו בסדר עולה החל מ 1 (ראו דוגמת הרצה בסוף התרגיל).

לאחר הדפסת התיקונים תודפס השורה:

Enter your choice:

No possible corrections! תודפס ההודעה, תודפס הבאה (שלב f).  
והתוכנית תמשיך למילה הבאה (שלב f).

- d. בשלב זה המשתמש מתבקש להזין את בחירתו (מספר בין 1 למספר התיקון האחרון).
- e. ניתן להניח שהקלט שיתקבל מהמשתמש יהיה מספר שלם, אך עליכם לוודא שאכן נבחרה אופציה חוקית. במידה והמשתמש מזין מספר החורג מטווח התיקונים האפשריים, עליכם להדפיס את ההודעה "[WARNING] Invalid choice, try again!" (ראו דוגמת הרצה בסוף התרגיל).
- f. לאחר מכן, התוכנית תעבור למילה הבאה ותבצע את אותו התהליך.
- iv. לאחר סיום המעבר על כל המשפט, התוכנית תדפיס את המשפט המתוקן וכן את מספר המילים שתוקנו (ראו פלט ריצת התוכנית). השתמשו בשירות buildSentence בשביל לייצר את המשפט.
- v. לאחר מכן התוכנית תבקש מהמשתמש להזין משפט חדש. ניתן לסיים את ריצת התוכנית ע"י הקלדת הפקודה "quit" (לפני המילה quit מופיע מקף). זכרו לסגור בסוף התכנית את ה-Scanner – שפתחתם.

לנוחותכם, מצורף שלד למחלקה בו תוכלו להשלים את המימוש שלכם. השלד כולל מתודות אשר מבצעות את כל ההדפסות הנדרשות בתרגיל. מומלץ להשתמש בהן על מנת לוודא שאתם שומרים על הפורמט הנדרש, אך זה לא חובה.

### טסטר:

לתרגיל זה מצורפת מחלקת טסטר בשם SpellingCorrectorTest. מחלקה זו מיועדת לרוץ מאותו ה package של המחלקה SpellingCorrector אותה אתם מגישים. הריצו את הטסטר לאחר סיום המימוש - במידה וכל הבדיקות עברו, פלט הריצה של הטסטר יהיה "done!" בלבד, אחרת יודפס מספר השגיאה. אל תסתפקו בבדיקות שמופיעות בטסטר – הוסיפו בדיקות משלכם על מנת לוודא שהקוד עובד באופן תקין לכל קלט. אין צורך להגיש את הטסטר שלכם.

להלן אינטראקציה לדוגמה המדגימה את ההדפסות בכל שלב. שימו לב לנוסחים של ההודעות שהתוכנית מדפיסה. וודאו שהודעות אלה משמעותיות ומסבירות את הבעיה למשתמש. ניתן להניח ששני קבצי הקלט המופיעים בדוגמה נמצאים בתוך תקיית ההרצה. קלט מהמשתמש צבוע בכחול.

```
Loaded vocabulary from blackbird.txt
Loaded corrections from corrections.txt
Enter your sentence:
Blackbird singing im the dead af night
the word im is incorrect
1. am
2. in
Enter your choice:
2
the word af is incorrect
1. if
2. of
Enter your choice:
3
[WARNING] Invalid choice, try again!
2
The correct sentence is: blackbird singing in the dead of night
Enter your sentence:
I love Python
the word python is incorrect
No possible corrections!
The correct sentence is: i love python
Enter your sentence:
I love Java
The correct sentence is: i love java
Enter your sentence:
-quit
```

(כאשר אתם מריצים את התכנית שלכם לצרכי בדיקה, עליכם להעביר כארגומנטים כתובות של קבצים השמורים על המחשב שלכם).

**בהצלחה!**