

תוכנה 1 – סתיו תשע"ה

תרגיל מספר 4

משחקי מילים וקריאה מקבצים

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, כולל מבנה התיקיות של החבילה.

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

הקדמה: בדיקות קלט

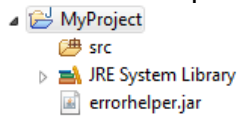
- החל מתרגיל זה ואילך, יהא עליכם לבצע בדיקות של תקינות הקלט המתקבל לתכנית. למשל, האם מס' הארגומנטים תקין, האם ערכי הארגומנטים תקינים וכו'. עם זאת:
- אם מצוין בסעיף כלשהו כי ניתן להניח משהו על הקלט, אין צורך לבדוק זאת בקוד של אותו סעיף (הנחת קדם).
 - בפונקציות בהן מתקבל כקלט מסלול לקובץ, אין צורך לבדוק 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 קלט לא תקין או על כך שהתקבלה שגיאה, נשתמש במחלקה `il.ac.tau.cs.sw1.util.ErrorHelper`. מחלקה פשוטה זו מאפשרת להדפיס שגיאות משני סוגים:
- שגיאות שלאחריהן התכנית ממשיכה בריצתה כרגיל. אלו שגיאות שניתן להתאושש מהן, למשל, בסיטואציה בה קיבלנו קלט לא תקין בקלט הסטנדרטי מן המשתמש, ויש באפשרותנו לבקש ממנו לתקן את הקלט. במקרה כזה נשתמש באחד מן השירותים של `ErrorHelper` באופן הבא:

```
// will print the message to the user
ErrorHelper.notifyRecoverable("Invalid input, please try again");
:א

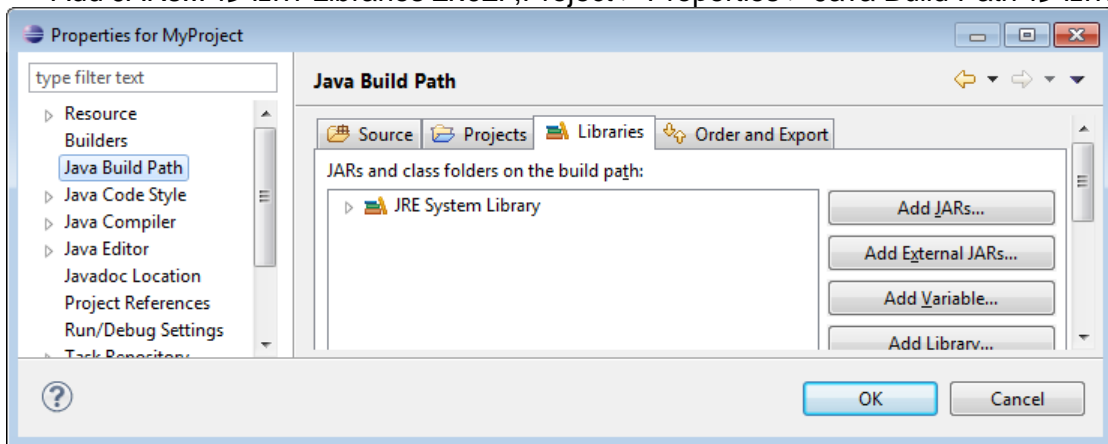
// will print the message only if the value of the first argument is
// false, i.e., input <=0
ErrorHelper.assertRecoverable(input > 0,
    " Invalid input, please try again ");
```

ב. שגיאות שלאחריהן נרצה שהתכנית תסתיים באופן מייד, למשל, קלט לא תקין משורת הפקודה.
 במקרה כזה נשתמש ב- `ErrorHelper.notifyUnrecoverable` או ב-
`ErrorHelper.assertUnrecoverable`, המקבילים לשתי המתודות הקודמות, אך במקום רק
 להדפיס שגיאה, גם גורמים לתעופה של התכנית.

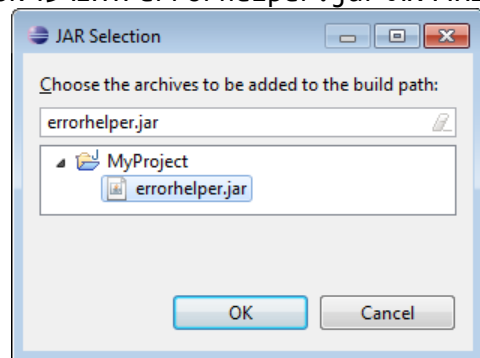
הוספת `ErrorHelper` לפרוייקט שלכם ב- Eclipse תתבצע באופן הבא:
 הורידו את `errorhelper.jar` מאתר הקורס ושמרו אותו בתוך הפרוייקט שלכם באקליפס.



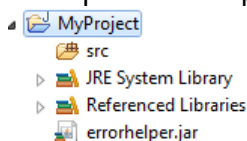
לחצו על `Project > Properties > Java Build Path`, ובטאו `Libraries` לחצו על `Add JARs...`



בחרו את `errorhelper.jar` ולחצו על `OK`.



לחצו שוב על `OK` לאישור סופי. כעת, `errorhelper.jar` יסומן בצלמית מיוחדת המעידה על כך שניתן
 להשתמש בו כספרייה חיצונית. כדי להשתמש ב- `ErrorHelper` יש להוסיף `import` מתאים בקוד.



תיעוד: באתר הקורס מופיע קישור המצביע לדפי התיעוד של `ErrorHelper`.

חלק א' – בניית אוצר מילים

כתבו את מחלקת העזר `il.ac.tau.cs.sw1.wordgame.Vocabulary` אשר תשמש לבניית אוצר מילים לצרכי המשחק שלנו.

(1) [20 נק'] נכתוב מתודה לקריאת מילים באמצעות `Scanner`. בתוך המחלקה הגדירו את השירות:
`public static String[] scanVocabulary(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 ללא חזרות.

- נגדיר "מילה" כרצף של תווים שמופד ע"י רווחים לבנים (whitespaces) מהרצפים האחרים. רשימת ה-whitespaces מופיעה כאן. כדי לגרום ל-Scanner להשתמש ברווחים לבנים כמפריד (delimiter) אפשר להשתמש במתודה `reset()`.
- עליכם להמיר תחילה את כל המילים שתקראו ל-`lowercase`.
- אם מילה שקראתם היא באורך 0, או מכילה תו שאינו אות בא"ב האנגלי (a-z), היא אינה "חוקית" ועליכם להתעלם ממנה. למשל, המילים "mp3" ו-"however" אינן חוקיות כי הן מכילות תו שאינו בתחום a-z.
- יש להחזיר רק את 5000 המילים ה"חוקיות" השונות הראשונות. בפרט, גודל המערך המוחזר לא יעלה על 5000.
- אם בסיום הקריאה מצאתם פחות מ-5000 מילים שונות, גודל המערך המוחזר צריך להיות בהתאם. למשל, אם קראתם 1177 מילים, החזירו מערך בגודל 1177 שמכיל אותן.
- אין לסגור את ה-Scanner בתוך המתודה.

(2) [20 נק'] הגדירו גם את השירות הבא

`public static String[][] getVocabularyByLetter(String[] vocabulary)`

שירות זה מקבל מילון שהוא פלט של `scanVocabulary`, ולשם יעילות עתידית בחיפוש, מחלק את המילים בו ל-26 מערכים לפי האות הראשונה של כל מילה. למשל, אם `vocabulary` הכיל את המילים {"apple", "apricot", "banana"} המתודה תחזיר את מערך מערכי המחרוזות הבא:

```
[ ["apple", "apricot"], ["banana"], [], [], ... [] ]
```

מכיוון שאין אף מילה שמתחילה באותיות z-c, אז בתא השלישי עד ה-26 יש מערכים בגודל 0 של מחרוזות (לא null). סדר המילים המתקבל בתוך כל מערך יכול להיות שונה מהדוגמא.

הנחיה:

- מכיוון שהקלט הוא פלט של `scanVocabulary`, אין צורך לחזור על בדיקות קלט ש-`scanVocabulary` מבצעת.
- ניתן לשמור עבור כל אות מערך בגודל 5000, לעבור על `vocabulary` ולהעתיק כל מילה למערך המתאים. בנוסף, עבור כל אות נשמור כמה מילים שמתחילות בה מצאנו. לסיום, מערכי המחרוזות שאינם מלאים עד הסוף יועתקו למערכים אחרים שגודלם הוא לפי מס' המילים בכל אות בעזרת `Arrays.copyOf`.
- לחילופין, פתרון מעט יותר מתוחכם ימין את פלט `scanVocabulary`, וייעזר במתודה `Arrays.binarySearch` כדי לחפש את נקודת ההתחלה והסיום של מילים המתחילות באות מסוימת בתוך `vocabulary` (רמז: חפשו, למשל, את המילה "a" והמילה "b"). את תתי המערכים ניתן להעתיק בעזרת `Arrays.copyOfRange`.

(3) [10 נק'] הוסיפו פונקציה main במחלקה Vocabulary, אשר מקבלת מסלול לקובץ בשורת הפקודה. התכנית:

- i. תבדוק שהתקבל ארגומנט יחיד, ואם לא, תשתמש ב-ErrorHandler לדווח על שגיאה ולצאת מהתכנית. אין צורך לבדוק שמדובר במסלול קובץ תקין.
- ii. תיצור Scanner שקורא מן הקובץ הזה.
- iii. תקרא ל- scanVocabulary שכתבתם כדי לחשב את מאגר המילים
- iv. תדפיס את מס' המילים השונות שנקראו (לכל היותר 5000) בפורמט
"Read %d unique words.%n"
- v. תקרא ל- getVocabularyByLetter על תוצאת המתודה scanVocabulary.
- vi. תדפיס את מערך המילים המתחילות ב-y בעזרת Arrays.toString
- vii. תסגור את ה-Scanner

ניתן לבדוק את עצמכם בעזרת הקובץ tale.txt המצורף לתרגיל באתר הקורס. בהינתן קובץ זה
קלט, התכנית אמורה להדפיס:

Read 2463 unique words.

[yard, year, years, yelled, yellow, yellowing, yet, you, young,
your, yours, yourself, yourselves]

(סדר המילים במערך יכול להיות שונה).

חלק ב' – משחק "מחברי המילים" (50%)

בחלק זה נפתח מחלקה שיודעת לשחק במשחק "מחברי המילים" על בסיס אוצר המילים. במשחק זה מקבל השחקן שלושה תווים מבין a-z, ומוצא מילה שמתחילה בתו הראשון, ומכילה גם את שני התווים האחרים בסדר כלשהו. אסור שתהיה חפיפה בין התווים במילה, כלומר, בהינתן התווים e e e, על השחקן למצוא מילה שמתחילה ב-e ומכילה שני מופעים נוספים של e. לכן, expected היא מילה חוקית, אך elephant (מכילה רק שני e) ו-seemed (לא מתחילה ב-e) אינן חוקיות עבור הקלט הנ"ל. המחלקה תיקרא il.ac.tau.cs.sw1.wordgame.PlayWordComposer

(4) [20 נק'] הוסיפו ל- PlayWordComposer את המתודה printWords בעלת החתימה הבאה:

```
public static void printWords(String[][] vocabularyByLetter,  
String firstLetter, String secondLetter, String thirdLetter)
```

המתודה מקבלת כקלט את פלט השירות getVocabularyByLetter ושלוש מחרוזות שכל אחת מהן מכילה תו a-z יחיד (ניתן להניח שהקלט תקין, אנו נבצע בדיקה זו במתודות אחרות בהמשך). המתודה מחפשת מילים מתאימות לשלושת התווים במערך הנתון, ומדפיסה כל אחת מהן בשורה נפרדת. לבסוף, היא מדפיסה את מספר המילים שנמצאו בפורמט "found %d words%n". לדוגמא, עבור אוצר המילים של הקובץ tale.txt הנתון והאותיות "e", "e", "y" יודפס

```
yelled  
yourselves  
found 2 words
```

עבור אותו אוצר מילים והאותיות "o", "o", "z" יודפס

```
found 0 words
```

(5) **[15 נק']** כעת, נוסיף למחלקה PlayWordComposer מתודת main. התכנית תקבל כקלט 4 ארגומנטים: את המסלול לקובץ קלט ושלושה תווים, תריץ את scanVocabulary ואת getVocabularyByLetter בכדי לקבל את אוצר המילים, ואז תריץ את printWords על התוצאה. יש לבדוק שהקלט תקין, כלומר:

- i. שישנם ארבעה ארגומנטים בדיוק
- ii. שהארגומנטים השני עד הרביעי הם אות a-z בודדת

אם הקלט אינו תקין, יש להודיע על שגיאה בעזרת ErrorHandler ולצאת מהתכנית. אם פתחתם Scanner, אל תשכחו לסגור אותו לאחר קריאת אוצר המילים.

(6) **[15 נק']** לבסוף, נוסיף למשחק שלנו אינטראקציה עם המשתמש דרך ה-console (כלומר, `System.in`). אם המשתמש מכניס 0 או 2 ארגומנטים ומעלה, התכנית תתנהג כפי שהוגדר בסעיף הקודם. אם המשתמש מכניס ארגומנט אחד, נניח שזהו מסלול לקובץ ונקרא ממנו את אוצר המילים. לאחר מכן תתנהל אינטראקציה עם המשתמש (ראו מצגת תרגול 4 לגבי יצירת Scanner וחיבורו ל-`System.in`):

- i. בכל פעם התכנית תבקש מהמשתמש להקיש 3 תווים מ-a-z ב-console, בשורה אחת, כאשר בין כל שני תווים מופיע רווח יחיד (ראו דוגמה למטה).
- ii. התכנית תוודא את נכונות הקלט.
- iii. אם הקלט אינו תקין, תודפס הודעה למשתמש בעזרת ErrorHandler והוא יוכל לנסות שוב.
- iv. אם הקלט תקין, התכנית תריץ עליו את printWords ושוב תאפשר למשתמש להקיש אותיות.
- v. התכנית תסתיים כאשר המשתמש יקיש "exit".

להלן אינטראקציה לדוגמא שגם מראה מה יש להדפיס בכל שלב. עם זאת, אתם רשאים לבחור בעצמכם את נוסח הודעות השגיאה שהתכנית שלכם תדפיס בעזרת ErrorHandler. וודאו שהודעות אלה משמעותיות ומסבירות את הבעיה למשתמש. קובץ הקלט בדוגמא למטה הוא tale.txt. קלט מהמשתמש מסומן בכחול.

```

Enter 3 letters or "exit"
b a a
balanced
breakfasted
found 2 words
Enter 3 letters or "exit"
b y
[ERROR #1] Expecting 3 letters
Enter 3 letters or "exit"
b u zz
[ERROR #2] Expected a letter but got: zz
Enter 3 letters or "exit"
exit

```

בהצלחה!