תוכנה 1 – חורף 2020/21

תרגיל מספר 4

<u>הנחיות כלליות:</u>

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

את התרגיל הבא ניתן להגיש באחת משתי הדרכים הבאות:

1. הגשה במערכת ה-moodle (http://moodle.tau.ac.il/) עפ"י ההנחיות הבאות:
   - יש להגיש קובץ **zip** יחיד הנושא את שם המשתמש ומספר התרגיל (לדוגמא, עבור המשתמש stav4 יקרא הקובץ stav4_hw4.zip). שימו לב שלא מדובר בשם שלכם, אלא במשתמש האוניברסיטאי, איתו אתם מתחברים למודל למשל. קובץ ה-zip יכיל:
   - קובץ פרטים אישיים בשם details.txt המכיל את שמכם ומספר ת.ז. (אפשר לכתוב בכל שפה – עברית או אנגלית. אתם לא נדרשים להיצמד לפורמט ספציפי, רק שהפרטים יהיו ברורים).
   - קבצי ה- java של התוכניות אותם התבקשתם לממש. יש לשים את תיקיית הsrc ובתוכה כל קבצי הג'אווה של סעיפי התרגיל כולל כל היררכיית התיקיות הפנימית של השלד והקבצים שסופקו בשלד. הקפידו ששמות הקבצים יהיו בדיוק כמו הקבצים שניתנו לכם בשלד. אין להוסיף קבצים או תיקיות נוספות! הקפידו שכאשר פותחים את הזיפ, מיד רואים את תיקיית הscr והיא לא נמצאת בתוך עוד תיקיה, למשל תיקיה ששמה זהה לשם ה-זיפ או הפרויקט (טעות נפוצה).
   - <u>שימו לב:</u> חשוב מאד להקפיד על פורמט ההגשה. כלומר אין להגיש קבצי rar, ויש לקרוא לקבצים בדיוק לפי ההנחיות שקיבלתם. כמו כן אין לצרף תיקיות או קבצים נוספים. **אי עמידה בהנחיות ההגשה תגרור הורדה משמעותית בניקוד!**

     נדגיש שוב: בזיפ יש רק קובץ details.txt, ותיקיית src שבתוכה כל קבצי הג'אווה שהתבקשתם לכתוב בתרגיל.

   - **הגשת מחלקה עם חבילות:** יש לכווץ בתוך קובץ ה-zip שאתם מגישים את כל היררכיית התיקיות מתחת ל-src באקליפס, כולל src עצמה.
2. הגשה ב-moodle וגם הגשה בgit.

   צרו את הrepository שלכם מתוך הקישור הבא:

   https://classroom.github.com/a/df3TcfgT

להנחיות להגשת git ניתן לעקוב אחר ההנחיות ממטלה 3.

התרגיל מנוסח בלשון נקבה ומיועד לסטודנטיות ולסטודנטים משני המינים.

ייבוא פרויקט:

יש לעקוב אחר ההנחיות מתרגיל 3. שימו לב כי בתיקיית ה-src שאתם מעתיקים יש גם תיקיית resources, בה נמצאים מקורות קלט לתוכנית שתוכלו להיעזר בהם.

הערות חשובות למימוש

1. שימו לב, בתוכנית זאת עליכם להשתמש **במערכים בלבד**, ולא בחומר מתקדם של מבני נתונים גנריים כמו Lists/Maps/Sets וכו'. שימוש במבני נתונים גנריים יכול לגרום להורדת ניקוד משמעותית עד כדי ציון נכשל. בפרט, אין להשתמש ב Arrays.asList שכן שירות זה מייצר אוסף גנרי מטיפוס List.
2. אין להשתמש בביטויים רגולרים (regex) למעט \s.
3. נא לא להשתמש אף פעם (גם במטלות עתידיות) במתודה exit.System כיוון שהיא משבשת את הבדיקות האוטומטיות.
4. מותר להוסיף מתודות עזר. ואין לשנות את ה-package שמופיע בשלד.
5. כל הקוד שלכם יכתב במחלקה il.ac.tau.cs.sw1.ex4.WordPuzzle, שלד המחלקה מצורף לתרגיל הבית. בשלד המחלקה מופיעות הגדרות קבועים ומימושים של פונקציות המייצרות את הפלט של התוכנית. השתמשו בהם בקוד שלכם, זה יעזור לכם לשמור על פורמט פלט תקין וימנע טעויות בהגדרות קבועים.
6. הנחות על הקלט לפונקציות:

   הניחו כי הקלט לכל הפונקציות חוקי, אלא אם כן צוין אחרת בגוף הפונקציה ויש התייחסות מפורשת לטיפול בערכים שאינם חוקיים. בפרט:

   - אם פונקציה מקבלת מילה (מחרוזת), המילה אינה **null** ואינה ריקה, ומכילה רק את התווים **a-z**.
   - אם פונקציה מקבלת חידה (מערך תווים), החידה אינה **null** ואינה מערך ריק.

משחק מילים אינטראקטיבי

בתרגיל זה נממש משחק אינטראקטיבי המורכב משני שלבים:

1. שלב ההגדרות שבו המשתמשת בוחרת חידה שהיא מילה שחלק מהאותיות שלה מוסתרות.
2. שלב המשחק שבו על המשתמשת לנחש את האותיות המוסתרות עד לחשיפת המילה בשלמותה.

המילים האפשריות לחידות יקראו מתוך קובץ שיכיל את אוצר המילים שלנו למשחק.

המטלה בנויה בצורה הבאה – ישנם 8 סעיפים אשר כל אחד מהם מורה על בניית של מתודה ספציפית. סעיף 9 והאחרון, הוא סעיף כתיבת ה**main** שיהווה את הבסיס למשחק, ולמעשה ישתמש בכל המימושים של סעיפים 1-8 במישרין או בעקיפין.

(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[] 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 **ממוינות לקסיקוגרפית** ו**ללא חזרות**.

- נגדיר "מילה" כרצף של תווים שמופרד ע"י רווחים לבנים (whitespaces) מהרצפים האחרים. ניתן להניח שמפריד ברירת המחדל של Scanner הוא לפי רווחים לבנים.
- מילה חוקית היא מילה שאינה ריקה, יש בה לפחות 2 אותיות, וכן, שמכילה אותיות בלבד. R2-D2 היא דוגמא למילה שאינה חוקית.
- את כל המילים יש להמיר ל `lowercase`.
- יש להחזיר רק את 3000 המילים ה"חוקיות" השונות הראשונות, כלומר גודל המערך המוחזר לא יעלה על 3000.
- אם בסיום הקריאה מצאתם פחות מ- 3000 מילים שונות, גודל המערך המוחזר צריך להיות בהתאם. למשל, אם קראתם 2463 מילים, החזירו מערך בגודל 2463 שמכיל אותן.
- אם אין מילים חוקיות כלל, יש להחזיר מערך ריק.
- **אין לסגור את ה-Scanner בתוך המתודה**.
- ניתן להיעזר בשירותים של המחלקה `java.util.Arrays`.

(2) **[5 נק']**

חידה הינה מילה שחלק מאותיותיה מוסתרות. חידה תיוצג בתוכנית ע"י מערך של תווים (char). התא ה `i` במערך יכול להכיל אחד משני ערכים: אות בין a ל z או את התו "_" (השתמשו בקבוע `HIDDEN_CHAR` אשר הוגדר עבורכם במחלקה). אם במקום ה `i` מופיע התו "_", פירושו שאות זו היא מוסתרת.

ממשו שירות אשר סופר את מספר האותיות המוסתרות בחידה puzzle כלשהי, כלומר, מספר הפעמים שהתו _ מופיע בחידה. הפונק' תחזיר 0 אם אף אות אינה מוסתרת בחידה.

חתימת השירות:

```
public static int countHiddenInPuzzle(char[] puzzle)
```

<u>יצירת מספרים רנדומליים עבור סעיפים 3 ו- 5</u>

במהלך ריצת התוכנית עליכם להגריל מספרים אקראיים. כדי לעשות זאת, השתמשו במחלקה **MyRandom** שניתנה לכם בשלד הקוד של התרגיל. מחלקה זו יורשת מ- Random שקיימת בספריית util של ג'אווה (על ירושה נלמד בהמשך הקורס, בשלב הזה אין צורך להבין מה משמעות המושג הזה). התוכנית אינה יוצרת מספרים אקראיים לגמרי, אלא שולפת מספרים מתוך מערכי מספרים שניתנים לה מראש כפי שיתואר בהמשך. אנו נתייחס למספרים הנשלפים האלה מהמערכים כמספרים אקראיים לצורך המטלה.

כדי להגריל מספרים רנדומליים (אקראיים) יש ליצור משתנה מחולל (generator) מסוג MyRandom באופן הבא:

```
Random generator = new MyRandom (intRands, floatRands);
```

כאשר intRands הוא מערך אשר מכיל מספרים שלמים וfloatRands הוא מערך המכיל מספרים עשרוניים בין 0.0 ל1.0 לא כולל. דוגמא ליצירת מחולל כזה קיימת בטסטר שסופק לכם.

לאחר יצירת המחולל, ניתן ליצור מספרים שלמים או עשרוניים חדשים באופן הבא:

```
int randomInt = generator.nextInt(n);

float randomDouble = generator.nextFloat();
```

השירות nextInt(n) אמור לייצר מספר שלם חדש בטווח 0 עד n-1.

השירות nextFloat() אמור לייצר מספר עשרוני חדש בטווח 0.0 עד 1.0 (לא כולל הקצוות)

בפועל, כל קריאה לאחד מהשירותים מחזירה מספר מהמערך הרלוונטי (intRands לnextInt וfloatRands לnextFloat, כך שהקריאה הבאה לאותו שירות תחזיר את המספר הבא במערך וכן הלאה. לאחר שהשירות תחזיר את המספר האחרון במערך, בקריאה הבאה לשירות יוחזר המספר הראשון במערך וכך הלאה. ניתן לקרוא לפקודות אלו ליצירת מספרים אקראיים ללא הגבלה.

בשימוש בMyRandom יש לקרוא לפונקציה nextInt(n) עם n שהוא הערך של האיבר הגדול ביותר במערך intRands בלבד ומיועד להיות גודל אוצר המילים (יוסבר בהקשר של מהלך המשחק). השימוש בצורה זו מאפשר תמיכה בMyRandom וRandom במקביל (על Random יוסבר בהמשך).

<u>הערות:</u>

1. יש לאתחל מחולל **אחד בלבד** למשך כל זמן ריצה יחידה של התוכנית, ואתחול זה יבוצע בפונקציית ה `main` שתממשו (סעיף 9).

2. על מנת לבדוק בנפרד את הפונקציות שמשתמשות במחולל, אתם כמובן יכולים לייצר מחוללים משלכם, בקוד של קובץ ה `tester` שלכם, בדומה ליצירה של המחולל בקובץ ה `tester` שלנו. לחילופין, על מנת ליצור את מערכי הקלט לבנייה של המחולל אתם יכולים להשתמש במתודות `getRandomFloatArr()` ו`getRrandomIntArr(int vocabularySize)` שמייצרות מערכים מעורבלים: מערך מעורבב בגודל 11 עם מספרים בין 0.0 ל1.0, ומערך מעורבב עם הערכים מ0 עד `vocabularySize` בהתאם.

3. בנוסף, אתם יכולים להשתמש במחולל מספרים אקראיים של ספריית ג'אווה כדי ליצור מספרים אקראיים באופן פסודו-אקראי אמיתי. כל מה שצריך לעשות זאת זה ליצור את המחולל עם השורה הבאה:

```
Random generator = new Random ();
```

הקריאות לשירותים `nextInt` ו`nextFloat` פועלות בצורה דומה לשירותים השירותים המקבילים ב`MyRandom`, רק שהמתודה `nextInt(n)` תחזיר באמת מספר אקראי בין 0 ל`n-1`/ אין חובה להשתמש במחולל זה, ואתם לא תיבדקו באמצעותו. בקבצי ה`java` שאתם מגישים יש להשתמש במחולל מסוג `MyRandom` בלבד. כל שימוש ב`Random` של ג'אווה יעשה בבדיקות העצמיות שלכם בלבד.

לסיכום אתם יכולים ליצור מחולל באחת מ3 דרכים, ברמת אקראיות הולכת וגדלה:

1. שימוש במחולל כפי שנוצר ב`Tester` (עם ערכים זהים או אחרים).
2. שימוש במחולל עם המתודות שמייצרות מערכים מעורבלים.
3. שימוש במחולל הרנדומלי של ג'אווה.

ב`main` נתונות לכם 3 שורות בהערה שמייצרות מחולל בהתאם לכל אחת מ3 הדרכים, לאחר שתסיימו את סעיף 9 בחרו את המחולל שמתאים לכם והוציאו אותו המערה.

הבדיקות יתבצעו באמצעות מחוללים דטרמיניסטיים קבועים מראש אך שלא ידועים לכם. אנו ממליצים לבדוק את הקוד שלכם עם המחולל שמוצע בדרך 1 כפי שמיוצר ב`Tester` באופן בסיסי ולבצע רק בדיקות מתקדמות עם מחוללים 1 ו2.

(3) **[5 נק']**

ממשו את השירות getRandomWord אשר מקבלת את אוצר המילים בצורת מערך של מחרוזות, ומחולל מספרים אקראיים. השירות תגריל מספר שלם i באמצעות המחולל ותחזיר את המילה מאוצר המילים שנמצאת באינדקס ה-i במערך.

חתימת השירות:

```
public static String getRandomWord(String[] vocabulary, Random generator)
```

(4) **[5 נק']**

חידה בעלת מבנה חוקי היא חידה המקיימת את התנאים הבאים:

א. בחידה יש לפחות אות אחת מוסתרת ולפחות אות אחת גלויה.

ב. אם אות מסוימת מופיעה יותר מפעם אחת במילה כלשהי, כל המופעים של אות זו יהיו מוסתרים מוסתרים בחידה, או שכל המופעים שלה יהיו גלויים.

ממשו את השירות checkLegal אשר מקבל מילה כלשהי וחידה שנוצרה על סמך המילה הזו, ובודק אם החידה היא חוקית. ניתן להניח שהאורכים של החידה והמילה שווים, ובכל מיקום בחידה יש את האות המקבילה מהמילה או את תו ההסתרה '_'.

```
public static boolean checkLegal(String word, char[] puzzle)
```

(5) **[5 נק']**

ממשו את השירות getRandomPuzzleCandidate אשר מקבלת ערך הסתברותי prob שמכיל מספר עשרוני בטווח בין 0.0 ל1.0 ומילה כלשהי. השירות יבנה חידה על פי המילה וההסתברות שהועברו. ניתן להניח כי הקלט תקין. בניה של חידה על פי מילה נתונה תיעשה ע"י החלפה חלק מהאותיות במילה לתו '_' בהסתברות שנתונה בprob.

ההחלפה תתבצע במעבר על כל האותיות במילה מהאות הראשונה עד האות האחרונה. עבור כל אות יש להגריל מספר float אחד בלבד באמצעות הפונקציה nextFloat() של המחולל. אם המספר שהוגרל הוא קטן שווה להסתברות prob, האות הזו תהיה מוסתרת בחידה.

שימו לב – שירות זה יכול לייצר חידות שאינן חוקיות על פי מה שהוגדר בשאלה 3. פונקציה זו תשמש כפונקציית עזר עבור בניית חידה חוקית בסעיף הבא.

לדוגמא, עבור המילה wheeps והערך 0.8 אנו נרצה להחזיר חידה כך שלכל אות יש הסתברות 0.8 להיות '_' בחידה. כלומר, 80% מהאותיות במילה wheeps אמורות להיות מוסתרות, בממוצע.

הפלטים הבאים הם פלטים תקינים (וישנם עוד המון פלטים תקינים):

```
wh___s, _h___s, w__eps, whe_ps,w_____
```

שימו לב שעבור prob=0.8, פחות סביר לקבל את החידה whe_ps מאשר את החידה w_____ (תחת ההנחה שהמספרים המוגרלים מתפלגים אחיד בין 0 ל 1), אבל שתי התוצאות הן אפשריות.

חתימת השירות:

```
public static char[] getRandomPuzzleCandidate(String word, double prob,
Random generator)
```

(6) **[5 נק']**

בסעיף זה נייצר חידה חוקית ע"י שימוש בשירותים שמומשו בסעיפים 4 ו 5.

על מנת ליצר חידה חוקית, אנחנו צריכים לנסות לייצר חידה באמצעות השירות שמומש בסעיף 5 ולבדוק אם החידה היא חוקית באמצעות השירות שמומש בסעיף 4. במידה והחידה היא חוקית, הפונק' תחזיר את החידה שהתקבלה, אחרת הפונקציה תנסה ליצר חידה חדשה מאותה מילה. אחרי 1000 ניסיונות על מילה יש לזרוק שגיאת זמן ריצה באמצעות השירות throwPuzzleGenerationException אשר מומש עבורכם.

(נלמד לעומק על שגיאות זמן ריצה בהמשך הסמסטר).

חתימת השירות:

```
public static char[] getRandomPuzzle(String word, double prob, Random
generator)
```

(7) **[10 נק']** נוסיף שירות אשר מקבל תו (guess), מילה (solution) וחידה (puzzle).
במידה וישנם מופעים מוסתרים של התו guess בחידה puzzle, נחשוף את המופעים הללו ע"י עדכון המקומות המתאימים puzzle מהתו '_' לתו guess. הפונקציה תחזיר את מספר המקומות שעודכנו.
למשל, עבור החידה [w, _, _, _, p], הפתרון wheep, והתו e, הפונקציה תעדכן את החידה ל [w, _, e, e, p] ותחזיר את הערך 2. אם עבור אותן חידה ומילה היינו מעבירים את התו f, האובייקט puzzle לא היה משתנה והפונקציה הייתה מחזירה את הערך 0.

ממשו את המתודה applyGuess על פי החתימה הבאה:

```
public static int applyGuess(char guess, String solution, char[] puzzle)
```

הנחות:

- guess יהיה תמיד אות חוקית באי"ב האנגלי ב lowercase.

- המילה `solution` מהווה פתרון חוקי לחידה `puzzle`
- `Puzzle` היא חידה בעלת מבנה חוקי (ראו הגדרה בסעיף 3)

(8) **[10 נק']** נוסיף שירות אשר מקבל מילה (`solution`) וחידה (`puzzle`) ומחזיר את החידה עם אות אחת מוסתרת פחות.
השירות יעבור על החידה, ימצא את המיקום של התו '_' הראשון שמופיע במחרוזת (בעל האינדקס הנמוך ביותר) ואז יחזיר את החידה אחרי שהוא חשף בה את כל המופעים של האות שנמצאת במיקום הזה, לפי מילת הפיתרון. ניתן לערוך את המערך המקורי ולהחזיר אותו, או ליצור מערך חדש להחזיר אותו, לבחירתכם.

למשל, עבור החידה `[w, _, _, _, p]` והפתרון `wheep`, האות לחשיפה היא `h` והפונקציה תחזיר את החידה:
`[w, h, _, _, p]`.

עבור החידה `[w, h, _, _, p]` והפתרון `wheep`, האות לחשיפה היא `e` והפונקציה תחזיר את החידה:
`[w, h, e, e, p]`.

ממשו את המתודה `getHelp` על פי החתימה הבאה:

```
public static char[] getHelp(String solution, char[] puzzle)
```

(9) **[45 נק']** כעת, נוסיף למחלקה `WordPuzzle` מתודת `main` שתפעיל את המשחק האינטראקטיבי באמצעות קבלת קלט מה`console` (כלומר, `System.in`). התכנית תקבל כארגומנט משורת הפקודה את המסלול לקובץ אוצר המילים, תפעיל את השירות `scanVocabulary` בכדי לקבל את אוצר המילים, ותדפיס את מס' המילים שנקראו בפורמט: `"Read DDD words from SSS"`, כאשר `DDD` הוא מספר המילים באוצר המילים שנוצר ו `SSS` הוא נתיב הקובץ בדיוק כפי שהועבר ברשימת הארגומנטים. במידה וחסר ארגומנט יש להדפיס הודעת שגיאה לבחירתכם ולצאת מהתוכנית בצורה מסודרת (ללא שימוש ב`system.exit()`) מבלי להדפיס דבר. הניחו כי אם מספר הארגומנטים תקין, שם הקובץ שהועבר הוא קובץ שאכן קיים בנתיב שהועבר, ואינו ריק.

לאחר מכן האינטראקציה עם המשתמשת תתנהל באופן הבא (ראו מצגת תרגול 4 או מדריך הIO לגבי יצירת `Scanner` וחיבורו ל- `System.in`):

A. שלב ההגדרות:

a. התוכנית תדפיס את השורה `--- Settings stage ---`

b. לאחר מכן, המשתמשת תתבקש להזין את ההסתברות להסתרת אותיות. התוכנית תדפיס את ההודעה `Enter your hiding probability:` ותמתין לקלט מהמשתמשת.

i. המחרוזת הראשונה שהמשתמשת מזינה הינה ההסתברות להסתרה. ניתן להניח שהמספר הוא מספר עשרוני תקין בין 0 ל1 ושניתן לשכן אותו במשתנה מטיפוס `float`.

ii. התוכנית תגריל מילה לפי סעיף (3) עם אוצר המילים, תיצור חידה חוקית כמפורט בסעיף (6) ותדפיס את החידה שנבחרה.

iii. התוכנית תדפיס בשורה נפרדת את ההודעה: `Replace puzzle?`

1. במידה והקלט מהמשתמשת יהיה `yes` על התוכנית יהיה לחזור לשלב `ii` (הגרלת חידה)
2. במידה והקלט מהמשתמשת יהיה `no` המשתמשת תועבר ישירות לשלב המשחק.
3. עבור כל קלט אחר, התוכנית תחזור על שלב `iii`.

B. שלב המשחק:

a. התוכנית תדפיס את השורה: `--- Game stage ---`

b. כל משחק מתחיל עם מספר ניסיונות השווה למספר התווים החסרים בחידה + 3. כלומר, אם בחידה מוסתרים 4 מקומות, למשתמשת יהיו סה"כ 7 ניסיונות.

c. התוכנית מדפיסה את החידה, ולאחר מכן, בשורה חדשה, את ההודעה: `Enter your guess:` וממתינה לקלט מהמשתמשת. ניתן להניח שהמשתמשת תזין תו אחד ואחריו תקיש על Enter.

d. התוכנית בודקת אם האות שהוזנה מופיעה כאות מוסתרת במילה.

i. אם אות זו מופיעה כאות מוסתרת במילה:

1. התוכנית בודקת אם כל החידה פוענחה (כל האותיות גלויות). אם כן, תודפס ההודעה

```
Congratulations! You solved the puzzle
```

והתוכנית תסתיים.

2. אחרת, מספר הניחושים יורד ב-1, התוכנית מדפיסה את ההודעה הבאה (XXX - מס' הניחושים שנותרו) ועוברת לשלב e

```
Correct Guess, xxx guesses left
```

ii. אחרת, (האות לא מופיעה או שהיא מופיעה וגלויה), התוכנית בודקת האם האות היא האות המיוחדת 'H'.

1. אם האות היא לא 'H' מספר הניחושים יורד ב-1 ומודפסת ההודעה:

```
Wrong Guess, xxx guesses left
```

והתוכנית עוברת לשלב e.

1. אם האות היא 'H' התוכנית תציע עזרה למשתמשת ע"י שינוי החידה כך שהיא תהיה עם אות אחת נוספת גלויה תוך שימוש במתודה getHelp. שימו לב שהחידה עצמה משתנה והאות נחשפת (ללא הדפסה בשלב זה). מספר הניחושים שנותרו למשתמשת יורד ב-1. לאחר מכן התוכנית בודקת אם כל החידה פוענחה (כל האותיות גלויות). אם כן, תודפס ההודעה

```
Congratulations! You solved the puzzle
```

והתוכנית תסתיים. אחרת, התוכנית חוזרת לשלב e עם האות הנוספת שנגלתה.

e. במידה ונשארו עוד ניחושים (כל ניחוש מוריד 1 ממספר הניסיונות), התוכנית חוזרת לשלב c. אחרת, מודפסת ההודעה `Game over!` והתוכנית מסתיימת.

לנוחותכם, מצורף שלד המחלקה 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 אין זה חובה.

הנחות נוספות:

1. בשלב המשחק, ניתן להניח שהקלט מהמשתמשת הוא חוקי, כלומר, בכל תור המשתמשת תזין אות אנגלית אחת ב lowercase.

טסטר:

לתרגיל זה מצורפת מחלקת טסטר בשם `WordPuzzleTester`. מחלקה זו מיועדת לרוץ מאותו ה package של המחלקה `WordPuzzle` אותה אתם מגישים. הריצו את הטסטר לאחר סיום המימוש, במידה וכל הבדיקות עברו, פלט הריצה של הטסטר יהיה `"done!"` בלבד, אחרת יודפס מספר השגיאה.

הטסטר משתמש גם לבדיקת נכונות החתימות של המתודות שתממשו (אם המחלקה לא מתקמפלת, חסרה מתודה או שחתימת אחת המתודות לא נכונה.) וגם לבדיקה שטחית של נכונות המימוש. **אל תסתפקו בבדיקות שמופיעות בטסטר. הוסיפו בדיקות משלכם על מנת לוודא שהקוד עובד באופן תקין לכל קלט.** הטסטרים לא יבדקו כך שאין צורך (אבל זה גם לא יפריע) להגיש אותם. ניתן להגיש את התיקייה `resources` וניתן גם לא להגיש אותה. הדברים שחשוב שישארו בהגשה הוא מבנה התיקיות, וקבצי `java` שהם לא ה`tester`.

להלן אינטראקציה לדוגמא המדגימה את ההדפסות בכל שלב. שימו לב לנוסחים של ההודעות שהתוכנית מדפיסה.

קובץ הקלט בדוגמא למטה הוא `vocabulary.txt` וניתן להניח שבהרצה זו הקובץ נמצא בתיקיה `resources/hw4`. קלט מהמשתמש מסומן בכחול.

בתיקיית קבצי התרגיל תוכלו למצוא אוצר מילים נוסף. שימו לב שריצה זו התבצעה עם מחולל שיש לו ערכים לא ידועים.

```
Read 15 words from src/resources/hw4/vocabulary.txt
--- Settings stage ---
Enter your hiding probability:
0.6
sw_ep___
Replace puzzle?
yes
__ite
Replace puzzle?
no
--- Game stage ---
__ite
Enter your guess:
a
Wrong Guess, 4 guesses left
__ite
Enter your guess:
H
w_ite
Enter your guess:
b
Wrong Guess, 2 guesses left
w_ite
Enter your guess:
h
Congratulations! You solved the puzzle
```

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

בהצלחה!