

תוכנה 1 – חורף 2021/22

תרגיל מספר 4

הנחיות כלליות:

- קראו בעיון את קובץ נהלי הגשת התרגילים אשר נמצא באתר הקורס. את התרגיל הבא צריך להגיש באופן הבא:
 - הגשה במערכת ה-Git תבצע על פי ההנחיות שראיתם בתרגול 0. צרו את ה repository שלכם מתוך הקישור הבא:
<https://classroom.github.com/a/edPAOsBK>
 - יש לוודא שבתיקיית הגיט שלכם נמצאים הקבצים הבאים:
 - a. קובץ פרטים אישיים בשם details.txt המכיל את שם המשתמש שלכם ב Moodle ואת מספר תעודת הזהות שלכם.
 - b. קבצי ה- java של התוכניות אותם התבקשתם לממש. בתרגיל הנוכחי ישנם 2 קבצים: WordPuzzle.java ו-WordPuzzleTester.java.
- הגשה במערכת ה Moodle (<http://moodle.tau.ac.il/>): עליכם להגיש את קובץ הטקסט assignment.txt ובו קישור ל git repository שלכם.

הערות חשובות למימוש

- 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.
- מלבד ה Scanner שמוגדר בפונקציה main, **אין לייצר Scanner נוסף שיחובר ל System.in**.
- נא לא להשתמש אף פעם (גם במטלות עתידיות) בשירות System.exit כיוון שהיא משבשת את הבדיקות האוטומטיות.
- מותר להוסיף מתודות עזר. ואין לשנות את ה-package שמופיע בשלד.
- כל הקוד שלכם יכתב במחלקה WordPuzzle.ex4.sw1.il.ac.tau, 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` (כלומר אותיות קטנות).
- אם פונקציה מקבלת חידה (מערך תווים), החידה אינה `null` ואינה מערך ריק.

7. הניחו כי הקלט שהמשתמשת מזינה לתוכנית הוא חוקי, אלא אם צוין אחרת בגוף השאלה ויש התייחסות מפורשת לטיפול בקלטים לא חוקיים.

משחק מילים אינטראקטיבי

בתרגיל זה נממש משחק אינטראקטיבי המורכב משני שלבים:

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

המטלה בנויה בצורה הבאה: 5 הפונקציות הראשונות הן פונקציות כלליות שכל אחת מהן מבצעת מטלה שונה הקשורה לבניית ופתרון חידות. שתי הפונקציות האחרונות מגדירות את התוכנית עצמה – המשחק האינטראקטיבי.

הגדרות:

חידה (puzzle) הינה מילה שחלק מאותיותיה מוסתרות. חידה תיוצג בתוכנית ע"י מערך של תווים (char). התא `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 `i` מופיע התו `"_"`, פירושו שתו זה הוא מוסתר, ויש לנחש אותו.

עבור מילה מסויימת, בניית החידה תיעשה על ידי שילוב של המילה המקורית ושל **תבנית** (template). תבנית היא מערך של בוליאנים (boolean). אם בתא `i` של התבנית מופיע `true`, זה אומר שהתו `i` מוסתר בחידה שתיבנה מתוך תבנית זו. אם במקום `i` מופיע `false`, זה אומר שהתו `i` יהיה גלוי. למשל, עבור המילה `while` והתבנית `[false, false, true, false, true]` נקבל את החידה:

`[w,h,_,l,_]`

(1) [5 נק'] ממשו את השירות:

```
public static char[] createPuzzleFromTemplate(String word, boolean[] template)
```

הפונקציה מקבלת מילה ותבנית, ומחזירה מערך המייצג חידה. ניתן להניח שאורך התבנית זהה לאורך המילה, שהחידה היא חידה חוקית (חוקיות תוגדר בסעיף הבא).

(2) [10 נק'] נגדיר כי חידה היא בעלת מבנה חוקי אם היא מקיימת את התנאים הבאים:

א. בחידה יש לפחות תו אחד מוסתר ולפחות תו אחד גלוי.

ב. אם אות מסוימת מופיעה יותר מפעם אחת במילה המקורית, כל המופעים של אות זו יהיו מוסתרים מוסתרים בחידה, או שכל המופעים שלה יהיו גלויים (למשל, עבור המילה keep, החידה k, _, e, p אינה חוקית, שכן האות e גם מוסתרת וגם חשופה).

ממשו את השירות checkLegal אשר מקבל מילה כלשהי ותבנית, ובודק אם התבנית מייצרת חידה חוקית עבור המילה הנתונה. עליכם לבדוק את אורכי החידה והתבנית (אם האורכים לא זהים, לא ניתן לייצר חידה בעלת מבנה חוקי מהתבנית והמילה הנתונות).

```
public static boolean checkLegalTemplate(String word, boolean[] template)
```

(3) [15 נק'] כעת נרצה למצוא את כל התבניות שיכולות לייצר חידה חוקית עבור מילה מסוימת. עבור

מילה word כלשהי, ומספר שלם k כלשהו, נרצה להחזיר את כל התבניות שמהן ניתן לבנות חידות חוקיות עבור word ובהן מוסתרים בדיוק k תוים (אם קיימת אות שמופיעה פעמיים, היא נספרת פעמיים. אנחנו סופרים את מספר התוים המוסתרים, לא את מספר האותיות שונות).

תחילה, נבין מהו מספר התבניות המקסימלי האפשרי. עבור מילה באורך n קיימות 2^n תבניות אפשריות.

אם נרצה לספור את כל התבניות עם בדיוק k תוים מוסתרים, הרי שקיימות לכל היותר $\binom{n}{k}$ תבניות כאלה. (כלומר, מתוך n תוים, נבחר את ה k שיוסותרו). מדובר בחסם עליון - לא כל התבניות האפשריות מייצרות חידות חוקיות. למשל, עבור המילה look, אם נרצה להסתיר תו אחד, יש רק 2 אופציות - הסתרת האות l והסתרת האות k.

ממשו את הפונקציה getAllLegalTemplates אשר מחזירה מערך אשר מכיל את כל התבניות

החוקיות עבור מילה word ומספר שלם k. ניתן להניח $0 < k < word.length()$.

על מנת לסדר את התבניות במערך המוחזר, נמפה כל תבנית למספר בייצוג בינארי ע"י המרה של כל true ב 1 ושל false ב 0. אם המספר שמוצג ע"י t1 קטן יותר מהמספר המיוצג ע"י t2, הרי שהתבנית t1 תופיע לפני התבנית t2 במערך שיחזור.

לדוגמא, עבור look ו k=1, שתי התבניות האפשריות הן:

```
t1 = [true, false, false, false] ו t2 = [false, false, false, true]
```

(תבניות נוספות לא אפשריות עבור k=1 כיוון שלא ניתן להסתיר רק אחד מהמופעים של האות o)

אם נמיר את הערכים הבוליאנים נקבל t1=1000, ו t2=0001. מכיוון ש t2 מייצגת מספר קטן יותר,

נקבל מערך עם 2 איברים. האיבר הראשון יהיה t2 והשני יהיה t1.

מכיוון שניתן לייצר כמות גדולה מאוד של תבניות, נגביל את הקלטים בחלק זה למילים שאורכן לא גדול מ

10. השתדלו להיות יעילים מבחינת כמות חישובים והקצאות הזכרון. שימו לב שניתן לבנות את המערך

על פי הסדר הנדרש, מבלי להידרש לסדר אותו לאחר יצירתו.

חתימת השירות:

```
public static boolean[][] getAllLegalTemplates(String word, int k)
```

הערה: ניתן, אבל לא חובה, להשתמש בפונקציה Integer.toString אשר מקבלת מספר

שלם וממירה אותו למחרוזת המכילה את הייצוג הבינארי שלו.

(4) **10 נק'** נוסף שירות אשר מקבל תו (guess), מילה (word) וחידה (puzzle).

במידה שהאות guess מוסתרת בחידה puzzle, נחשוף את כל המופעים המוסתרים שלה ע"י החלפת

כל אותם המיקומים מהתו '_' לאות guess בתוך puzzle. הפונקציה תחזיר את מספר המקומות

שעודכנו.

למשל, עבור החידה [s, p, _, _, w], הפתרון wheeps, והניחוש e, הפונקציה תעדכן את החידה

ל [s, p, e, e, w] ותחזיר את הערך 2. אם עבור אותן חידה ומילה היינו מעבירים את האות m,

האובייקט puzzle לא היה משתנה והפונקציה הייתה מחזירה את הערך 0.

ממשו את המתודה המתודה applyGuess על פי החתימה הבאה:

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

הנחות:

- guess יהיה תמיד אות חוקית בא"ב האנגלי ב lowercase.
- החידה puzzle היא חידה שנבנה באופן חוקי מהמילה word.

(5) [15 נק'] נוסף שירות אשר מקבל מילה (word) וחידה (puzzle) ומערך של המייצג את הניחושים שכבר בוצעו (already_guessed) ומחזירה רמז: 2 אותיות. אחת מהאותיות תהיה אות "נכונה" – אות אשר מוסתרת בחידה. האות השניה תהיה אות "לא נכונה" – לא מופיעה במילה וגם עדין לא נחשה ע"י המשתמש. המערך already_guessed מכיל 26 תאים, והתא ה- i מכיל true אם המשתמש כבר ניסח את התו ה- i (בין אם הניחוש היה נכון, או לא).
התווים יחזרו בתוך מערך של char-ים בגודל 2 ויסודרו על פי סדר הא"ב (בסדר עולה). עליכם לבחור בצורה אקראית את האות ה"נכונה" ואת האות ה"לא נכונה".
אתם לא נדרשים להקפיד על כך שכל תו יבחר אקראית בדיוק באותה ההסתברות, אך המימוש שלכם צריך לאפשר לכל תו (נכון ולא נכון) להיבחר.
ניתן להניח שיש לפחות אחת המותרת בחידה, ושקיימת לפחות אחת שלא מופיעה במילה וגם לא נחשה עדיין.
למשל, עבור החידה [p, _ , _ , _ , w], והפתרון wheep, הפעלת הפונקציה getHint יכולה להחזיר את המערך הבא (האות השניה היא האות הנכונה): ['h', 'a']

העזרו במחלקה Random על מנת להגריל מספרים אקראיים.
טיפ לשימוש ב Random: כאשר מייצרים את האובייקט Random ניתן להעביר לו מספר שלם כלשהו (seed). הפרמטר הזה, שאינו חובה, יכול "לקבע" את ההנהגות הרנדומליות כך שבכל ריצה, הערכים הרנדומליים שיוגרו יהיו זהים (כלומר, אם בריצה הראשונה הוגרל המספר 2 ואחריו המספר 10, זה יקרה בכל ריצה). השימוש ב seed מאוד מקובל ומאפשר שחזורים של ריצות ומעקב אחרי תוצאות, ויכול להיות שימוש עבורכם.

הערה – אין לשנות את המערך already_guessed אשר מועבר כפרמטר לפונקציה. ממשו את המתודה המתודה getHint על פי החתימה הבאה:

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

כעת נממש את התוכנית שתריץ את המשחק עצמו.

התוכנית היא תוכנית אינטרקטיבית, ומכילה שני שלבים: שלב ההגדרות ושלב המשחק. כל אחד משלבים אלה ימומש בפונקציה נפרדת. החלק האינטרקטיבי ימומש ע"י שימוש בפרמטר מטיפוס Scanner שיועבר לכל אחת מהפונקציות. ה Scanner יקרא את הקלט של המשתמש מה console. (כלומר, - [System.in](#) – הפרטים מופיעים במצגת תרגול 4 או במדריך ה IO שמיועד ללימוד עצמי).
המימוש של פונקציית ה main (התוכנית עצמה) נתון לכם, ואין לשנות אותו. התוכנית קולטת מהמשתמש מילה אחת שאורכה לא עולה על 10 תווים. מילה זו מועברת כפרמטר לתוכנית. עבור מילה זו, מופעל שלב ההגדרות, ולאחר מכן שלב המשחק.

(6) [25 נק'] שלב ההגדרות (בתוך הפונקציה mainTemplateSettings) :
הפונקציה מקבלת שני פרמטרים: המילה (word) וה Scanner איתו קולטים את הקלט מהמשתמש
(inputScanner), ופועלת באופן הבא:

- a. הפונקציה תדפיס את השורה --- Settings stage ---
- b. המשתמש תתבקש לבחור תבנית ליצירת החידה.
 - i. הפונקציה תדפיס את המחרוזת: Choose a (1) random or (2) manual template:
 - ii. המשתמש תקליד את הספרה 1 או 2.
 - iii. אם המשתמש בוחרת ב 1:
 - 1. הפונקציה תדפיס: Enter number of hidden characters:
 - 2. לאחר מכן, המשתמש תזין את מספר התוים שיוסותרו.
 - 3. במידה וניתן לייצר חידה עבור המילה + מספר התוים שיוסותרו, הפונקציה תגריל תבנית אקראית מתוך התבניות האפשריות ותחזיר אותה (מכאן, התוכנית תמשיך לשלב המשחק).
 - 4. במידה שזה לא אפשרי, הפונקציה תדפיס את ההודעה Cannot generate puzzle, try again.
(למשל, עבור המילה noon לא ניתן לייצר חידה עם תו מוסתר אחד).
לאחרת הדפסת ההודעה הזו, הפונקציה תחזור לשלב b.
 - iv. אם המשתמש בוחרת ב 2:
 - 1. הפונקציה תדפיס: Enter your puzzle template:
 - 2. המשתמש תזין את התבנית המבוקשת באופן הבא. עבור כל תו גלוי, היא תשתמש ב X (אות גדולה). עבור כל תו מוסתר היא תשתמש ב _ . בין כל שני תוים יפריד פסיק. דוגמא לקלט אפשרי: __,X,X,_. בתבנית זו יש שני תוים גלויים (תו שלישי ורביעי) ושלושה תוים מוסתרים.
 - 3. הפונקציה תבדוק אם התבנית מגדירה חידה חוקית עבור המילה שנבחרה. אם ניתן לייצר חידה חוקית, הפונקציה תחזיר חידה זו (והתוכנית תמשיך לשלב המשחק).
 - 4. אם לא, התוכנית תדפיס את ההודעה Cannot generate puzzle, try again.
לאחר הדפסת ההודעה הזו, התוכנית תחזור לשלב b.

חתימת הפונקציה:

```
public static char[] mainTemplateSettings(String word, Scanner inputScanner)
```

(7) [25 נק'] שלב המשחק (בתוך הפונקציה mainGame):

הפונקציה מקבלת שלושה פרמטרים: המילה (word), החידה (puzzle) וה Scanner איתו קולטים את הקלט מהמשתמש (inputScanner), ופועלת באופן הבא:

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

b. כל משחק מתחיל עם מספר ניסיונות השווה למספר התווים החסרים בחידה + 3. כלומר, אם בחידה מוסתרים 4 תווים, למשתמש יהיו סה"כ 7 ניסיונות. שימו לב שאנחנו סופרים תווים מוסתרים ולא אותיות (שונות) מוסתרות. בחידה n, _, n, שהפתרון שלה הוא nsoon מוסתרים 2 תווים, אבל למעשה צריך לנחש רק אות אחת.

c. התוכנית מדפיסה את החידה, ולאחר מכן, בשורה חדשה, את ההודעה: `Enter your guess:` וממתינה לקלט מהמשתמש.

d. המשתמש מזינה אות יחידה. יש שתי אפשרויות שהמשתמש יכולה להזין – אות קטנה מ a-z (הניחוש), ובקשת רמז: האות H (אות גדולה).

i. אם המשתמש הזינה את הקלט H, התוכנית מדפיסה רמז – שני תווים, שאחד מהם מהווה ניחוש נכון ואחד מהם לא. שני התווים שהתוכנית מציעה הם תווים שהמשתמש עדין לא ניחשה, ושאינם מופיעים כחשופים בחידה (על פי ההנחיות למימוש `getHint`). שימו לב שבכל שלב של התוכנית ניתן יהיה לייצר רמז – מכיוון שאורך החידה הוא לכל היותר 10, ואנחנו מורשים לעשות לכל היותר 3+9 ניחושים, עד לסוף המשחק יהיו מספיק תווים בשביל לייצר רמזים כנדרש.

הדפסת הרמז תיעשה בפורמט הבא:

`Here's a hint for you: choose either XXX or YYY.`

כאשר XXX ו YYY הן שתי האותיות שהמערכת מציעה כרמז, ו XXX מופיעה לפני YYY בא"ב.

לאחר הדפסת הרמז, התוכנית חוזרת לשלב c. מכיוון שהמשתמש לא ביצעה שום ניחוש, מספר הניחושים שנשארו לה לא משתנה.

ii. אחרת, התו שהוזן הוא ניחוש שהמשתמש ביצעה. הפונקציה שומרת את הניחוש הנוכחי (בשלב ה"רמז" אנחנו מציעים למשתמש רק תווים שהיא עדין לא ניחשה, ולכן נדרשים לשמור את כל הניחושים שנעשו במשחק). לאחר מכן, הפונקציה בודקת אם האות שהמשתמש ניחשה אכן מוסתרת בחידה.

1. אם אות זו מופיעה כאות מוסתרת בחידה:

(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) `Correct Guess, xxx guesses left.`

לאחר הדפסת ההודעה, אם למשתמשת נשארו ניחושים נוספים, הפונקציה חוזרת לשלב c. אם לא נשארו ניחושים, התוכנית עוברת לשלב e.

2. אם הניחוש לא נכון, (האות לא מופיעה או שהיא מופיעה וגלויה), מספר הניחושים יורד ב-1 ומודפסת ההודעה:

`Wrong Guess, xxx guesses left.`

לאחר הדפסת ההודעה, אם למשתמשת נשארו ניחושים נוספים, הפונקציה חוזרת לשלב c. אם לא נשארו ניחושים, התוכנית עוברת לשלב e.

e. במקרה שהמשתמשת ניצלה את כל הניחושים שלה ולא הצליחה לפתור את החידה, מודפסת ההודעה `Game over!` וריצת הפונקציה מסתיימת, מה שלמעשה מסיים גם את התוכנית.

הנחיות כלליות:

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.

תוכנית בדיקות:

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. להלן אינטראקציה לדוגמא המדגימה את ההדפסות בכל שלב. שימו לב לנוסחים של ההודעות שהתוכנית מדפיסה. התוכנית קיבלה כקלט את המילה wheeps. שימו לב שבגלל השימוש בערכים רנדומליים, הריצה שלכם לא חייבת להיות זהה לריצה המתוארת כאן. הדוגמא ניתנת על מנת להבהיר את אופן פעולת התוכנית (הקלט מהמשתמש צבוע בכחול)

```
--- Settings stage ---
Choose a (1) random or (2) manual template:
1
Enter number of hidden characters:
3
--- Game stage ---
wh__s
Enter your guess:
t
Wrong Guess, 5 guesses left.
wh__s
Enter your guess:
e
Correct Guess, 4 guesses left.
whee_s
Enter your guess:
H
Here's a hint for you: choose either f or p.
whee_s
Enter your guess:
p
Congratulations! You solved the puzzle!
```

בהצלחה!