

# תוכנה 1 – אביב 2019/20

## 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 aviv4 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 ה-src ובתוכה כל קבצי הג'אווה של סעיפים התרגיל. 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 ה-src והיא לא נמצאת בתוך עוד תיקיה, למשל תיקיה ששמה זהה לשם ה-zip או הפרויקט (טעות נפוצה).
- שימו לב: חשוב מאד להקפיד על פורמט ההגשה. כלומר אין להגיש קבצי far, 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.

<p><b>הגשת מחלקה עם חבילות:</b> יש לכווץ בתוך קובץ ה-zip שאתם מגישים את כל היררכיית התיקיות מתחת ל-src באקליפס, כולל src עצמה.</p>
--

- מותר להוסיף מתודות עזר. ואין לשנות את ה-package שמופיע בשלד.
- נא לא להשתמש בפקודה System.exit()! היא מחבלת בבדיקות אוטומטיות. אין כל צורך לעשות בה שימוש, כאשר תוכניות יכולות להסתיים ע"י הגעה לסוף מתודת ה-main או בפקודות .return.

### יבוא 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.

## משחק מילים אינטרקטיבי

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.

שימו לב, בתוכנית זאת עליכם להשתמש **במערכים בלבד**, ולא במבני 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`. כל הקוד שלכם יכתב במחלקה `il.ac.tau.cs.sw1.ex4.WordPuzzle`, שלד המחלקה מצורף לתרגיל הבית.

הערה: 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.

### 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.

## המחלקה Scanner - מדריך ללימוד עצמי

בתרגיל בית זה אתם תעשו שימוש במחלקה `Scanner` על מנת לקרוא את תוכנם של קבצי טקסט (שאלה 1) ולקרוא קלט שהמשתמשת מזינה דרך המקלדת (שאלה 9). לשם כך, לפני שתתחילו לממש את המתודות, עיינו במדריך ללימוד עצמי של פעולות קלט/פלט המצורף לתרגיל זה. עליכם לקרוא את כל המדריך, שכן הוא מכיל דיונים חשובים מעבר לשימוש ישיר במחלקה `Scanner`. עם זאת, ניתן בינתיים לא להתעמק בחלקים הדנים בזרמים, בהם תשתמשו בתרגיל הבית הבא (תרגיל בית 5).

(1) [10 נק'] בשלב הראשון, נוסיף למחלקה שירות לטעינת אוצר המילים שלנו מקובץ ע"י מימוש מתודה לקריאת מילים באמצעות `Scanner`. בתוך המחלקה ממשו 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 **ממיונות לקסיקוגרפית ולא 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. שימו לב, כי מדובר ב-3000 המילים הראשונות מבחינת סדר ההופעה שלהן בקובץ, ולא לפי סדר המיון.
- 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.
- 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)
```

## יצירת מספרים רנדומליים עבור סעיפים 3 ו-5

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). 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 ו-`floatRands` הוא מערך המכיל מספרים עשרוניים בין 0.0 ל 1.0 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)` מייצרת מספר שלם חדש בטווח 0 עד `n-1`.

השירות `nextFloat()` מייצרת מספר עשרוני חדש בטווח 0.0 עד 1.0 (לא כולל הקצוות)

ניתן לקרוא לפקודות אלו ליצירת מספרים אקראיים ללא הגבלה.

כל קריאה לאחד מהשירותים מחזירה מספר מהמערך הרלוונטי (`nextInt` ל-`intRands` ו-`nextFloat` ל-`floatRands`), כך שהקריאה הבאה לאותו שירות תחזיר את המספר הבא במערך וכן הלאה. לאחר שהשירות יחזיר את המספר האחרון במערך, בקריאה הבאה לשירות יוחזר המספר הראשון במערך וכן הלאה.

## הערות:

1. יש לאתחל מחולל אחד בלבד למשך כל זמן ריצה יחידה של התוכנית, ואתחול זה יבוצע בפונקציית ה main שתממשו (סעיף 9). על מנת לבדוק בנפרד את הפונקציות שמתמשות במחולל, אתם כמובן יכולים לייצר מחוללים משלכם, בקוד של קובץ ה tester שלכם, בדומה ליצירה של המחולל בקובץ ה tester שלנו.

2. אתם יכולים ליצור את המחולל עם מערכי מספרים כרצונכם או להשתמש בדוגמא שיש בטסטר. בנוסף, אתם יכולים להשתמש במחולל מספרים אקראיים של ספריית ג'אווה כדי ליצור מספרים אקראיים באופן פסאודו-אקראי אמיתי. כל מה שצריך לעשות זה ליצור את המחולל עם השורה הבאה:

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

הקריאות לשירותים nextFloat() ו nextInt() פועלות בצורה דומה לשירותים השירותים המקבילים במחולל MyRandom. 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.

### (3) [5 נק']

ממשו את השירות getRandomWord אשר מקבלת את אוצר המילים בצורת מערך של מחרוזות, ומחולל מספרים אקראיים. השירות יגריל מספר שלם i באמצעות המחולל ויחזיר 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:

- בחידה יש לפחות אות אחת מוסתרת ולפחות אות אחת גלויה.
- אם אות מסוימת מופיעה יותר מפעם אחת במילה כלשהי, כל המופעים של אות זו יהיו מוסתרים בחידה, או שכל המופעים שלה יהיו גלויים.

ממשו את השירות checkLegal אשר מקבל מילה כלשהי וחידה שנוצרה על סמך המילה הזו, ובדוק האם 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 ל 1.0 (לא כולל 0 ו 1 עצמם) 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.

החלפה תבצע 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 או 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 לייצר 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'] נוסף 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 ב-puzzle מהתו '\_' לתו guess. הפונקציה תחזיר את מספר המקומות שעודכנו.

למשל, עבור החידה [p, \_\_, \_\_, \_\_, w], הפתרון wheep, והתו e, הפונקציה תעדכן את החידה ל [p, \_\_, e, e, w] 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 המתודה applyGuess על פי החתימה הבאה:

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

הנחות:

- guess יהיה תמיד אות חוקית בא"ב האנגלי ב lowercase.
- המילה solution מהווה פתרון חוקי לחידה puzzle
- Puzzle היא חידה בעלת מבנה חוקי (ראו הגדרה בסעיף 3)

(8) [10 נק'] נוסף שירות אשר מקבל מילה (solution) וחיידה (puzzle) ומחזיר את החידה עם אות אחת מוסתרת פחות (ניתן להניח שהחידה שהתקבלה היא חוקית ויש בה לפחות אות מוסתרת אחת). 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.

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

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

ממשו את המתודה getHelp על פי החתימה הבאה:

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

(9) [45 נק'] כעת, נוסף למחלקה WordPuzzle מתודת main שתפעיל את המשחק האינטרקטיבי באמצעות קבלת קלט מהconsole (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 שם הקובץ כפי שהועבר ברשימת הארגומנטים. במידה וחסר ארגומנט יש להדפיס הודעת שגיאה לבחירתכם ולצאת מהתוכנית. הניחו כי אם מספר הארגומנטים תקין, שם הקובץ שהועבר הוא קובץ שאכן קיים בנתיב שהועבר, ואינו ריק.

שימו לב, שבשלב כבר אותחל בתוך המתודה גנרטור שהוא מופע של MyRandom עם מערכים ספציפיים (אלא אותם המערכים כמו בטסטר שקיבלתם). **על מנת לאפשר בדיקה דטרמיניסטית של התרגיל שלכם, אנא הקפידו לא לשנות את המערכים איתם הגנרטור מאותחל בסעיף זה!**

לאחר מכן האינטראקציה עם המשתמשת תתנהל באופן הבא (ראו מדריך קלט/פלט ללימוד עצמי 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 ל 1 ושניתן לשכן אותו במשתנה מטיפוס float.  
ii. התוכנית תגדיל 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 מהמשתמש יהיה yes על התוכנית יהיה לחזור לשלב ii (הגרלת חידה)
  2. במידה והקלט מהמשתמש יהיה no המשתמש 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 נסיונות השווה למספר התווים החסרים בחידה + 3. כלומר, אם בחידה מוסתרים 4 מקומות, למשתמש יהיו בסה"כ 7 נסיונות.
  - c. התוכנית מדפיסה את החידה, ולאחר מכן, בשורה חדשה, את ההודעה: `Enter your guess:` וניתן להניח שהמשתמש 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, 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' מודפסת ההודעה: `Wrong Guess, xxx guesses left` והתוכנית עוברת לשלב e.
      1. אם האות היא 'H' התוכנית תציע עזרה למשתמש ע"י שינוי החידה כך שהיא תהיה עם אות אחת נוספת גלויה תוך שימוש במתודה `getHelp`. שימו לב שהחידה עצמה משתנה והאות נחשפת. מספר הניחושים שנותרו למשתמש 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 הנסיונות), 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. כמו כן, אל תשכחו לסגור את ה-Scanner בתום השימוש. הנחות נוספות:
1. בשלב המשחק, ניתן להניח שהקלט מהמשתמש הוא חוקי, כלומר, בכל תור המשתמש 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.

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 מהמשתמשת מסומן בכחול. בתיקיית קבצי התרגיל תוכלו למצוא אוצר מילים נוסף.

```
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!**