

# תוכנה 1 – אביב 2020/2021

## תרגיל מספר 10

### GUI, enum, BufferedWriter

#### הנחיות כלליות:

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

- את התרגיל יש להגיש באופן הבא: הגשה במערכת ה Moodle (<http://moodle.tau.ac.il/>): עליכם להגיש את קובץ הטקסט assignment.txt ובו קישור ל git repository האישי שלכם.
- הגשה במערכת ה-Git תבצע על פי ההנחיות שראיתם בתרגול 0. צרו את ה repository שלכם מתוך הקישור הבא:

<https://classroom.github.com/a/cJsystf>

יש לוודא שבזמן ההגשה בתיקיית הגיט שלכם נמצאים הקבצים הבאים:

א. קובץ פרטים אישיים בשם details.txt המכיל את שמכם ומספר ת.ז.

ב. שתי התיקיות src | resources. יש להגיש את ה repository ואת התיקיות בדיוק באותה ההיררכיה שבה

הם נוצרו.

## חלק א': חידות ג'אוה (20%)

בשאלה זו עליכם קוד ג'אוה בחבילה enumRiddles, שבה שתי מחלקות TLightTest, DayTest:

עליכם להשלים את המחלקות מבלי לשנות את פונקציות ה-main כלל, ומבלי להוסיף קבצים אחרים, כך ש:

- כל הקוד יעבור קומפילציה ללא שגיאות וללא אזהרות.
- אם נריץ את התכנית, הן תמיד תסיים את הריצה ללא שגיאות וידפוס:

TLightTest:

RED: 30 seconds, next is GREEN

AMBER: 10 seconds, next is RED

GREEN: 30 seconds, next is AMBER

DayTest:

MONDAY (1), next is TUESDAY

TUESDAY (2), next is WEDNESDAY

WEDNESDAY (3), next is THURSDAY

THURSDAY (4), next is FRIDAY

FRIDAY (5), next is SATURDAY

SATURDAY (6), next is SUNDAY

SUNDAY (7), next is MONDAY

הערות:

- TLightTest תדפיס 3 שורות בלבד ו DayTetsi תדפיס 7 שורות בלבד, ללא ירידות שורה מיותרות בסוף.
- אין מגבלה על הקוד של המחלקות, כל עוד הוא קוד ג'אוה תקין, והוא מממש את enum. הדבר היחידי שאסור לשנות אלו מתודות ה main של המחלקות.
- ניתן וצריך (אך לא חובה) להשתמש במתודות אבסטרקטיות ולממש באופן שונה עבור כל ערך.

## חלק ב' (30% נק')

בשאלה זו עליכם לממש גירסא בסיסית של המחלקה BufferedReader עליה למדתם עצמאית בתרגילי בית 4 ו 5 (כדאי לחזור על ההסברים של המוטיבציה לעבודה עם buffer-ים). הקוד ימומש בחבילה `il.ac.tau.cs.sw1.ex10.bufferedIO`.

שימו לב, אתם נדרשים לממש מחלקה דומה (אך לא זהה בהתנהגותה) ל `BufferedReader`, כך שאינ לעשות שימוש ב `BufferedReader` במימוש שלכם.

כזכור, העקרון המנחה של מחלקה זו הוא שהיא עוטפת זרמים אחרים (לרוב `FileReader`) ודרכם קוראת מספר קבוע של תוים לתוך מערך, באופן שקוף למשתמש (ה `BufferedWriter` עובד באופן דומה).

המחלקה `MyBufferedReader`:

מחלקה זו מממשת את הממשק `IBufferedReader` (מוגדר עבורכם בחבילה `bufferedIO`).

בנאי המחלקה: מקבל אובייקט מטיפוס `FileReader` וכן מספר שלם – גודל ה `buffer`. להזכירכם, ה `FileReader` מאפשר קריאה של תוים מהקובץ לתוך מערך של תוים.

הפונק' `getNextLine` מחזירה את השורה הבאה בקובץ אליו מחובר ה `FileReader`. את סוף השורה נזהה על פי תו ירידת השורה `\n`. תו זה אינו חלק מהמחרוזת המוחזרת, בדיוק כמו ב `BufferedReader` הרגיל. בכל גישה ל `FileReader` עלינו לקרוא מספר תוים על פי גודל ה `buffer` שהוגדר לנו מראש (בקריאה האחרונה יתכן שנקרא פחות תוים, במקרה שהקובץ מכיל מספר תוים שאינו מתחלק בגודל ה `buffer`).

הפונק' תחזיר `null` אם כל תוכן הקובץ כבר נקרא (בדיוק כפי שפועל `BufferedReader`).

על מנת להחזיר שורה שלמה, יתכן ונרצה לקרוא מ `FileReader` יותר מפעם אחת בריצה אחת של `getNextLine`. למשל, אם ה `Buffer` גודל הוא 5 תוים והשורה אותה נרצה לקרוא מכילה 18 תוים – נצטרך למלא את ה `buffer` 4 פעמים (סה"כ לקרוא 20 תוים) לפני שנגיע לתו ירידת השורה.

הפונק' `close` – ניתן להשאיר את המימוש שלה ריק. אין לסגור את ה `FileReader` בתוך המחלקה.

הגעה לסוף הקובץ:

יש שני מקרים שבהם תדעו שה `FileReader` הגיע לסוף הקובץ:

1. ביקשתם לקרוא x תוים והפונקציה `read` של `FileReader` קראה פחות מ x תוים.

## 2. הפונקציה read של FileReader מחזירה 1-.

בשני מקרים אלה עליכם לזהות שכל תוכן הקובץ נקרא ולא להמשיך בקריאות read. בפרט, אם גודל ה buffer הוא x ומספר התווים בקובץ שלנו הוא קטן מ x, ה FileReader יבצע פעולת read אחת בלבד בשביל להחזיר את כל השורות.

אחרי ש MyBufferedReader יחזיר את כל השורות שקיימות בקובץ, קריאות נוספות ל getNextLine יחזירו null (זה יכול לשמש אתכם כתנאי עצירה בלולאות על תוכן כל הקובץ).

### בדקו את עצמכם:

הטסטר של חלק זה הוא מאוד בסיסי ובודק את התוכן הנקרא, אך לא את השימוש ב buffer. חלק מהעבודה שלכם היא לתכנן בדיקה שתוכל לבדוק גם את פעולת ה buffer (כלומר, שאתם משתמשים ב FileReader רק כאשר ה buffer מצריך זאת, ולא בכל פעולת read של המשתמש).

הנחיה: השתמשו במחלקה MyFileReader אשר מימושה מופיע בחבילת התרגיל. מחלקה זו יכולה לסייע לכם לנטר את מספר הקריאות שנעשו בפועל מהקובץ. מחלקה זו בנויה על פי design pattern שנקרא decorator והוא מאוד שימושי בבעיות רבות, כמו גם בתרגיל שלנו. ע"י מעבר על מימוש המחלקה הסיקו כיצד ניתן לשלבה בתסריט הבדיקות שלכם.

### הנחות והנחיות:

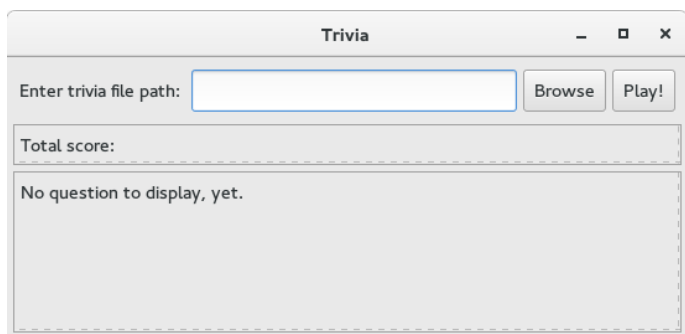
- א. עליכם לזהות ירידות שורה ע"י זיהוי התו '\n'. הניחו כי כל הקבצים שיקראו באמצעות הקורא שלכם נכתבו ב linux כך שאין צורך לטפל בתו '\r' (נוצר בקבצים נכתבו בעורכי טקסט של windows).
- ב. אין לייצר ולהשתמש בשום Stream למעט זה שמתקבל ע"י הבנאי של הקורא. שימו לב שאתם לא מקבלים את שם הקובץ ממנו אתם קוראים כך שכל העבודה מתבצעת ע"י הפעלת ה FileReader שקיבלתם בבנאי. כמות המידע שתיקרא בכל פעם תלויה בגודל ה buffer שגם הוא מאותחל בבנאי.
- ג. השורה האחרונה בקובץ לא חייבת להסתיים בירידת שורה – אך בשני המקרים (עם ירידת שורה ובלעדיה, ההתנהגות צריכה להיות זהה).
- ד. הניחו שגודל ה buffer שיתקבל יהיה תמיד חיובי וגדול מ 0.

## חלק ב': משחק טריוויה - GUI (50%)

בתרגיל זה נכתוב ממשק משתמש גרפי אשר מתפקד כמשחק טריוויה בסיסי. נתרגל עבודה עם רכיבי GUI שונים והבנת קוד נתון - שלדי המחלקות והמחלקות אשר מופיעים באתר. אינכם נדרשים לבצע פעולות של עיצוב ה-GUI, אלא רק של תפעול האפליקציה לפי מהלך המשחק.

### תיאור השימוש בממשק הגרפי

כדי לפתוח את הממשק הגרפי יש להריץ את התכנית TriviaMain. כעת, ייפתח חלון הראשי של התכנית.

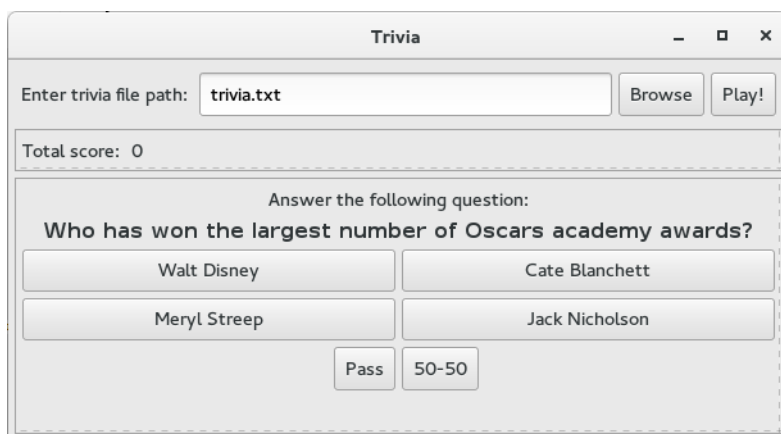


ניתן להקליד מסלול לקובץ בשדה הנתון, או ללחוץ על הכפתור Browse. במקרה הזה, ייפתח חלון לבחירת קובץ.

קבצי טריוויה הם קבצי טקסט בפורמט הבא: כל שורה מכילה שאלה ולאחריה 4 תשובות אפשריות, כאשר הראשונה מביניהן היא נכונה והיתר שגויות. בין השאלה לתשובות ובין התשובות מפריד טאב בודד. לדוגמא,

What is an apple?	Fruit	Vegetable	Fungus	Meat
When was the Beijing Summer Olympic Games?	2008	2014	2002	2000

לאחר בחירת מסלול הקובץ הרצוי ניתן ללחוץ על Play! בחלון הראשי. אם הקובץ תקין, הוא ייטען ע"י האפליקציה ותוצג למשתמש שאלה **אקראית מתוכו** ואת ארבע התשובות האפשריות עבורה **בסדר אקראי**. בשלב זה הניקוד הכולל של המשתמש הוא 0.



המשתמש יכול לבחור באחת התשובות או להשתמש באחד משני גלגלי הצלה: דילוג על השאלה (Pass) ו-50-50.

חוקי המשחק:

- תשובה נכונה = +3 נקודות
- תשובה שגויה = -2 נקודות
- המשחק מסתיים בעקבות אחד מהאירועים הבאים:
  - 3 תשובות שגויות ברצף (פסילה)
  - נגמרו השאלות במאגר
- גלגלי ההצלה זמינים לשימוש, מתחילת המשחק, ובכל שאלה מחדש תחת התנאים הבאים:
  - השימוש הראשון בכל גלגל הצלה (בנפרד) אפשרי ללא הגבלה או תלות במאזן הניקוד של השחקן
  - לאחר שימוש ראשון בגלגל הצלה מסוים, כל עוד מאזן הניקוד של השחקן קטן או שווה ל-0 לא ניתן לחזור ולהשתמש בו בשאלות נוספות.
  - לדוגמא, לאחר שימוש ראשון בכפתור 50-50, אם הניקוד הוא -2 הגלגל לא זמין לשימוש (לא ניתן ללחוץ עליו). ברגע שהשחקן יקבל ניקוד גדול מ-0 הגלגל יחזור לשימוש ויהיה זמין ללחיצה.
  - לאחר שהשחקן צובר נקודות ומאזנו חיובי ניתן לפתוח את השימוש בכל אחד מגלגלי ההצלה שוב
  - ניקוד שימוש בכל אחד מגלגלי הצלה:
    - חינם בפעם הראשונה
    - כל שימוש נוסף מעבר לפעם הראשונה = -1 נקודה
- כמו כן, תוצג למשתמש שאלה אקראית **חדשה** (המשתמש לעולם לא יקבל את אותה שאלה פעמיים באותו משחק). המשחק כאמור יסתיים לאחר 3 תשובות שגויות ברצף או כאשר סיימנו לעבור על כל השאלות בקובץ. בשלב זה תוצג למשתמש הודעה מסכמת. אחרי ההודעה התוכנית תישאר פתוחה ויהיה ניתן לטעון קובץ חדש וללחוץ על play להתחלת משחק חדש (או גם play עם אותו קובץ שטענו כבר). לאחר סיום המשחק והצגת ההודעה אין צורך לממש התנהגות מיוחדת לכפתורים 50-50 וpassi.



## תכנית לדוגמא

תכנית לדוגמא נמצאת בתיקייה src. בכל מקרה של ספק לגבי אופן פעולת המשחק במצב מסויים, ניתן להיעזר בדוגמא זו שכן ההתנהגות שלה היא הקובעת. יש לקחת את הקובץ Trivia.jar מתיקיית src ולשמור אותו בתיקייה כלשהי במחשב. באותה תיקיה, שמרו עותק של swt.jar המתאים למערכת ההפעלה שלכם כפי שיוסבר בחלק הבא.

## הורדה של SWT

ניתן להוריד את ה-jar מ-<http://www.eclipse.org/swt> ולנוחיותכם זה הקישור משם לגרסה האחרונה של swt שניתן להשתמש בה לתרגיל - <https://download.eclipse.org/eclipse/downloads/drops4/R-4.18-202012021800>

יש לבחור את הגרסה המתאימה של zip למערכת ההפעלה שלכם. מאחת מהאופציות הנ"ל:

### SWT Binary and Source

Platform	Download	Size
Windows (64 bit version)	swt-4.18-win32-win32-x86_64.zip	3.9 MB
Linux (64 bit version)	swt-4.18-gtk-linux-x86_64.zip	3.6 MB
Linux (64 bit version for Power PC)	swt-4.18-gtk-linux-ppc64le.zip	3.6 MB
Linux (64 bit version for AArch64)	swt-4.18-gtk-linux-aarch64.zip	3.6 MB
Mac OSX (64 bit version)	swt-4.18-cocoa-macosx-x86_64.zip	3.4 MB

למי שיש מערכת הפעלה עם 32 ביט (אפשר לבדוק האם זה המצב לפי המדריך [הזה](#)) אפשר להוריד גרסה ישנה יותר של swt שתומכת בכך כאן: <https://archive.eclipse.org/eclipse/downloads/drops4/R-4.9-201809060745/>

### SWT Binary and Source

Platform	Download	Size
Windows (32 bit version)	swt-4.9-win32-win32-x86.zip	4.1 MB
Windows (64 bit version)	swt-4.9-win32-win32-x86_64.zip	4.2 MB
Linux (32 bit version)	swt-4.9-gtk-linux-x86.zip	3.8 MB
Linux (64 bit version)	swt-4.9-gtk-linux-x86_64.zip	3.9 MB
Linux (64 bit version for Power PC)	swt-4.9-gtk-linux-ppc64le.zip	3.8 MB
Mac OSX (64 bit version)	swt-4.9-cocoa-macosx-x86_64.zip	3.4 MB

הזיפ שתורידו יכיל בתוכו את swt.jar, ויש להוציא אותו מתיקיית src ולשמור באותה תיקיה יחד עם Trivia.jar. כעת, כדי להריץ את התכנית לדוגמא מ-command line היכנסו לאותה תיקיה (ב-Windows למשתמשי terminal, ולמשתמשי Linux וכו'), והקישו את הפקודה

```
java -jar Trivia.jar
```

(לחלופין, בחלק ממערכות ההפעלה, לחיצה כפולה על קובץ ה-jar תפעיל אותו) אם פעלתם לפי ההנחיות, ו- java 8 מותקנת על המחשב, יפתח חלון עם משחק הטריוויה. (אם עדיין לא עובד לכם, נסו לשנות את שם ה-jar להיות swt בלבד ללא הסיימת jar). לעיתים יש להגדיל (למתוח) את החלון הנפתח כדי לראות את כל הכפתורים. על מנת להריץ על מחשבי Mac יתכן שיהיה צורך להשתמש בפקודה:

```
java -XstartOnFirstThread -jar Trivia.jar
```

## מה עליכם לעשות

באתר הקורס נתונים לכם שלדים של מחלקות ה-GUI בחבילה `il.ac.tau.cs.sw1.trivia`. הוסיפו ושנו קוד לפי הצורך וראות עיניכם. יש להוסיף ל-`build path` של הפרויקט את `swt.jar` המתאים למערכת ההפעלה.

איך לטעון את קובץ `swt.jar` לפרויקט שלכם

כדי להוסיף ל-`build path` של פרויקט קובץ `jar` של `eclipse` יש ללחוץ מקש ימני על הפרויקט->`Build Path` Configure `build path` (ניתן להגיע לשם גם ע"י מקש ימני על הפרויקט->`properties`) בחלון שיפתח יש לגשת לטאב `Libraries` שם יש ללחוץ על `Add External JARs..` ולבחור את `swt.jar` שהורדתם בחלק הקודם וללחוץ `Apply`. לאחר מכן קוד המחלקה אמור להתקמפל ויהיה ניתן להריץ את `TriviaMain`.  
באתר הקורס גם נתון קובץ טריוויה תקין לדוגמא, `trivia.txt`.

## פירוט

**[5 נק'] הכפתור `Browse`:** הוסיפו מאזין לכפתור זה כך שלחיצה עליו תפתח חלון לבחירת קובץ, וכך שאם נבחר קובץ המסלול אליו יתעדכן בשדה הטקסט המתאים. היעזרו במתודה `GUIUtils.getFilePathFromFileDialog` הנתונה לכם.

**[20 נק'] הכפתור `Play!`:** הוסיפו מאזין לכפתור כך שלחיצה עליו תגרום לטעינת השאלות והתשובות מהקובץ הנתון. ניתן להניח שפורמט הקובץ תקין.

שמרו את השאלות והתשובות במבנה נתונים לבחירתכם. ניתן להוסיף מחלקות כרצונכם. שימו לב, המזהה הייחודי של שאלה הוא הטקסט של השאלה בצירוף קבוצת התשובות שלה. כלומר, ייתכנו שתי שאלות עם אותו טקסט, אך עם קבוצת תשובות שונה (ראו דוגמא ב-`trivia.txt` שורות 6-7). לא ייתכנו שתי שאלות עם אותו טקסט ואותה קבוצת תשובות בדיוק.

לבסוף, הציגו שאלה אקראית מן הקובץ ואת התשובות שלה בסדר אקראי. כדי ליצור את הכפתורים היעזרו במתודה `updateQuestionPanel`. ניתן להיעזר במתודה `nextInt` של המחלקה `java.util.Random` כדי לקבל מספר אקראי, ו־או במתודה `Collections.shuffle` כדי לסדר את רשימת התשובות בסדר אקראי. אתחלו את הניקוד להיות 0, ושדות אחרים לפי הצורך.

**[15 נק'] כפתורי התשובות:** הוסיפו מאזינים לכפתורים האלה כך שלחיצה על הכפתור הנכון תגדיל את הניקוד ב-3 ולחיצה על כפתור של תשובה שגויה תקטין את הניקוד ב-2 ותגדיל ב-1 את מספר התשובות השגויות.

בדומה לכפתור ה-Play!, יש להציג את השאלה הבאה. נסו לשתף קוד בין מאזיני הכפתורים. במידה והמשחק נגמר (התקבלו 3 תשובות שגויות ברצף או סיימו את כל השאלות) יש להציג למשתמש הודעה מסכמת. לשם כך, היעזרו במתודה `GUIUtils.showInfoDialog`.

**101 נק' | כפתור Pass:** הוסיפו לכפתור זה מאזין המתקדם לשאלה הבאה מבלי לעדכן את מספר השאלות שנענו (המצוין בהודעה המסכמת). זיכרו לעדכן את מצב הכפתור והניקוד בהתאם לחוקי המשחק. שוב, נסו לשתף קוד עם המאזינים לכפתור Play! וכפתורי התשובות.

**101 נק' | כפתור 50-50:** הוסיפו לכפתור זה מאזין אשר מנטרל שני כפתורי תשובות לא נכונות ומשאיר רק כפתור עם תשובה נכונה ואחד עם תשובה לא נכונה באופן אקראי. זיכרו לעדכן את מצב הכפתור והניקוד בהתאם לחוקי המשחק (הניקוד יכול לרדת מיידית עם הלחיצה על הכפתור או לאחר מענה על השאלה). שוב, נסו לשתף קוד עם המאזינים לכפתור Play! וכפתורי התשובות.

## הערות

- לצורך פתרון התרגיל, חשובה יכולת קריאה והבנה של הקוד הנתון, תוך זיהוי החלקים הרלוונטיים לקוד שתרצו להוסיף. הקוד הנתון מתועד ברובו, אך ייתכן שתצטרכו להיעזר גם בתיעוד של SWT, בתיעוד של java ובאינטרנט כדי להבין חלקי קוד מסוימים. הבנה ושימוש בשלד הקוד הקיים הוא אתגר מרכזי בתרגיל.
- נדגיש שוב – בכל ספק בנוגע להתנהגות הרצויה, יש לבדוק את ההתנהגות לפי התוכנית לדוגמא ולממש את הקוד שלכם כך שיפעל באותה צורה.
- בחלק זה של התרגיל אתם רשאים לשנות את המחלקות והמתודות הנתונות לפי הצורך, מלבד פונקציית ה-main הראשית במחלקה `il.ac.tau.cs.sw1.trivia.TriviaMain`.
- לפני תחילת פתרון התרגיל, מומלץ לעבור על כל הקוד, ובפרט על המחלקה `GUIUtils`. ייתכן שתמצאו שירותים שיעזרו לכם בפתרון התרגיל (מומלץ אבל אין חובה להשתמש בהם).
- בדקו את עצמכם ע"י משחק בתכנית.
- ניתן להניח כי ה-path שמתקבל כקלט הוא תקין ומוביל לקובץ טריוויה תקין.
- יש להגיש את כל המחלקות (קבצי .java), גם אם לא שיניתם אותן. אין צורך להגיש את `swt.jar`, אבל זה לא משנה גם אם תגישו אותו.
- את חלק ה-GUI אין חובה לבדוק בהרצה בcmd ובלינוקס, מספיק שהתוכנית עובדת באקליפס (אך עדיין שימו לב לא להשתמש בimportים שלא נראו בקורס ולא להשתמש בפתרונות לpath שספציפיים למערכת הפעלה מסוימת). על מנת כן לבדוק את עצמכם יש לארוז את הקוד שלך בקובץ `jar` בדומה לקובץ הדוגמה, ניתן להיעזר במדריך [הזה](#).

בהצלחה !