**תרגיל 2 – הגשה עד ה-2.5.21 בשעה 23:55**

**חלק ראשון**

המחלקה Turtle מגדירה עבורנו צב רגיל. יצירה של צב מציירת דמות של צב במרכזו של מסך גרפי. המסך הגרפי נוצר עם יצירתו של צב בפעם הראשונה. לצב יש יכולת תנועה. הוא יכול להסתובב סביב עצמו ימינה ושמאלה בכל מעלה שלמה ולפנות לכיוונים שונים. צב חדש נוצר כשהוא פונה כלפי מעלה. צב יכול לנוע קדימה אל הכיוון אליו הוא פונה לכל מרחק נתון.

לצב יש זנב. הזנב יכול להיות מורם או מורד. אם הזנב מורד והצב נע קדימה הצב משאיר לאורך מסלולו עקבות בצורה של קו על המסך הגרפי. אם הזנב מורם הצב לא משאיר עקבות. צב גם ניתן להסתרה וגילוי מחדש. בין אם הצב גלוי ובין אם הוא מוסתר הוא מבצע את כל הפעולות באותו האופן. כלומר, הוא יכול להסתובב, להתקדם קדימה, להשאיר עקבות כאשר זנבו מורד, או לא להשאיר סימן כאשר זנבו מורם.

לממשק הציבורי של המחלקה Turtle המאפיינים והיכולות הבאים:

Turtle (); // construct turtle

public void show (); // show yourself

public void hide (); // hide yourself

public void tailDown (); // lower the tail

public void tailUp (); // lift the tail

public void turnLeft (int degrees); // turn left in the given degrees

public void turnRight (int degrees); // turn right in the given degrees

public void moveForward (double distance); // advance forward in given distance

**שימו לב!** לצב יש מספר שיטות נוספות הקשורות לאופן הפעולה שלו ושל הממשק הגרפי. התעלמו משיטות אלה, הן אינן נחוצות לעבודה ועלולות לפגום בתוכנית שלכם.

כדי להשתמש במחלקה Turtle יש להוריד מאתר הקורס אל המחשב שלכם את החבילה Turtle.jar, שמאגדת בתוכה מספר קבצים הנדרשים לעבודה עם הצב, יש להגדיר פרויקט ולהוסיף לו את החבילה ולייבא (import) את החבילה לקבצים כך שנוכל להשתמש בה בקבצים (ראו הוראות מפורטות לכך בסוף המסמך)

לכתיבת החלק הראשון עקבו אחרי המשימות הבאות.

משימה ראשונה – הכרות עם המחלקה Turtle

כתבו תוכנית בשם TwoTurtles.java אשר יוצרת שני צבים בשם Rick ו- Morty.

ריק ומורטי רוצים ליצור מפלצת חדשה עם ראש מרובע. כדי לעשות זאת, עליהם לעצב סקיצה עבורה. ראשית ריק יצייר את הראש ופלג גוף עליון (כחול), ולאחר מכן מורטי יצייר את הידיים ורגליים (כתום). אין צורך לצבוע את הקווים בצבעים המופיעים פה, הם לצורך המחשה.

שימו לב, מטרת המשימה היא לאפשר לכם הכרות עם המחלקה Turtle, לכן אין גודל מסויים שנדרש לחלקים השונים בציור. סביר מאוד להניח שהאובייקטים לא יצאו ישרים לחלוטין. אין בעיה עם זה, כל עוד יזכירו בצורתם את התמונה הבאה:

משימה שניה – הגדרה של צבים עם תכונות שונות ע"י הורשה

לא כל הצבים נוצרו שווים. יש חכמים, יש מוכשרים, ולא עלינו, מוזרים. הוסיפו מחלקות שיגדירו את הצבים הבאים:

* צב אינטיליגנטי (IntelligentTurtle) – צב אינטליגנטי, מעבר להיותו צב רגיל לכל דבר ועניין, מבין גם משהו בגיאומטריה: הוא יודע לצייר מצולע משוכלל בעל מספר צלעות נתון באורך נתון. כתבו את המחלקה IntelligentTurtle , שימרו אותה בקובץ בשם IntelligentTurtle.java במחיצת העבודה, והוסיפו לה את השיטה הבאה:

public void draw (int sides, double size);

// draw a polygon in the given sides and size

שימו לב: כיוון שהזויות העוברת כפרמטר בשיטות turnLeft ו- turnRight היא זווית שלמה עלולה להתעורר בעיה במקרה של פוליגון משוכלל עם זווית לא שלמה. התעלמו מהבעיה. דאגו רק שהצב האינטיליגנטי יצייר נכון פוליגונים בעלי זווית שלמה. Size הינו הגודל של כל צלע.

* צב שיכור (DrunkTurtle) – צב שיכור הוא צב רגיל ששתה מעט וכתוצאה מכך קשה לו קצת ללכת. כשהוא מתבקש לנוע קדימה במרחק **x** הוא מבצע את הפעולות הבאות אחת אחרי השנייה לפי הסדר (הוא מבצע את כולן בכל פעם שהתבקש לנוע קדימה):
  + הוא מתקדם למרחק מקרי בין 0 ל- **x**.
  + בהסתברות של 45% הוא פונה 90 מעלות ימינה.
  + בהסתברות של 35% הוא ילך אחורה מרחק מקרי בין 0 ל-**x**, אחרת הוא ילך קדימה מרחק מקרי בין 0 ל- **x**.

הערה: שימו לב שאם בהסתברות 45% הוא פונה 90 מעלות ימינה, בהסתברות המשלימה הוא לא יפנה וימשיך באלגוריתם.

בנוסף, בכל פעם שהוא מתבקש לבצע פנייה בזווית y הוא יפנה בזווית אקראית בין 0 ל- y1.5.

כתבו את המחלקה DrunkTurtle ושימרו אותה בקובץ בשם DrunkTurtle.java במחיצת העבודה.

* צב מקרטע (JumpyTurtle) – צב מקרטע הוא צב אינטיליגנטי שמקרטע: כאשר זנבו מורד (מצייר) הוא מנתר לסירוגין מאושר. התוצאה היא שכאשר זנבו מורד הוא משאיר קו מקווקו, כלומר כאשר הורידו את זנבו למטה וביקשו ממנו ללכת התוצאה תהיה קו מקוטע ולא חלק. הקווקו יהיה בנוי כך שיופיע בכל פעם קו באורך 4 יח' ולאחר מכן 4 יח' מרחק ללא קו.

כתבו את המחלקה JumpyTurtle ושימרו אותה בקובץ בשם JumpyTurtle.java במחיצת העבודה שלכם. שנו בה את הדרוש שינוי.

**שימו** **לב**: כיוון שצב מתקדם על פני סריג (- - -) של נקודות, התקדמות למרחק קצר עלולה ליצור אי דיוק גדול יחסית . אם ההתקדמות של צב מקרטע תשבר לרצף ארוך של פסיעות ודילוגם קטנים, אי הדיוק יצטבר והצב עלול להתקדם בפועל למרחק שונה מהנדרש. התעלמו מהבעיה.

**בנוסף**, שימו לב כי הקלט לשיטות לא בהכרח יתקבל בכפולות של 4. לכן עליכם להחליט על אופן ההתמודדות עם הבעיה (לדוגמא, אפשר לוותר על היחידות (לעגל), אפשר להמשיך ללכת במספר היחידות שנותרו.)

* צב עצלן (LazyTurtle)- צב עצלן הוא כמו צב שיכור, אך לא תמיד יהיה לו כוח לבצע את הפעולות שהוא נדרש לבצע. בכל פעם שמבקשים ממנו משהו (יבצע אחד מהבאים):
  + בהסתברות של 30% הוא יבצע את הפעולה כמו צב שיכור.
  + בהסתברות של 20% הוא יבצע את הפעולה כמו צב פשוט.
  + בהסתברות של 50% הוא לא יבצע כלום.

\*הערה- ההסתברויות משלימות, כלומר הוא בטוח יבצע אחת משלושת הפעולות.

כתבו את המחלקה LazyTurtle ושימרו אותה בקובץ בשם LazyTurtle.java במחיצת העבודה. שנו בה את הדרוש שינוי.

* צב בורמזי (BurmeseTurtle)- צב בורמזי הוא כמו צב חכם, אך הוא צב פחות ממושמע ולכן לא מבצע פעולות מסוימות כפי שנדרש. אם נבקש ממנו לצייר מצולע:
  + בהסתברות של 50% הוא יצייר מצולע עם אורך צלע שקבעתם, אך בעל צלע אחת חסרה ויעלם מיד לאחר מכן.
  + בהסתברות של 50% הוא יצייר מצולע עם מספר הצלעות שקבעתם, אך עם אורך צלע של 18 (ללא קשר לאורך הצלע שקבעתם).

\*הערה- ההסתברויות משלימות, כלומר אם לא ביצע את האופציה הראשונה, הצב יבצע את האופציה השנייה.

כתבו את המחלקה BurmeseTurtle ושימרו אותה בקובץ בשם BurmeseTurtle.java במחיצת העבודה. שנו בה את הדרוש שינוי.

חשוב ביותר: על כל הצבים מכל הסוגים לא להשאיר מאחוריהם עקבות כאשר הזנב שלהם מורם.

משימה שלישית – ניהול צבא של צבים באמצעות פולימורפיזם

כתבו תוכנית בשם Army.java שתנהל צבא (מערך) של 5 צבים.

בשלב ראשון אפשרו למשתמש לבחור את צבא הצבים כרצונו. הציגו לפניו את התפריט שלמטה וקבלו את בחירתו עבור כל אחד מחמשת הצבים בנפרד. אפשרו לו לבחור כל תערובת של צבים. להלן התפריט:

Choose the type of a turtle:

1. Simple
2. Intelligent
3. Drunk
4. Jumpy
5. Lazy
6. Burmese

* בשלב שני צרו את הצבים הנדרשים וקדמו אותם שלב אחר שלב על פני השלבים הבאים:
  1. הסתדרות בשורה (מרחק בין צב לצב 120, כאשר לא נשארים סימנים בהליכה)
  2. הורדת זנב
  3. צעידה קדימה למרחק של 65
  4. פניה של 40 מעלות שמאלה
  5. צעידה קדימה למרחק של 75
  6. כל מי שיודע לצייר צריך לצייר משושה בעל צלע בגודל 40.
  7. העלמות הצבים

בסוף הריצה נשארים ציורי הצבים בלבד, ללא הצבים.

שימו לב:

* על הצבים להתקדם כולם יחד. אף צב לא יכול לעבור לשלב גבוה יותר בטרם גמרו כל הצבים האחרים את השלב הקודם. השלבים יחלו באופן אוטומטי לאחר שהמשתמש בחר את הצבים. כאשר הצבים פועלים על פי הסדר שבו הם נבחרו.
* כל הצבים הולכים בוודאות וישר למיקום הראשוני ורק לאחר ארגונם מתחילים להתנהג לפי אופיים.

הערות לחלק הראשון:

* יתכן כי הצב השיכור, צב עצלן וצב בורמזי "ידרסו" חלק משרטוטי הצבים האחרים –זה בסדר.
* יתכן ותדרשו להגדיל את המסך הנוצר ע"מ לראות את הציורים – זה בסדר.
* הצבים יהיו ממוקמים משמאל לימין לפי סדר בחירתם.

חלק שני

בתרגיל הראשון התבקשתם לממש את המשחק דמקה, המנוהל בין משתמש למחשב. בתרגיל זה תצטרכו לממש את המשחק דמקה יונקים. על המימוש להיות מודולרי ובהתאם לעקרונות הנלמדים בקורס. המשחק מכיל לוח בגודל 8X8 של משבצות המכילות יונקים מסוגים שונים.

לכל כלי משחק – יונק, ישנם חוקי תזוזה ואכילה משלו, כאשר ייתכנו חיות עם חוקים חופפים.

#### חוקי הדמקה הבסיסיים – מופיעים בעבודה 1 (במידה ולא צוין אחרת, הכלים מתנהגים לפי כללי התנועה והאכילה המופיעים בעבודה 1.)

#### סוגי הכלים - חיות

* כלב (Dog) – הכלב נע בהתאם לחוקים רגילים (רק באלכסון), אך מסוגל לבצע אכילה אחת בלבד בתור. אם בתום האכילה אפשרי לבצע אכילה נוספת, לא יתאפשר לו לבצע אותה.
* חתול (Cat) – חתול יכול לנוע משבצת אחת לכל כיוון שייבחר ולכל סוג משבצת (גם למשבצות "-" וגם "\*" - לאו דווקא באלכסון). עם זאת, אכילה מתבצעת בהתאם לחוקים הרגילים- רק באלכסון. משמע אם חתול נמצא על משבצת \*, לא יתאפשר לאכול כלי **שאינו** על משבצת \* .
* פיל (Elephant) – הפיל נע באלכסון 2 משבצות בבת אחת (רק באלכסון). כאשר יבצע אכילה, האכילה תתבצע בהתאם לחוקים הרגילים (הפיל חייב להיות סמוך במרחק של משבצת אחת באלכסון ליונק אחר על מנת לבצע את האכילה והוא קופץ מעליו לתא הבא אחריו ומעלים את הכלי שאכל).
* עכבר (Mouse)– העכבר נע באלכסון 2 משבצות כמו הפיל, אך אינו מעלים כלים כאשר הוא ״אוכל״ אותם. כאשר יגיע למצב בו תתאפשר לו ״אכילה״, יהיה חייב לבצע את התנועה (ידלג מעל כלי היריב), אך כלי היריב יישאר במקומו. באותו אופן אם קיימת לו אכילה כפולה, יהיה מחוייב לבצע את התנועה אך הכלי של היריב יישאר במקומו.

#### מלכות

כשכלי משחק (יונק) רגיל מגיע לשורה האחרונה הנגדית, הוא הופך להיות "מלכה". ייצוג מלכה יתבצע ע"י הוספת האות Q לייצוג הכלי. למשל, אם מדובר בפיל של המחשב שהפך למלכה הייצוג יהיה E2Q.

רק פילים ועכברים (E, M) יכולים להפוך למלכות. כלים אחרים שיגיעו לשורה האחרונה ימשיכו ככלים רגילים – הכלב ייתקע בשורה האחרונה והחתול ימשיך כרגיל (הוא יכול לנוע לכל הכיוונים מבלי להיות מלכה).

בנוסף, אם חולף כלי על פני השורה האחרונה תוך ביצוע 'אכילה' כפולה, ומסיים את המהלך באחת השורות שלפני השורה האחרונה, הוא אינו הופך למלכה, ולמעשה ממשיך ככלי רגיל בהתאם לחוקים שלו.

פיל מלכה – יכולה לנוע באלכסון 2 משבצות לכל הכיוונים. מחויבת לאכילה מרובה במידה ומתאפשרת.

עכבר מלכה – יכולה לנוע לכל הכיוונים באלכסון. כאשר תגיע למצב בו תתאפשר לה אכילה, היא מחויבת לבצע את התנועה (תדלג מעל כלי היריב), אך כלי היריב יישאר במקומו.

#### סידור כלים על הלוח

כל שחקן יתחיל עם 3 כלבים, 3 חתולים, 3 פילים, 3 עכברים. סידור הכלים ההתחלתי יתבצע על משבצות ה"\*" בצד של כל שחקן **ויהיה זהה לאיור המצורף בדוגמת ההרצה.**

המשתתפים ימוספרו ע"י המספרים 1 ו-2, כאשר המשתמש הינו שחקן 1 והמחשב הינו 2.

באמצעות צירוף של האות הראשונה בשם החיה באנגלית ומספר השחקן. למשל, עבור חתול השייך למחשב הייצוג יהיה "C2". עבור כלב השייך למשתמש הייצוג יהיה "D1".

#### מהלך המשחק

את פעולת הזזת כלי או האכילה מבצע השחקן ע"י הזנת קואורדינטות הכלי וקואורדינטות היעד, לאחר הצגת שורת הפלט:

It's your turn, please enter your move:

לדוגמא: אם המשתמש ירצה לסמן כלי (3,2) , ולהזיזו לתא (4,1) יהיה עליו להקליד את הקלט 41-32. על המשחק לבחון שאכן המהלך שמבצע השחקן הוא חוקי (אם מבצע תנועה – לבדוק שלא קיים כלי שמפריע, ואם מבצע 'אכילה' יש לבדוק שאכן יש כלי יריב שניתן לדלג מעליו, וכו'...) במידה והמהלך לא חוקי, תוצג לשחקן ההודעה הבאה:

This move is invalid. Please enter a new move.

לאחר שהמשתמש משחק, ישחק המחשב את תורו. המחשב ייבחר כלי אחד אקראי מבין הכלים שברשותו, ויזיז אותו למיקום אקראי אפשרי (מבין המיקומים החוקיים אליהם אותו הכלי יכול לזוז). במידה וכלי של המחשב יהיה מחויב לאכילה, יבצע אותה במקום לבצע מהלך אקראי.

יש להמשיך ולקלוט מהלכים מהשחקנים (המשתמש והמחשב) לסירוגין עד אשר יקרו אחד משני מקרים: מקרה ראשון – אחד השחקנים ניצח ולא נותרו עוד כלים לשחקן היריב, מקרה שני – מקרה של תיקו בו לשחקן שתורו כעת לשחק אין מהלכים חוקיים לבצע (כל כליו חסומים), מקרה שלישי – מצב בו השחקן מכניס לתוכנית STOP שמסמל כי השחקן נכנע. במידה ואחד השחקנים ניצח יוצג למשתמש ההודעה הבאה במידה והמשתמש ניצח:

Congratulations, User has won :)

וההודעה הבאה במידה והמחשב ניצח:

Sorry, Computer has won :(

במידה ויווצר תיקו בין השחקנים, ההודעה שתוצג היא:

Well, unfortunately it’s a Tie…

לאחר סיום המשחק יופעל המשחק מחדש, כאשר הלוח יסודר מחדש, והשחקנים יוכלו לחזור ולהתחרות על הניצחון.

דוגמת הרצה חלקית

(החץ 🡨 מייצג enter):

Welcome to Fatma Checkers. To start the game press 1, to exit press 0:

1

🡨

The board:

1 2 3 4 5 6 7 8

1 - C2 - D2 - E2 - M2

2 D2 - E2 - M2 - C2 -

3 - E2 - M2 - C2 - D2

4 \* - \* - \* - \* -

5 - \* - \* - \* - \*

6 E1 - M1 - C1 - D1 -

7 - D1 - E1 - M1 - C1

8 C1 - D1 - E1 - M1 -

It's your turn, please enter your move.

52-61

🡨

שימו לב שבמקרה בו המשתמש נדרש לבצע מספר "אכילות", הוא יבצע אותן בשניים או יותר תורים שונים ועוקבים, כאשר ביניהם המשחק לא יעבור למחשב.

#### כיצד לכתוב את העבודה

#### שלב 1

כתבו את המחלקות השונות של החיות, כאשר לכל חיה ממשו את שיטות התנועה, אכילה, והאופן בו תוצג על הלוח. שימו לב לכללי ירושה, וחישבו על הקשרים בין המחלקות השונות. ניתן להוסיף מחלקות נוספות שאינן מתוארות בהוראות.

#### שלב 2

בשלב זה הפכו את הקוד שלכם מתרגיל 1 למודולארי ע"י שימוש בפולימורפיזם. כלומר תיצרו לוח המסוגל לאחסן חיות, תמקמו את החיות לפי ההוראות, והריצו את אלגוריתם המשחק. שימו לב להיצמד לעקרונות מודולריות ואבסטרקציה. בנוסף, שימרו על כללי הכמסה והכלה.

הערות לשני החלקים

* הוראות כיצד להוסיף את המחלקה צב פשוט מצויינות בהמשך המסמך.
* הדגש בעבודה זו הוא על שימוש נכון במבנה ירושה ופולימורפיזם.
* מותר להוסיף מחלקות נוספות כראות עיניכם, בשני החלקים, גם אם לא נרשמו במפורש בעבודה. עם זאת, עליכם לתכנת באותם שמות את כל המחלקות המופיעות בהוראות.
* יש להשתמש בעקרונות וכלים של פיתוח תוכנה מונחה עצמים שנלמדו בכיתה. עבודות בהן לא יישומו העקרונות יקבלו ציון נמוך גם אם הפלט יהיה נכון.
* הריצו את התוכנית ובדקו אותה. העבודה כוללת סה"כ 3 תכניות (קבצים עם main) - משימת ההיכרות עם הצבים, צבא הצבים, ומשחק הדמקה המשודרג.
* שימו לב, אל תשתמשו ב- casting אלא אם כן הוא הכרחי.
* אין צורך בבדיקת קלט יש להניח כי הקלט תקין.
* הקפידו על כתיבה קריאה וברורה -שמות ברורים למשתנים, אינדנטציות וכו'.
* יש להקפיד על רישום הערות **באנגלית**, בפרט בתחילת כל פונקציה. חוסר בהערות יגרור הורדת נקודות.
* שימו לב – התשובות לשאלות הנשאלות בפורום מהוות הבהרות לדברים שלא הובנו בהנחיות, ועל כן מחייבות את כלל הסטודנטים לכל דבר ועניין. אך ורק במידה ויהיה צורך בשינוי נוסח ההנחיות ייצא מייל בנושא. אין לשאול בפורום שאלות הקשורות לאופן מימוש העבודה

#### הוראות הגשה

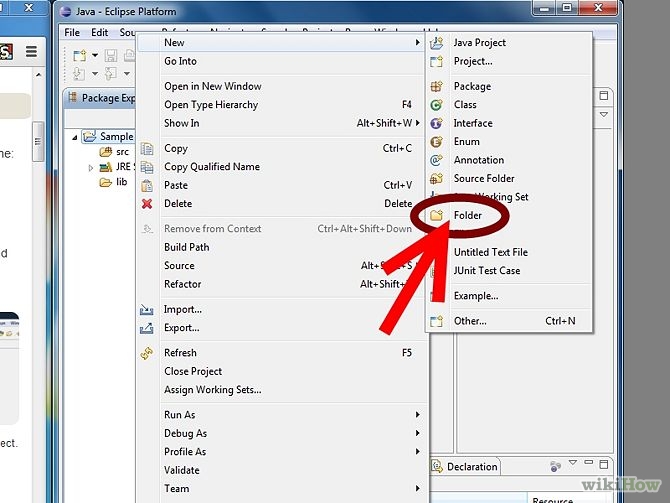
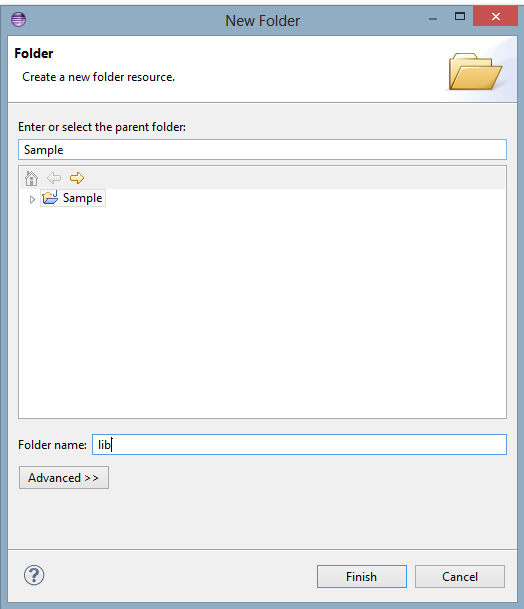
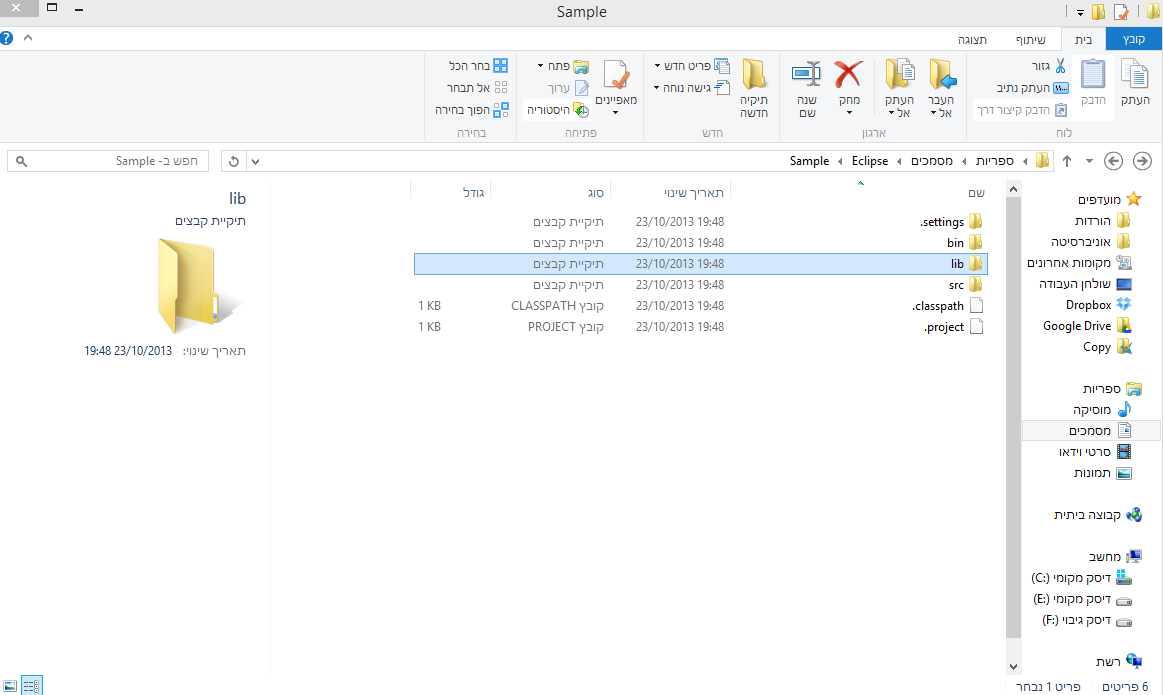
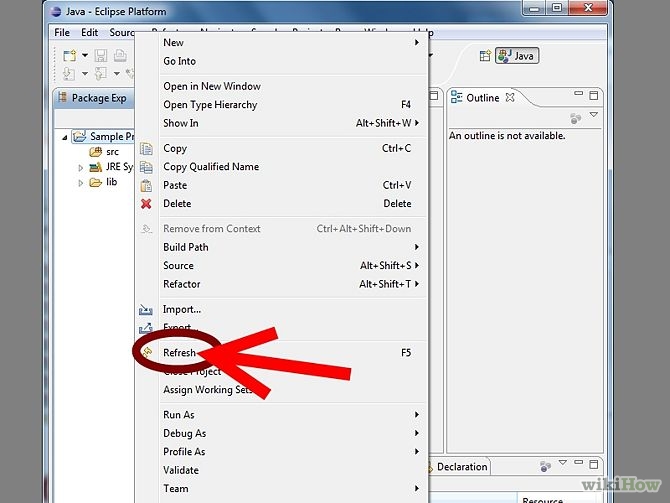
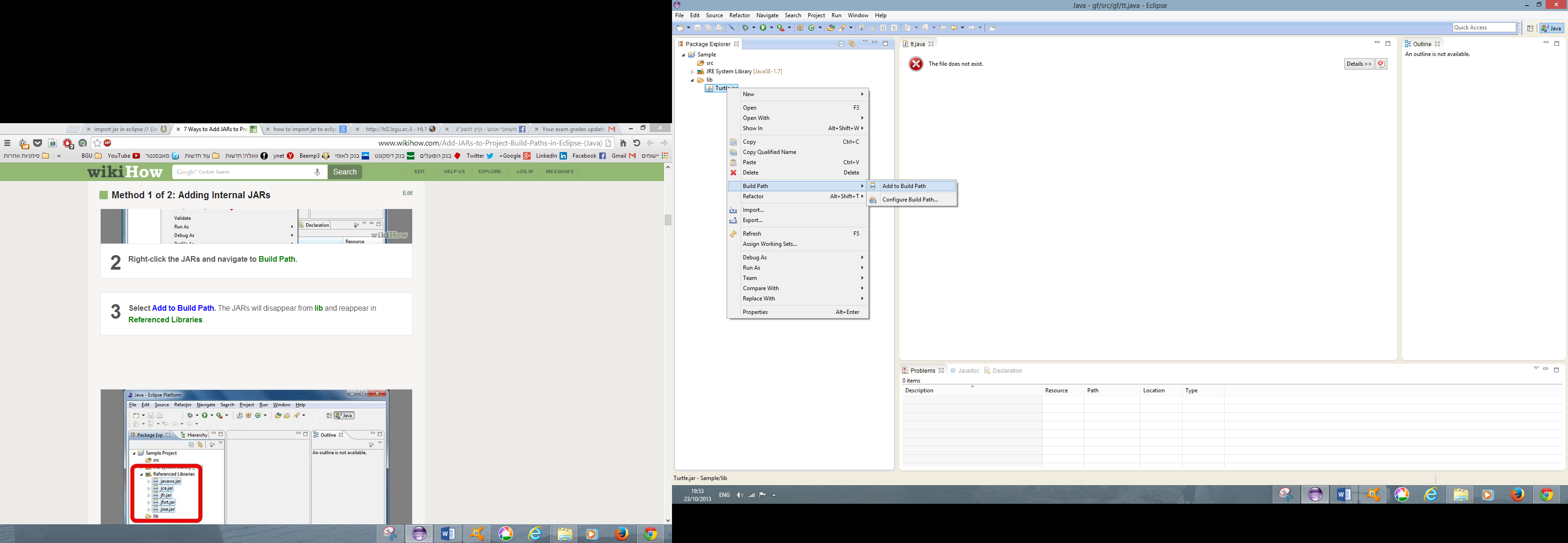
* **עליכם להגיש מסמך תכנון למחלקות (בקובץ וורד או pdf) עבור החלק השני בלבד – דמקה. המסמך יכלול תרשים המתאר את קשרי ההורשה בין המחלקות, שיטות חשובות הנמצאות בכל מחלקה, ואילו מהן נדרסות אצל היורשים.**
* הגשה בזוגות עד לתאריך – 2.5.21 בשעה 23:55.
* יש להגיש קובץ המכיל את כל הפרויקט מהאקליפס. ניתן לצפות בסרטון הדרכה בלינק הבא: <https://www.youtube.com/watch?v=nR0aNcWQn_I>
* יש להגיש תיקייה אשר שמה יהיה מספרי תעודת הזהות. לדוגמא: 111111111\_222222222

בתוכה יש לשים את תיקיית הפרויקט וקובץ תיאור הצב שלי.

* בודק התרגילים- אורי מילבאוור, ורד בר-ארבל, רועי וולף.
* מתרגלת אחראי- מאיה לביא
* ההגשה היא לתיבת ההגשה בלבד ולא למייל.
* דחיות לעבודה יינתנו ע"י אחראי הקורס רועי זיוון בלבד.
* כל יום איחור גורר הורדה של 5 נקודות.

עבודה מהנה!

How To Add the Turtle Package:

1. Download Turtle.zip from the course site via Moodle.
2. Extract Turtle.jar from Turtle.zip.
3. Start Eclipse and create a project
4. Right click on the project -> New -> Folder
5. Name the new folder "lib".
6. From Windows Explorer, copy Turtle.jar to the new library (will be found under %project path%\lib\)
7. In Eclipse, right click on the project -> Refresh
8. Under the project, inside the lib library, the Turtle.jar will be found. Right click -> Build Path -> Add to Build Path

Reminder : Don’t forget to import the package to your files that use it by writing :

"import Turtle.\*;"

As the first line in your file.

Good luck

The course staff