Known Issues

ראשית לשם הבנה כללית של התכנית אצרף את מטלת הפרוייקט מקורס 10118 (עשיתי את הקורס בסמסטר א' הקודם אצל ויקטור טאובקין). מדובר בהעמסת קונטיינרים על גבי אונייה באמצעות כפתור הmove. ניתן להעמיס מס' מכולות אחת על גבי השנייה. כפתור הrestart יאתחל את האונייה שבנמל מחדש.

הערות כלליות לגבי השינויים שנעשו בפרויקט המקורי לטובת המטלה:

- בpackage "application" במחלקה "Controller" שיניתי את הsetOnAction כך שיתפקד על ידי lambda expression במקום בדרך הארוכה. כמו כן במחלקה זו מופיעים האתחולים עבור כל אחד מהכפתורים מהחלונית עם הGUI. היצירה עצמה של הכפתורים נמצאת בpackage "View" במחלקה "MainView" כנהוג במודל MVC.
- בpackage "Model" במחלקה "Port" בשורה 136 - במקור, ההגרלה הרנדומלית עבור הייצוג העשרוני של כל גוון צבע למכולה (מודל RGB) היה בטווח הערכים המקסימאלי (0 עד 256), אך לטובת סעיף 13 בו נדרשת כפילות בין שתי תכונות במספר מופעים של האובייקט, ביצעתי שינוי בערכי הred והgreen של האובייקט כך שיוגרלו בין 0 ל 2, מה שגרם למכולות להיצבע כעת בגווני כחול במקום בצבעוני. (אם היה נשאר כמקור כך שעבור כל ערך יוגרל טווח הצבעים המקסימאלי - היה סיכוי קלוש ששני גוונים ייצאו זהים לשני אובייקטים בשתי תכונות כמבוקש, ובשאר התכונות שאינן צבע לא ייתכן שיוויון בין שני אובייקטים).
- בpackage "Model" במחלקה "Container" בשורות 142 ו151 הוספתי מתודות hashCode ו- equals עבור סעיף 13. השיטות הללו לא היו קיימות ולא היה בהן צורך בפרוייקט המקורי.
- כל שאר המתודות שנוספו עבור המטלה נמצאות בpackage "Model" במחלקה "Model" החל משורה 137.

כעת אתייחס לכל אחד מהסעיפים במטלה הנוכחית:

12. עבור ביצוע הסעיף הוספתי בחלונית של הJavaFX כפתור בשם "add to collection 1". בלחיצה על הכפתור, מופעלת המתודה copyContainersToColl1() (נמצאת בpackage "Model" במחלקה "Model" בשורה 138) והקונטיינרים שעל מחסנית הנמל מועתקים לתוך TreeSet. בנוסף מודפסים לconsole כל פרטי הקונטיינרים שנמצאים כעת על האונייה בסדר יורד לפי תכונת הleft שלהם (מעין ערך הX של הפינה השמאלית העליונה של המכולה). נשים לב שאם הועברו כבר חלק מהמכולות אל המשאית ולוחצים שוב על הכפתור " add to

collection 1" אז מודפסים לconsole רק הקונטיינרים שכעת על האונייה, פחות אלו שכבר על המשאית או לצד הדרך. המיון מבוצע בעזרת comparator.

13. עבור ביצוע הסעיף הוספתי בחלונית של הJavaFX כפתור בשם "add to collection 2". בלחיצה על הכפתור, מופעלת המתודה copyContainersToColl2() (נמצאת בpackage "Model" במחלקה "Model" בשורה 145) והקונטיינרים שעל מחסנית הנמל מועתקים לתוך TreeSet וממנו לתוך HashSet. בלחיצה זו נוספים לHashSet ומודפסים לconsole רק פרטי הקונטיינרים כך שאם קיימת מכולה/מכולות שערכי תכונות הred והgreen שלה/ם שווים לאותם ערכי red וgreen של מכולה/מכולות אחרים, ייכנס ויודפס רק אחד מהם, כלומר הHashSet לא יכניס איברים חדשים שהם כפילות עם מה שכבר נמצא בו. הברירה נעשית בעזרת HashCode ו- Equals.

14. כל אחת מהשיטות לעיל מפעילה את המתודה print(Set<?> set) אשר מדפיסה את תוכן הcollenction שהתקבל כפרמטר בעזרת Iterator.

15. כפי שציינתי מעלה, בpackage "application" במחלקה "Controller" שיניתי את הsetOnAction כך שיתפקד על ידי lambda expression במקום בדרך הארוכה.

16. מתועד לעיל.

17. הוספתי בPackage "Model" מחלקה בשם "MyArrayList". **הערה – ייתכן שבהקשר הMVC זה אינו המקום המתאים ליצור את המחלקה הזו, מכיוון שהקורס הנוכחי אינו עוסק בMVC אז לא שמתי דגש על כך.

18. עבור ביצוע הסעיף הוספתי בחלונית של הJavaFX כפתור בשם "My Array List". בלחיצה על הכפתור, מופעלת המתודה copyContainersToMyArrayList() (נמצאת בpackage "Model" במחלקה "Model" בשורה 208) והקונטיינרים שנמצאים בHashSet מועתקים לתוך MyArrayList. בלחיצה זו נוספים לMyArrayList ומודפסים לconsole רק פרטי הקונטיינרים כך שאם קיימת מכולה/מכולות שערכי תכונות הred והgreen שלה/ם שווים לאותם ערכי red וgreen של מכולה/מכולות אחרים, ייכנס ויודפס רק אחד מהם, כלומר הMyArrayList מעתיק מהHashSet אחד לאחד לפני הסדר את המכולות ללא הכפילויות.

19. הוספתי למחלקה "MyArrayList" את המתודה iterator() אשר מחזירה מימוש איטרטור פרטי (המימוש במחלקה פרטית "MyIt" אשר גם כך נמצאת ב"MyArrayList"). כמו כן הדפסת התוכן נעשית דרך הכפתור "My Array List". הפלט אכן זהה לפט שהודפס בעזרת הHashSet.

20. עבור ביצוע הסעיף הוספתי בחלונית של הJavaFX כפתור בשם "remove 1". בלחיצה על הכפתור, מופעלת המתודה removeContainers1() (נמצאת בpackage "Model" במחלקה "Model" בשורה 219) והיא מוחקת מהרשימה את כל המכולות אשר ערך התכונה red שלהם (רמת הצבע האדום במודל RGB) שווה ל-0. לאחר מכן גם מבוצעת הדפסה של התוכן החדש של הMyArrayList.

21. עבור ביצוע הסעיף הוספתי בחלונית של הJavaFX כפתור בשם "Java Array List". בלחיצה על הכפתור, מופעלת המתודה ()copyContainersToJavaArrayList (נמצאת בpackage "Model" במחלקה "Model" בשורה 214) והקונטיינרים שנמצאים בHashSet מועתקים לתוך ArrayList של ג'אבה. התוצאה המתקבלת זהה לזו שבסעיף 18. כמו כן אותו כנ"ל לגבי הremove: הוספתי בחלונית של הJavaFX כפתור בשם "remove 2". בלחיצה על הכפתור, מופעלת המתודה ()removeContainers2 (נמצאת בpackage "Model" במחלקה "Model" בשורה 225) והיא מוחקת מהרשימה את כל המכולות אשר ערך התכונה red שלהם (רמת הצבע האדום במודל RGB) שווה ל-0. לאחר מכן גם מבוצעת הדפסה של התוכן החדש של הArrayList. ***הערה מסעיף 17 ועד כאן – ניתן לשחק ולהעמיס מכולות מהאונייה למשאית ואז ללחוץ שוב על הכפותורים, ולראות שהתוכן של הסטים שמודפס משתנה בהתאמה.
22. מתועד לעיל.
23. **הערה לגבי הסעיף הנ"ל – לא הייתה התייחסות בשיעורים להיכן צריך להוסיף את המחלקות החדשות שנדרשות בהתאם למודל הMVC, לכן הוספתי לפי שיקול דעתי. יצרתי בpackage "Model" את הממשק "Observer" ואת המחלקות "MyButton" ו-"MyLabel", המשמשות כListeners עבור הSubject שהוא המחלקה "MyArrayList". כאשר נלחץ על הכפתור "My Array List" המאזינים יגיבו לכך ויופיע על החלונית הכפתור והתווית המבוקשים (נעשה בעזרת השיטה ()click הנמצאת בpackage "Model" במחלקה "MyArrayList" בשורה 62). כמו כן הלחיצה על הכפתור שהופיע, יודפס למסך תוכן האיטרטור כפי שנוצרו בו קודם לכן בלחיצה על "My Array List". *הערה נוספת – אם נוציא מכולות מהאונייה למשאית, ולאחר מכן נלחץ שוב על הכפתור "iterator" אז התוכן לא ישתנה בהתאם למצב הנוכחי, אלא ישאר התוכן הקודם (כי לא משוגר מידע חזרה לSubject). כמו כן כפתור הrestart מאפס את התהליך (נעשה בעזרת השיטה ()clickRestart הנמצאת בpackage "Model" במחלקה "MyArrayList" בשורה 71). אציין כי הוספתי כפתורים הטיפוסי המחלקות החדשים בpackage "View" במחלקה "MainView", וכן בController הוספתי רשימה מסוג MyArrayList בשורה 45. התפקוד של הצופה-מאזין נמצא בכפתורים הרלוווטיים בsetOnAction .
24. לא משוגר בחזרה לSubject מידע כלשהו.
25. מתועד לעיל.

נספח: הפרוייקט מקורס 10118:

פרויקט סיום בקורס "תכנות מונחה עצמים".

הנחיות טכניות להגשה

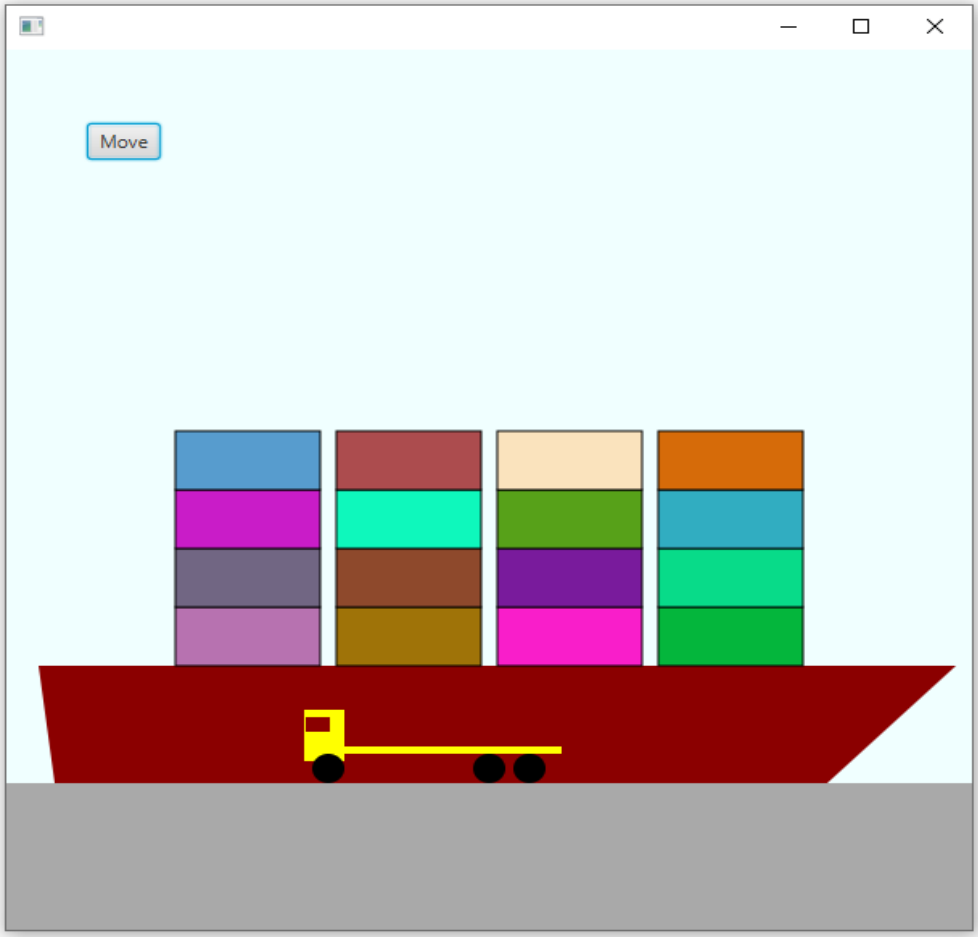
1. הגשת הפרוייקט בתיבת ההגשה במודל עד התאריך המצוין בתיבה.
2. ניתן להגיש את הפרוייקט בזוגות או ביחיד.
3. רק אחד מבני הזוג מעלה את הפרוייקט למודל

מה עליכם לעשות?

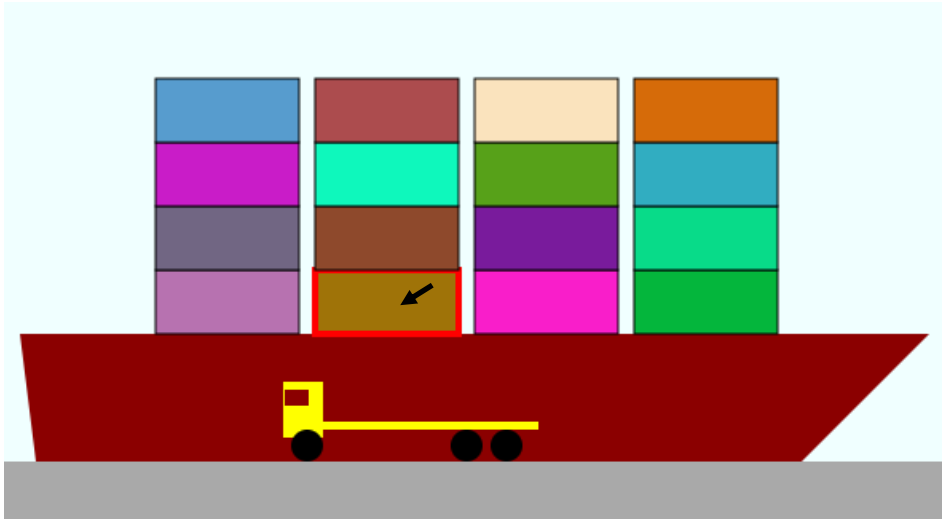
כתבו תכנית המשתמשת ב-javafx ובתבנית MVC לפתרון הבעיה המתאורת בהמשך. יש להשתמש בעקרונות של תכנות מונחה עצמים כפי שנלמד בכיתה.

תיאור הבעיה

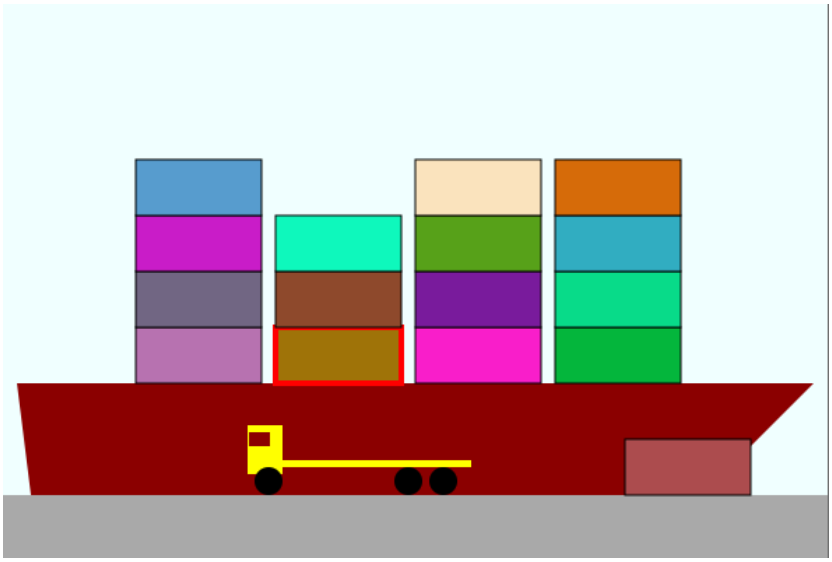
בנמל עוגנת ספינת מכולות ועל המזח נמצאת משאית אשר יכולה להעביר מכולה אחת בלבד למגרש לאחסון מכולות.

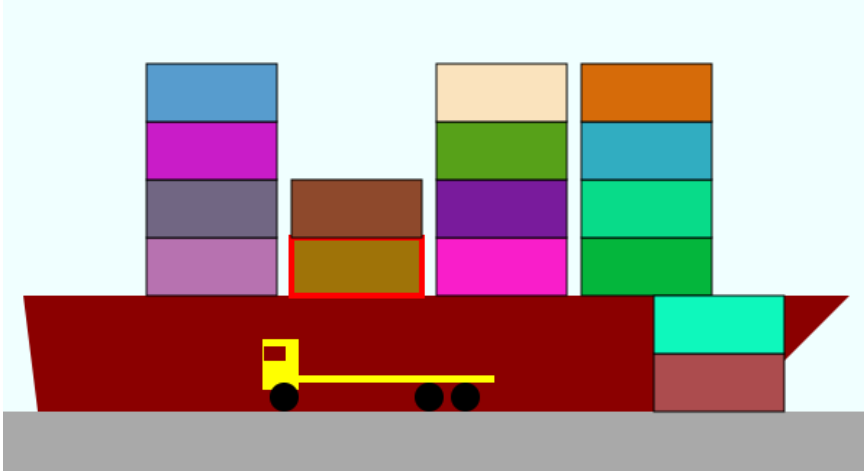


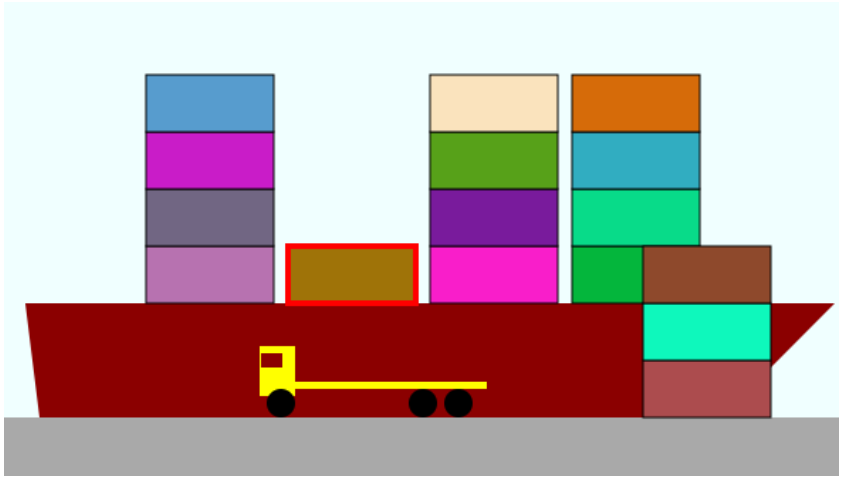
משתמש המערכת צריך לסמן מכולה (לאו דווקא עליונה) בעזרת קליק העכבר ולאחר מכן להתחיל ללחוץ על לחצן "Move" כדי לראות איך המערכת מוציאה את המכולה מהספינה למשאית. המנוף (שאין צורך לציירו) יכול להעביר בפעולה אחת מכולה אחת מסיפונה של הספינה אל המזח או על המשאית. ניתן לשים מכולה על מכולה אחרת. להלן סימון המכולה:



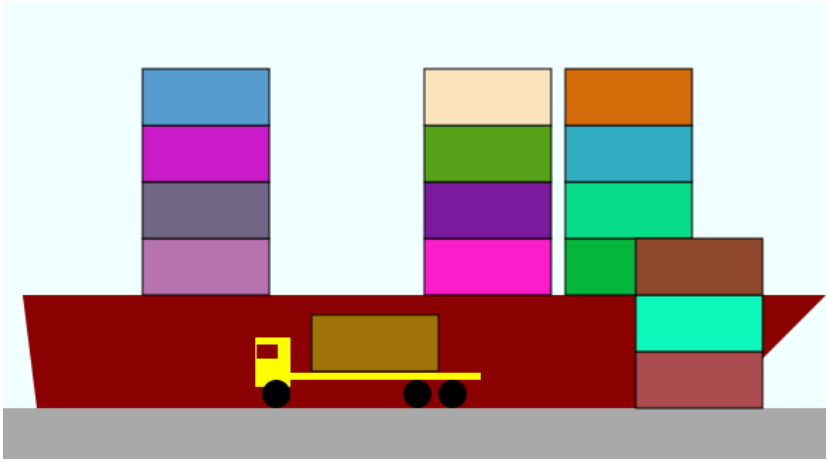
תמונות הבאות מציגות תוצאות של לחיצות עוקבות על לחצן "Move". שימו לב, כי המכולות נאספות על המזח בסדר הפוך מזה שהיה להן על הסיפון.



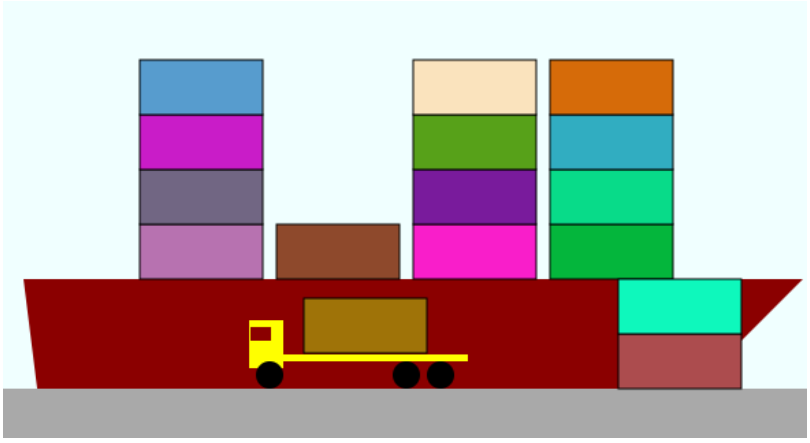


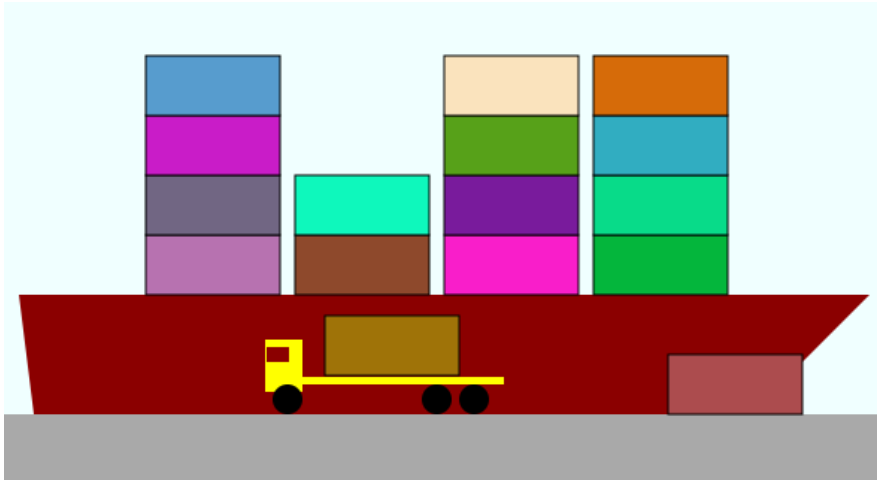


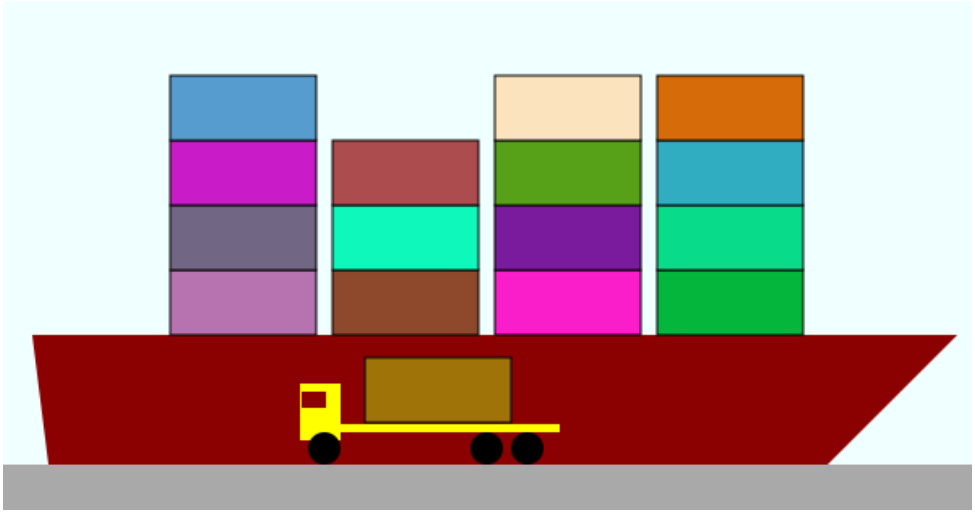
כאשר מגיע תורה של המכולה המסומנת על ידי המשתמש היא עוברת ישירות אל המשאית:



לחיצות נוספות על לחצן "Move" מחזירות את המכולות מהמזח חזרה אל הסיפון







בהצלחה!

נספח 2: המטלה הנוכחית בקורס 10119:

חלק 1: Collections, Genercis, Iterator, Lambda Expression

12. יש להוסיף אפשרות בתפריט, שהפעלתה תעביר את אחד מהמערכים של אובייקטים/ ArrayList הקיימים, לבחירתך, ל- Collection הכי מתאים, כך שבכל שלב אם נרצה להדפיס את תוכן ה- Collection הזה, נקבל תמיד סדר לפי מיון **יורד**, כאשר המיון נקבע לפי אחת התכונות באובייקט לבחירתך.

    יש לבחור תכונה שהיא לא מספר מזהה או משהו ייחודי לכל אובייקט, אלא תכונה עם ערך שחוזר על עצמו פה ושם באובייקטים השונים.

    יש לבצע כך, **שכל האיברים מהמערך המקורי יוכנסו תמיד**, אפילו אם כל התכונות שלהם זהות לחלוטין. יש להדפיס את תוכן ה- Collection בעזרת Iterator כפי שלמדנו בכיתה.

13. כעת, יש להוסיף אפשרות בתפריט, שהפעלתה תעביר את כל תוכן ה- Collection הקודם ל- Collection נוסף מתאים, כאשר הדגש יהיה כעת על המהירות (הוספה /חיפוש/ הסרה), ואין משמעות למיון כלשהו. בנוסף, ה- Collection הפעם לא יכניס איברים חדשים שהם כפילות עם מה שכבר נמצא בו. כפילות מוגדרת לפי שוויון מוחלט בין 2 תכונות לבחירתך.

    יש לבחור תכונות שהן לא מספר מזהה או צירוף ייחודי לכל אובייקט, אלא תכונות עם צמד ערכים שחוזרים על עצמם פה ושם באובייקטים השונים.

14. יש להדפיס את תוכן ה- Collection בעזרת Iterator, כפי שלמדנו בכיתה, ורוצים שההדפסה תהיה לפי סדר **ההכנסה** של האיברים ל- Collection החדש, כלומר לפי הסדר המקורי שהיה בעצם עם המיון, אבל בלי הכפילויות.

15. בקוד שאתה מוסיף, באם יש צורך, וניתן, אזי יש לעבוד עם Lambda Expression.

16. בקובץ ה- Known Issues, יש להוסיף הסבר, באיזה מערך / ArrayList מקורי השתמשתם מתוך הפרויקט, איפה הוא נמצא בקוד (שם הקובץ, מספרי שורות וכו'), ולפי איזה תכונה מתבצע המיון, ולפי איזה תכונות נקבעת הכפילות ב- Collection השני, **וכיצד להפעיל את שני הסעיפים הנ"ל דרך התפריט או דרך איזה כפתורים.**

חלק 2: ArrayList, Design patterns: Iterator, Observer

17. יש להוסיף אפשרות נוספת בתפריט, שבה ניצור מחלקה משלנו שתיקרא MyArrayList, שהיא תבצע בעצם את מה ש- ArrayList מבצעת באופן הכי בסיסי, כלומר תהיה לנו רק מתודה add. (אין צורך לרשת מ- Collection/List כמו ב- ArrayList המקורי)
18. יש להעביר את כל האיברים מ- ה- Collection של הסעיף הקודם (זה שבלי הכפילויות), אל MyArrayList, לפי הסדר שבו הם נמצאים ב- Collection.
19. בעזרת ה- design pattern של ה- Iterator, יש להוסיף למחלקה מימוש של Iterator, ואז להפעיל אותו בדיוק כמו שמפעילים Iterator רגיל של Java, כלומר להדפיס את תוכן ה- MyArrayList. (הפלט צריך להיות זהה לפלט של ה- Collection מהסעיף הקודם.)
20. יש להוסיף גם מימוש למתודה remove של ה- Iterator, ולהציג תפעול שלה.
21. כעת, יש להציג גם במקביל, אותו דבר בדיוק עם ArrayList של Java ועם Itrator מובנה של Java, ולוודא שקיבלנו אותן תוצאות.
22. עבור כל הנ"ל, בקובץ ה- Known Issues, יש להוסיף הסבר, **כיצד להפעיל את הסעיף הנ"ל דרך התפריט או דרך איזה כפתורים, ומה בדיוק מבצע ה- remove ו/או איך להפעיל אותו, הן במה שכתבת והן בהרצת 'הביקורת' של ה- Iterator המקורי של java.**
23. בשלב הזה, יש לעבוד עם ה- design pattern של Observer שלמדנו בכיתה.

    אין להשתמש במחלקות מוכנות של java כגון Observable וכך הלאה.

    יש להניח ש- MyArrayList הוא ה- Subject. כאשר אנחנו יוצרים Iterator חדש כפי שבנינו בנ"ל, ה- Subject יודיע ל- Listeners (Observers) שלו על כך שנוצר iterator חדש והוא מוכן לפעולה. יש לצור שתי מחלקות משלך, אחת של Button ואחת של Label, שהם יהיו ה- Listeners (Observers). יש להשתמש בירושה בהתאם.

    כעת (תוך שימוש ב- setVisible, setDisable וכו') : כאשר ה- Listeners מקבלים הודעה שנוצר ה- Iterator, ה- Label מופיעה, ומציגה צבע ירוק עם כיתוב מתאים, וגם ה- Button מופיע ומאפשר עצמו לפעולה, כך שעכשיו אם נלחץ עליו נקבל רק את **<u>הפעלת</u>** ה- Iterator הזה שכבר נוצר, של MyArrayList. (יש לוודא שמקבלים אותו פלט כמו קודם)
24. הערה : בשלב זה, אין צורך לשגר בחזרה ל- Subject מידע כלשהו.
25. עבור כל הנ"ל, בקובץ ה- Known Issues, יש להוסיף הסבר, כיצד להפעיל את הסעיף הנ"ל דרך התפריט או דרך איזה כפתורים, איך נקראות המחלקות של ה- Listeners, ואיך להשוות עם הפלט הקודם.