הנדסת תוכנה – תרגיל בית 3

דגשים להגשת המטלה

1. תאריך הגשה: יום חמישי 12.01.2023 בשעה 23:59.
2. **הגשה בזוגות בלבד!**
3. הקוד חייב להיכתב בהתאם למוסכמות כתיבת הקוד בקורס כולל תיעוד כנדרש. קוד שלא עומד בדרישות יגרור הורדת ניקוד. ניתן למצוא את קובץ מוסכמות הקידוד באתר הקורס תחת הלשונית "קבצי עזר".
4. ההגשה מתבצעת ב-Moodle באזור המיועד על ידי אחד מהשותפים, לאחר יצירת קבוצה.
5. ניתן להגיש את התרגיל לכל היותר עד 48 שעות לאחר מועד ההגשה ללא הורדת ניקוד. לאחר 48 שעות תיבת ההגשה תיסגר ולא יהיה ניתן להגיש את התרגיל כלל.
6. פורמט הגשת התרגיל נמצא בקובץ ההנחיות ב-Moodle. **כל חריגה מפורמט זה תגרור ציון אפס.**

מטרת התרגיל

עבודה עם מבני נתונים, העתקת עצמים ואיטרטורים.

הכנות טרם תחילת התרגיל

1. פתיחת פרויקט ג'אווה חדש.
2. הורדת קבצי התרגיל, והעתקת הקבצים Main.java ו-Queue.java **בלבד** אל תוך תיקיית ה-src.

הוראות כלליות

1. יש לבדוק שהקוד עובר הידור (קומפילציה) ללא שגיאות.
2. מומלץ להריץ את התוכנית עם מספר קלטים שונים ולחשוב על מקרי קצה אפשריים.
3. מומלץ לחזור על התרגולים וההרצאות וכן להיעזר באינטרנט.
4. יש להשתמש בגרסה 9.0.4 של ג'אווה בעת פתרון התרגיל.
5. מומלץ להשתמש ב-Git במהלך כתיבת התרגיל.

הוראות הגשה

1. יש למלא אחר הוראות ההגשה בהתאם למסמך הדרישות "הנחיות כלליות לפתרון והגשת תרגילי הבית" אשר מופיע באתר הקורס.
2. הגשה אלקטרונית **בלבד** דרך אתר הקורס ב-moodle. ההגשה תכלול קובץ ה-zip בלבד בפורמט HW3_<id1>_<id2> כאשר <id1> ו-<id2> הם תעודות הזהות של המגישים. פירוט נוסף נמצא בסוף המסמך.
3. ההגשה מתבצעת על ידי אחד מבני הזוג לאחר שיצר קבוצה ובן הזוג השני הצטרף אליה.
4. תרגיל בית שלא יוגש על פי הוראות ההגשה – **לא ייבדק**.
5. יש להקפיד על יושרת הכנת התרגיל וההגשה.
6. יש לוודא כי הקוד מתקמפל – קוד אשר לא יעבור הידור יקבל ציון אפס.
7. אין צורך להגיש את קובץ הפלט אשר ניתן כחלק מתרגיל זה.

חלק א – מבני נתונים

בהרצאות ובתרגולים דנו במספר מבני נתונים. ראינו כיצד מבני הנתונים השונים ממומשים בג'אווה, ואף מימשנו את חלקם בעצמנו.

בחלק זה של התרגיל תממשו מבנה נתונים נוסף אותו פגשנו – תור.

תור הינו מבנה נתונים אשר עובד בשיטת FIFO: First In First Out. כלומר – האיבר הראשון שנכנס לתור יהיה הראשון לצאת ממנו. כאשר איבר מוכנס לתור, הוא נכנס אל זנב התור (rear), וכאשר איבר מוצא מן התור הוא יוצא מראש התור (front).

אחד מן הקבצים אשר ניתנו כחלק מהתרגיל הוא קובץ בשם Queue.java. בקובץ זה מוגדר ממשק אשר מייצג תור כללי. בממשק מוגדרות מספר פעולות אשר כל תור צריך לממש:

- `enqueue`: פעולה אשר מקבלת איבר ומוסיפה אותו אל זנב התור.
- `dequeue`: פעולה אשר מוציאה איבר מראש התור ומחזירה אותו.
- `peek`: פעולה אשר מחזירה את האיבר אשר נמצא בראש התור, מבלי להוציאו.
- `size`: פעולה אשר מחזירה את מספר האיברים בתור.
- `isEmpty`: פעולה אשר בודקת האם יש איברים בתור או לא ומחזירה את התוצאה.
- `clone`: פעולה אשר מבצעת העתקה עמוקה של התור.

פרטים נוספים לגבי הממשק:

- הממשק הינו גנרי, והטיפוס הגנרי מוגבל מלמעלה על ידי הממשק `Cloneable`.
- ממשק זה מרחיב את הממשק `Iterable` על מנת לאפשר מעבר על איברי התור.
- ממשק זה מרחיב את הממשק `Cloneable` על מנת לאפשר העתקה עמוקה של התור.

עליכם ליצור מחלקה גנרית בשם `LinkedListQueue` אשר תממש את הממשק `Queue` שמוגדר בקובץ שקיבלתם.

התור אותו תממשו ייעזר ברשימה מקושרת לצורך שמירת הנתונים. במידה והתור ריק ומנסים להוציא ממנו איבר או להציץ באיבר אשר נמצא בראשו, תיזרק חריגה בלתי מסומנת בשם `EmptyQueueException`. חריגה זו איננה מוגדרת בג'אווה ועליכם להגדירה באופן עצמאי.

נוסף על כך, על מנת לשמור על יעילותו של התור, עליכם לממש את פעולות ההכנסה וההוצאה מן התור ללא שימוש בלולאות ו/או בפעולות אשר משתמשות בלולאות.

<u>העתקת התור</u>

על התור לתמוך בפעולה העתקה אשר מבצעת העתקה עמוקה של התור. על מנת להבטיח כי ניתן יהיה להעתיק את האיברים אשר נמצאים בתור, הטיפוס הגנרי של התור מוגבל מלמעלה על ידי הממשק `Cloneable`.

עליכם לממש במחלקה `LinkedListQueue` את הפעולה `clone` תוך שימוש ב-Covariant Return Type. יש להשתמש בבלוק `try-catch` בעת כתיבת פעולת ה-`clone`. במידה ונזרקת חריגה בעת ההעתקה יש להחזיר את הערך `null`. אין לזרוק חריגה בפעולות ה-`clone`.

הכוונה: תוכלו להיעזר בפעולה `invoke` לצורך העתקת איברי התור.

מעבר על איברי התור

על התור אותו תממשו לתמוך במעבר על איבריו באמצעות לולאת foreach. לשם כך, עליכם להגדיר במחלקת התור מחלקה פנימית בשם QueueIterator אשר מממשת את הממשק הגנרי Iterator ומייצגת איטרטור שישמש לצורך מעבר על איברי התור. המעבר על האיברים יתבצע לפי סדר התור, החל מהאיבר הנמצא בראש התור ועד לאיבר אשר נמצא בזנב. בעת מעבר על איברי התור אין לשנות את התור, גם לא באופן זמני.

הנחיות לחלק א

- בחלק זה אין להשתמש באף מבנה נתונים אשר מוגדר בג'אווה, כולל מערך.
- אין לשנות את הממשק Queue אשר קיבלתם ואין לשנות את האופן בו מוגדרות הפעולות שבו.
- שימו לב כי לא ניתן להניח דבר על הטיפוסים אשר נשמרים בתור, פרט לכך שהם ניתנים לשכפול. כחלק מהבדיקה של המחלקה, מוגדרת בקובץ Main.java מחלקה בשם MyCloneable. מחלקה זו נוצרה אך ורק על מנת שתוכלו לבדוק את הקוד שלכם, וייתכן מאוד כי במהלך בדיקת התרגיל מחלקה זו תוחלף במחלקה אחרת.

חלק ב – רשימת מטלות

תיאור המשימה

עקב העומס של סטודנטים רבים במהלך הסמסטר, נשיא הטכניון ביקש מכם לסייע לסטודנטים בניהול המטלות שעליהם לבצע.

לשם כך הוא ביקש מכם לבנות מערכת אשר תייצג את רשימת המטלות שמוטלות על הסטודנטים.

הנשיא סיפר לכם כי כל רשימת מטלות מורכבת ממספר לא מוגבל של מטלות, וכן כי כל מטלה מורכבת ממחרוזת אשר מתארת אותה ומתאריך הסיום שלה. עוד הוא הוסיף כי לאחר שמטלה נוצרת לא ניתן לשנות את התיאור שלה.

לצורך נוחות, הנשיא ביקש מכם שתהיה באפשרות הסטודנטים היכולת להציג את כל המטלות שלהם באופן הבא:

[(description, dueDate), (description, dueDate), …, (description, dueDate)]

כאשר description הינו התיאור של כל מטלה ו-dueDate הינו תאריך הסיום של כל מטלה בפורמט DD.MM.YYYY.

יתרה מכך, הנשיא ביקש כי סטודנטים יוכלו לסרוק את רשימת המטלות שלהם בשתי צורות שונות:

- מעבר על כל המטלות.
- מעבר רק על המטלות שתאריך הסיום שלהן הוא לא מאוחר יותר מתאריך נתון.

בכל סריקה של רשימת המטלות נעבור על כל המטלות אשר עומדות בתנאי הסריקה, לפי סדר עולה של תאריך הסיום שלהן. במידה וישנן מטלות בעלות אותו תאריך סיום, יש לעבור עליהן לפי סדר אלפביתי של תיאורן.

בנוסף, הנשיא רצה שהסטודנטים יוכלו לבצע העתקה עמוקה של רשימת המטלות שלהם, תוך שימוש ב-Covariant Return Type על מנת לחסוך לסטודנטים את הצורך להמיר את רשימת המטלות בעת העתקתה.

נוסף על כך, הנשיא רוצה לאפשר לסטודנטים לבדוק האם רשימת המטלות שלהם ושל חבריהם שוות אחת לשנייה. לצורך כך, הוא הגדיר כי שתי רשימות מטלות שוות זו לזו אם ורק אם הן מכילות את אותן מטלות והתאריכים לסיום של אותן המטלות שווים זה לזה בשתי הרשימות בהתאמה. עוד הנשיא הבהיר כי אין חשיבות לסדר ההופעה של המטלות ברשימות של הסטודנטים בהגדרת השוויון.

מימוש המשימה

עליכם להיענות לכל בקשותיו של נשיא הטכניון, ולהכין בעבורו את מערכת המטלות על מנת שיוכל להפיצה לשימוש בקרב כל הסטודנטים בטכניון כבר במהלך הסמסטר הנוכחי.

לשם כך, התחילו בלהגדיר מחלקה בשם `Task` אשר מייצגת מטלה בודדת ומחלקה בשם `ToDoList` אשר מייצגת רשימת מטלות. בכל מחלקה יש לכלול את התכונות הנדרשות לה לפי התיאור של הנשיא.

במחלקה `ToDoList` יש להוסיף פעולה בשם `addTask` אשר מקבלת מטלה ומוסיפה אותה לרשימת המטלות במידה ולא קיימת ברשימה מטלה נוספת בעלת אותו תיאור. במידה וכן קיימת מטלה בעלת תיאור זהה יש לזרוק חריגה בשם `TaskAlreadyExistsException`. חריגה זו הינה חריגה בלתי מסומנת אותה עליכם להגדיר.

הצגת המטלות

לצורך הצגת המטלות, עליכם לדרוס את הפעולה `toString` במחלקות `Task` ו-`ToDoList` כך שתתקבל הצגה של כלל המטלות בפורמט אותו הנשיא הגדיר. שימו לב שהמטלות מוצגות לפי סדר ההוספה שלהן.

העתקת המטלות

על מנת לאפשר העתקה עמוקה של רשימת מטלות, עליכם לממש במחלקות Task ו-ToDoList את הממשק Cloneable ולדרוס בהן את הפעולה clone, תוך שימוש ב-Covariant Return Type בשתי המחלקות. יש להשתמש בבלוק try-catch בעת כתיבת פעולות ה-clone. במידה ונזרקת חריגה בעת ההעתקה יש להחזיר את הערך null. אין לזרוק חריגה בפעולות ה-clone.

הערה: שימו לב בכל מחלקה לתכונות אשר ניתנות לשינוי (mutable).

שוויון רשימות מטלות

על מנת לאפשר השוואה בין שתי רשימות מטלות יש לדרוס את הפעולה equals במחלקות Task ו-ToDoList כך שיתקיים יחס השוויון אותו הנשיא הגדיר. בנוסף, יש להקפיד על שאר הדרישות אשר קיימות עבור יחס שוויון ופעולת ה-equals.

נוסף על כך, עליכם לדרוס בשתי המחלקות את פעולת ה-hashCode בהתאם לדרישות אשר קיימות במפרט הפעולה. בעת דריסת הפעולה אין להחזיר ערך קבוע: ערך הגיבוב של כל עצם צריך להיות תלוי בערכי התכונות של העצם.

סריקה של רשימת מטלות

הנשיא ביקש שסטודנטים יוכלו לסרוק את רשימת המטלות שלהם בשתי צורות סריקה שונות. לשם כך נרצה להגדיר איטרטור אשר נוכל לבצע באמצעותו את הסריקה. עליכם ליצור במחלקהToDoList מחלקה פנימית בשם ToDoListIterator אשר מממשת את הממשק Iterator<Task>. יש לממש במחלקה זו את הפעולות hasNext ו-next בהתאם לסוג הסריקה.

לאחר מכן, עליכם ליצור ממשק בשם TaskIterable אשר מרחיב את הממשק Iterable<Task>. בממשק זה תגדירו פעולה בשם setScanningDueDate אשר מקבלת את התאריך שמשמש בעת מעבר על רשימת המטלות. במידה וערכו של הפרמטר שהועבר הינו null, המעבר על המטלות יתבצע לפי סוג הסריקה הראשון. אחרת, תאריך זה משמש בתור התאריך הנתון בסוג הסריקה השני.

לבסוף, ממשו במחלקה ToDoList את הממשק TaskIterable על מנת לאפשר סריקה נוחה של רשימת המטלות על ידי לולאת foreach. שימו לב כי מימוש הממשק כולל מימוש של הפעולה setScanningDueDate. התאריך אשר מועבר לפעולה זו ישמש בקביעת סוג הסריקה הנוכחי (ובתור התאריך הנתון של סוג הסריקה השני) עד לקריאה הבאה לפעולה. ניתן להניח כי התאריך אשר משמש לסריקה של רשימת המטלות וכן רשימת המטלות עצמה לא ישתנו במהלך סריקה של הרשימה. בנוסף, סוג הסריקה ההתחלתי של רשימת מטלות הינו סוג הסריקה הראשון.

הכוונות:

- על מנת למיין רשימה לפי פעולה מותאמת אישית ניתן להיעזר בפעולה Comparator.comparing.
- על מנת לבצע מיון משני של רשימה לפי פעולה מותאמת אישית ניתן להיעזר בפעולה thenComparing.

הנחיות לפתרון

- בעת פתרון התרגיל ניתן להגדיר מחלקות נוספות.
- בכל מחלקה יש לכלול את התכונות המתאימות לה, ולספק בעבורן פעולות get ו-set **במידת הצורך**.
- בכל מחלקת חריגה אותה אתם יוצרים יש להגדיר את שלושת הבנאים הסטנדרטיים עליהם דיברנו בתרגולים.
- במהלך כתיבת הקוד, יש לשמור על עקרונות תכנות נכונים, כפי שנלמד בקורס.
- בעת פתרון התרגיל ניתן ואף מומלץ להגדיר קבועים ולא להשתמש במספרי קסם.
- במהלך פתרון התרגיל, יש להקפיד על הרשאות הגישה השונות בהתאם לנלמד בקורס.
- יש להקפיד על מתן שמות משמעותיים למשתנים, לפעולות, למחלקות ולממשקים.
- יש ליצור כל מחלקה וממשק בקובץ נפרד, פרט למחלקות הפנימיות. בנוסף, יש להגדיר את הרשאת הגישה של כלל המחלקות והממשקים כפומבית.
- יש לתעד את כל הפעולות והמחלקות אותן אתם מגדירים בעזרת שימוש ב-JavaDoc בהתאם לקובץ מוסכמות התיעוד אשר מופיע באתר הקורס. בנוסף, יש לתעד שורות קוד אשר עשויות להיות קשות להבנה. עבור פעולות שזורקות חריגה מכל סוג (מסומנת או בלתי מסומנת) יש לציין זאת בתיעוד על ידי פסקת `@throws` ביחד עם הסבר על מתי החריגה תיזרק.

הגשת התרגיל

לפני הגשת התרגיל, עליכם ליצור קבצי html מתוך תיעוד ה-JavaDoc שכתבתם בקוד. ניתן ליצור את הקבצים בעזרת ה-IntelliJ על ידי לחיצה על כרטיסיית ה-Tools ובחירת האפשרות Generate JavaDoc. בעת יצירת המסמכים, יש לבחור את הרשאת הגישה private על מנת שכל המחלקות והפעולות (משני חלקי התרגיל) יופיעו במסמכים. את כל הקבצים שנוצרים יש לשמור בתיקייה בשם JavaDoc אשר ממוקמת בתוך תיקיית ה-src.

כפי שהוזכר, קובץ ההגשה הסופי הינו קובץ zip בודד. שם הקובץ מפורט בתחילת המסמך ובמסמך ההוראות הכללי. קובץ ה-zip יכלול תיקייה בודדת בשם src בה יימצאו כל קבצי הקוד אותם כתבתם ותיקייה נוספת בשם JavaDoc אשר תכיל את כל קבצי התיעוד.

הרצת התוכנית וביצוע בדיקות

בקובץ Main.java קיימות מספר מחלקות אשר משמשות לבדיקת הקוד שכתבתם.

להזכירכם, חלק מן הבדיקה נעשה באופן אוטומטי, ולכן אין לשנות את התוכן של הקובץ, ובפרט אין לשנות את פעולות ההדפסה המתבצעות בו.

מצורף לתרגיל זה קובץ הפלט HW3_output.txt, על מנת שתוכלו לבצע את ההשוואה באופן ידני (או על ידי שימוש ב-DiffMerge).

בהצלחה