פיתוח תוכנה מתקדם 2 – סמסטר ב' מועד א'

תזכורת: כתובת מערכת הבדיקות: https://cktest.cs.colman.ac.il/. שם הקורס PTM2, מועד א'. לאחר הורדת המבחן ממערכת הבדיקות. העתיקו את כל קובצי ה Java לתוך הפרויקט ב package בשם test.

במבחן זה 3 שאלות, חובה לענות על כל 3 השאלות ולהגיש למערכת הבדיקות במוד הגשה סופית לפני סוף המבחן.

שאלה 1 - תכנות מקבילי באמצעות ת'רדים (35 נק')

עליכם לממש Active Object בקובץ PTasker.java. ה Active Object צריך לתמוך בשתי המתודות הבאות:

א. `<V> Future<V> apply(List<V> buff, V identity, BinaryOperator<V> bo)`

המתודה תריץ **ברקע** משימה שמתאימה לפסאודו-קוד הבא:

```
V result = identity;
for (V element : buff)
    result = bo(result,element)
return result;
```

לדוגמה, ב MainTrain1 ההפעלה הבאה מסכמת את כל איבריי הרשימה:

```
PTasker pt=new PTasker();
Future<Integer> f=pt.apply(buff, 0, (x,y)->x+y);
```

שכן הערך ההתחלתי הוא 0 והאופרטור הבינארי מגדיר לכל זוג איברים להחזיר את הסכום שלהם.

מכיוון שהמתודה apply מריצה את המשימה בצורה אסינכרונית, עליה להחזיר את התוצאה בתוך Future<V> (הטיפוס האמתי, לא חיקוי).

ב. המתודה close() – לא תאפשר כניסה של משימות חדשות, תריץ את כל המשימות שכבר קיימות ב Active Object ולאחריהן תסגור את הת'רד.

טיפ: במקום לעבוד קשה, ניתן עם הטכניקות המיוחדות שלמדנו לממש כל אחת מהמודות בשורת קוד אחת בלבד!

מוד האימון ניתן ב MainTrain1.java. שאלה זו תיבדק בצורה אוטומטית בלבד, ולכן חובה שתתקמפל ותרוץ ללא שגיאות. מוד ההגשה זהה למוד האימון.

שאלה 2 - Singleton גנרי (35 נק')

תזכורת:

משתנה מסוג Class<V> מכיל מידע אודות המחלקה מטיפוס V. האמצעות המתודה getName ניתן לחלץ מחרוזת עבור שם המחלקה, ובאמצעות המתודה newInstance ניתן ליצור מופע של המחלקה.

בקובץ Singleton.java עליכם ליצור Singleton גנרי שיכול לקחת כל טיפוס V ולנהל אותו כ Singleton.

עליכם לממש את המתודה:

```
public static <V> V getInstance(Class<V> c) throws Exception
```

כך שבהינתן Class<V> היא תשתמש בטכניקה של double check locking (כן, למרות שהיא אינה מושלמת) על מנת ליצור מופע מסוג V בצורה שהיא Thread safe.

תשימו לב שהמתודה היא זו שתלויה בטיפוס הפרמטרי V ולכן בכל הפעלה שלה ניתן לבחור טיפוס אחר.

לדוגמה ב MainTrain2 תוכלו לראות שיש כמה ת'רדים שמנסים בו"ז ליצור מופעים של הטיפוסים TestA, TestB, TestC. אולם, הצפייה היא שמכל טיפוס נוצר רק אובייקט אחד.

```
TestA a=Singleton.getInstance(TestA.class);
TestB b=Singleton.getInstance(TestB.class);
TestC c=Singleton.getInstance(TestC.class);
```

שאלה זו אף תיבדק ידנית. עם זאת משקל קוד מתקפל ורץ ללא שגיאות ריצה = 10 נק'.

מוד ההגשה דומה למוד האימון

שאלה 3 - אופטימיזציות קוד (30 נק')

בקובץ BadCode.java מצויה הפונקציה dists אשר בהינתן מערך של int-ים היא מחזירה מערך חדש באותו האורך. יהי x_i איבר במערך החדש במקום ה i, ו o_i איבר במערך המקורי במקום ה i, ו $\overline{o}$ ממוצע איברי המערך המקורי, אז כל x_i שווה לריבוע ההפרש בין o_i ל $\overline{o}$.

$$(\forall i, x_i \leftarrow (o_i - \overline{o})^2)$$

האימפלמנטציה בקובץ זה היא לא יעילה. עליכם לערוך את הקוד בקובץ GoodCode.java כך שהאימפלמנטציה תהיה לפחות פי 200 יותר מהירה.

כמובן, יש להחזיר את התוצאה הנכונה.

מוד האימון **זהה** למוד ההגשה והוא ניתן בקובץ MainTrain3.java.

תחילה מיצרים קלט של מערך אקראי.

לאחר מכן מתבצעת מדידת זמן של BadCode ושל המימוש שלכם ב GoodCode.

קוד שהוא לפחות פי 200 יותר מהיר יקבל את מלוא 30 הנק'

קוד שהוא פחות מפי 200 יותר מהיר יקבל את החלק היחסי (למשל קוד שהוא פי 100 מהיר יותר יקבל 15 מתוך 30 הנק').

שאלה זו אף תיבדק ידנית. עם זאת הקנס על קוד עם שגיאת ריצה או קומפילציה הוא 5 נק'.

הגשה

עליכם להיכנס למערכת הבדיקות בכתובת: https://cktest.cs.colman.ac.il/ ולהגיש ל PTM2 ומועד א' את הקבצים Ptasker.java, Singleton.java, GoodCode.java

בכל הגשה יש להגיש את כל הקבצים (ולהתייחס לפלט רק של השאלות שעניתם עליהן)

ניתן להגיש במוד אימון ובמוד הגשה כמה פעמים שתרצו עד לסוף המבחן.

בסוף המבחן יש להגיש במוד הגשה ואז במוד הגשה סופית. אחריה תקבלו מס' אסמכתא בין 4 ספרות. לאחר הגשה במוד זה לא תוכלו להגיש יותר.

בהצלחה!