פיתוח תוכנה מתקדם 2 – מועד א'

תזכורת: כתובת מערכת הבדיקות: https://cktest.cs.colman.ac.il/. שם הקורס PTM2_test, מועד א'. לאחר הורדת המבחן ממערכת הבדיקות. העתיקו את כל קובצי ה Java לתוך הפרויקט ב package בשם test.

שאלה 1 - תכנות מקבילי באמצעות ת'רדים (30 נק')

א. עבודה עם Future (12 נק')

בקובץ Q1a.java עליכם לממש את המתודה threadIt כך שבהינתן פונקציה f שמחזירה ערך מטיפוס פרמטרי V, המתודה threadIt תריץ את f בת'רד נפרד ותחזיר את הערך V לתוך אובייקט מסוג Future<V>.

טיפ: מותר ואף רצוי להשתמש בספריות קוד קיימות של Java.

מוד האימון ניתן ב MainTrain1a.java

ב. Active Object (18 נק')

בקובץ Q1b.java עליכם לממש Active Object.

- במתודה push נזריק לו משימות מסוג Runnable.
- בת'רד נפרד ברקע, ה Active Object שלנו יריץ את המשימות בזו אחר זו (באותו הת'רד).
- באמצעות המתודה close נבצע יציאה מסודרת של ה Active Object שכוללת סגירת הת'רד שפתחנו. עליכם להריץ את כל המשימות שניתנו לפני הקריאה ל close. לאחר הקריאה ל close אין לקבל משימות חדשות.

<u>אין להיעזר בשאלה זו במשתנה מסוג ExecutorService או כל Thread Pool מוכן אחר.</u>

מוד האימון ניתן ב MainTrain1b.java. שאלה זו תיבדק אוטומטית, ולכן חובה שתתקמפל ותרוץ ללא שגיאות.

שאלה 2 - תכנות מקבילי (20 נק')

תכנות מקבילי באמצעות הפרד ומשול (20 נק')

בקובץ Q1c.java עליכם לממש את הפונקציה recThis. בהינתן הפרמטרים:

- ExecutorService
- מערך של אובייקטים מהטיפוס הפרמטרי T. ניתן להניח כי גודלו הוא 2^k כלשהו.
- אינדקסים start, end המציינים את תאי המערך מהאינדקס **start** ועד לאינדקס **end-1**.
- BinaryOperator<T> (זה functional interface עבור פונקציה משני T-ים ל T)

הפונקציה recThis תפעל באופן הרקורסיבי הבא:

- בכל שלב היא תעבוד על החצי השמאלי והימני של המערך עד שתגיע לגודל של שני תאים.
- אז היא תפעיל את האופרטור הבינארי ותקבל תוצאה מסוג T.
- ביציאה מהרקורסיה היא תפעיל את האופרטור הבינארי על התוצאות שחזרו מתת-המערך השמאלי ומתת-המערך הימני. כך, לבסוף נקבל תוצאה מסוג T.
- מכיוון שהחישוב על תת-המערך השמאלי אינו תלוי בחישוב של תת-המערך הימני, נבצע את החישוב של תת המערך-השמאלי **במקביל**.

מוד האימון ניתן בקובץ MainTrain1c.java. השתמשנו בפונקציה לעיל כדי לחשב את הערך המקסימאלי במערך.

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

שאלה 3 - אופטימיזציות קוד (50 נק')

הגדרה: מרחק H בין שתי מחרוזות הוא סכום הפעמים שנמצאו תווים שונים בין שתי המחרוזות באותו האינדקס.

דוגמאות:

- H("abc","abc")=0 – כי כל התווים שונים
- H("Abc","abc")=1 – כי A שונה מ a באינדקס הראשון
- H("abc","acb")=2 – כי יש שני תווים שנמצאים באותו המיקום אך שונים זה מזה
- H("abc","xyz")=3 – כי כל התווים שונים (והמחרוזת באורך 3)
- H("ab","abcd")=2 – אורך שונה. התווים cd נמצאים במחרוזת השנייה אך לא בראשונה

בקובץ Q2bad.java נתונה הפונקציה findMinH **<u>שאינה יעילה.</u>** בהינתן רשימה של מחרוזות היא מחפשת ומחזירה את המרחק המינימאלי שנמצא בין אי אלו שתי מחרוזות שברשימה.

עליכם לממש את הפונקציה הזו מחדש בקובץ Q2.java כך שתהיה יעילה ככל הניתן.

ככל שהקוד שלכם יהיה יותר יעיל ביחס ל Q2bad כך תקבלו יותר נקודות.

כדי לקבל את מלוא 50 הנק' על הקוד שלכם לרוץ לפחות פי 50 יותר מהר מהמימוש של Q2bad.

כמובן, יש להחזיר את התוצאה הנכונה.

<u>טיפ: הימנעו ככל הניתן מחיפושים שברור שהם מיותרים.</u>

מוד האימון **זהה** למוד ההגשה והוא ניתן בקובץ MainTrain2.java.

תחילה מיצרים קלט של מחרוזות אקראיות מעל {0,1}. שימו לב שיתכנו מחרוזות באורך שונה.

לאחר מכן מתבצעת מדידת זמן של Q2bad ושל המימוש שלכם ב Q2.

ע"פ היחס בין זמני הריצה נקבע הניקוד לשאלה.

שאלה זו אף תיבדק ידנית. עם זאת משקל קוד מתקפל ורץ ללא שגיאות ריצה = 5 נק'.

הגשה

עליכם להיכנס למערכת הבדיקות בכתובת: https://cktest.cs.colman.ac.il/ ולהגיש ל PTM2_test ומועד א' את הקבצים Q1a.java, Q1b.java, Q1c.java, Q2.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 סופית. אחריה תקבלו מס' אסמכתא בין 4 ספרות שאותו יש להציג לבוחנות. לאחר הגשה במוד זה לא תוכלו להגיש יותר. בהצלחה!