פיתוח תוכנה מתקדם 2 – סמסטר ב' מועד ב' תשפ"ג

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

במבחן זה 4 שאלות, חובה לענות על כל 4 השאלות ולהגיש למערכת הבדיקות במוד הגשה סופית לפני סוף המבחן. כל השאלות נבדקות אוטומטית בלבד.

שאלה 1 – Active Object גנרי (25 נק')

ברצוננו לכתוב Active Object גנרי – כלומר הוא יוכל לעטוף כל אובייקט רגיל, ולאפשר להריץ את המתודות שלו ע"פ התבנית של Active Object.

למשל לו היו לנו הטיפוסים הזרים A ו B, יכלנו ליצור מופעים של GenericActiveObject כך:

```
GenericActiveObject gao;
gao=new GenericActiveObject(new A());
gao=new GenericActiveObject(new B());
```

נרצה להפעיל את המתודות של אותו האובייקט ע"פ שם המתודה כמחרוזת + אוסף הפרמטרים שיש להזין לה.

למשל אם בטיפוס B קיימת המתודה public void place(float[] a,int ind) אז ההפעלה תהיה כך:

```
gao=new GenericActiveObject(new B());
int ind=5;
float ar[]=new float[10];
gao.execute("place", ar,ind);
```

הגדרות:

- **אין לפתוח את הת'רד של ה Active Object סתם**. יש לשמור על עקרון של laziness – כלומר, לא נחזיק ת'רד לחינם כבר ביצירה של GenericActiveObject, אלא רק כאשר בוצעה קריאה ל execute בפעם הראשונה נפתח את הת'רד.
- ע"פ תבנית Active Object יש לפתוח **ת'רד אחד בלבד** – שיריץ את כל הבקשות לפי הסדר אחת אחרי השנייה ע"פ סדר הגעתן. כאשר אין משימות הת'רד נח, אך לא נסגר.
- משימות יוזנו ע"י **המתודה execute**. המתודה מקבלת את 2 הפרמטרים:
  - name – מחרוזת המהווה את שם המתודה של האובייקט הנעטף שאותה נרצה להפעיל
  - Object...args – סדרה של פרמטרים מסוג Object המהווים את הפרמטרים של אותה מתודה.
- בהינתן הפרמטרים לעיל המשימה תוזן לתור המשימות וכשיגיע תורה לרוץ היא תריץ את המתודה name עם הפרמטרים args מתוך האובייקט שה GenericActiveObject עטף.
- **המתודה shutdown** תגרום לכך שלא יתקבלו יותר משימות חדשות, כל משימה שכבר נמצאת בתור תרוץ, ואחרי כן הת'רד ייסגר.

טיפים:

- תוכלו להיעזר בממשק ParamRunnable שמגדיר את המתודה run שמקבלת Object...args
- מותר להשתמש ב ExecutorService
- שימו לב לתזכורת בסוף המבחן שמתארת כיצד ניתן לדעת אלו מתודות יש לאובייקט ולהריץ אותן.

בדיקה: במוד אימון הבדיקה בוחרת אקראית בין מחלקה A ל B ואילו במוד ההגשה ייתכן שבין מחלקות אחרות בכלל כדי לבדוק את הגנריות של הקוד שלכם.

- הבדיקה בודקת שלא נפתח ת'רד לאחר יצרה של GenericActiveObject
- בדיקה מריצה ע"י execute מתודות שונות ובודקת שהן רצו בת'רד אחר ובודקת גם את התוצאה
- לאחר shutdown נבדק שחזרנו למס' הת'רדים המקורי לפני היצירה של ה GenericActiveObject

שאלה 2 – threads (25 נק')

עליכם לממש את המחלקה Q2 בקובץ Q2.java באופן הבא:

- Q2 היא סוג של Recursive Task שמחזירה Integer
- בבנאי היא תקבל מערך של int-ים. ניתן להניח כי גודלו הוא חזקה של 2.
- הפעולה של ה recursive task היא לחשב את הערך המקסימלי במערך
- אולם עליה לעשות זאת ע"י הפרד ומשול, באמצעות Fork Join Pool.
  - בכל שלב של הרקורסיה הערך המקסימלי הוא המקסימום בין
    - הערך המקסימלי של חצי המערך השמאלי לבין
    - הערך המקסימלי של חצי המערך הימני.
  - את החצי הימני יש לחשב כמשימה חדשה של ה Fork Join pool במקביל לחצי השמאלי.

הבדיקה בודקת שאכן חוזר הערך המקסימאלי הרצוי ושמספר המשימות שנפתחו תואם את ההגדרה לעיל.

מוד האימון זהה למוד הבדיקה.

שאלה 3 – אופטימיזציות קוד (25 נק')

בקובץ MainTrain3.java נתונה פונקציה לא יעילה (badContains) אשר בהינתן מערך המכיל ערכים אקראיים arr, ובהינתן ערך val, הפונקציה מחשבת מהו האינדקס **האחרון** בו מופיע איבר במערך השווה ל val.

שני ערכים נחשבים כשווים אם הם בעלי מרחק שווה מ 0.

לכן, השוויון בפונקציה נמדד ע"פ המרחק מ 0 בעזרת הפונקציה dist (המחשבת מרחק אוקלידי חד-ממדי).

בקובץ Q3.java עליכם לממש את הפונקציה goodContains אשר מחשבת את אותו הדבר. אולם על הפונקציה הזו לרוץ לפחות פי 2 יותר מהר מהקוד הלא יעיל. עליכם להתייחס רק לקבועי הזמנים שבמערכת הבדיקות.

מוד האימון זהה למוד ההגשה. בבדיקה נבדוק שעבור מטריצה עם ערכים אקראיים הפונקציה הטובה מקבלת את אותן התוצאות אולם בזמן מהיר פי 2. הניקוד לבדיקה הוא ביחס ישיר למהירות הריצה של הקוד שלכם. ככל שהקוד שלכם קרוב יותר לפי 2 מהר יותר מהקוד הלא יעיל, כך ירדו לכם פחות נקודות.

שאלה 4 – אופטימיזציות קוד (25 נק')

הגדרה: יהי איבר במערך מספרים שערכו x. יהי A מספר האיברים במערך **שגדולים ממש** מ x. יהי B מספר האיברים במערך **שקטנים ממש** מ x. הערך x ייחשב כחציוני במערך אם ורק אם A=B.

בקובץ MainTrain4.java נתונה פונקציה לא יעילה (badCode) אשר בהינתן מערך arr וערך val היא מחזירה את בערך מוחלט את ההפרש בין val לבין הערך החציוני של המערך.

בקובץ Q4.java עליכם לממש את הפונקציה goodCode אשר גם היא מחשבת ערך זה, אולם במהירות הגדולה פי 10. עליכם להתייחס רק לקבועי הזמנים שבמערכת הבדיקות.

מוד האימון זהה למוד ההגשה.

בבדיקה נבדוק שעבור מערך באורך אי-זוגי עם ערכים אקראיים וזרים, הפונקציה הטובה מקבלת את אותן התוצאות אולם בזמן מהיר פי 10.

הניקוד לבדיקה הוא ביחס ישיר למהירות הריצה של הקוד שלכם. ככל שהקוד שלכם קרוב יותר לפי 10 מהר יותר מהקוד הלא יעיל, כך ירדו לכם פחות נקודות.

הגשה

עליכם להיכנס למערכת הבדיקות בכתובת: https://cktest.cs.colman.ac.il/ ולהגיש ל PTM2 ומועד ב' את הקבצים GenericActiveObject.java, Q2.java, Q3.java, Q4.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!

תזכורת – עבודה עם Class<?>

- בהינתן Object o ניתן לחלץ את המחלקה שלו ע"י o.getClass()
- בהינתן המחלקה ניתן לחלץ את רשימת המתודות שלה ע"י getMethods()
  - זו רשימה של אובייקטים מסוג Method
- בהינתן מתודה ניתן
  - לחלץ את שמה ע"י getName()
  - להריץ אותה מתוך אובייקט o כלשהו ע"י invoke(o, args)
    - כאשר args הם אוסף הפרמטרים השונים של אותה מתודה המיוצגים ע"י Object…args