מבוא לתכנות מונחה עצמים מטלה 4

חלק א

הערות:

- כדי לפתור את התרגיל יש צורך לתכנן היטב את הפתרון שלו - מומלץ מאוד לחפש חומר ודוגמאות ברשת.
- התרגיל הינו בזוגות או ביחידים – לא ניתן להגיש בשלשות.
- יש להוסיף העורות לכל מחלקה ולכל שיטה ולבנות **JavaDoc**

חלק א.1 – הבנת הבעיה

נניח שיש בידינו פונקציה (בשם f) שמקבלת ומחזירה מספר טבעי. הפונקציה לעיתים נתקעת, אבל ברוב הפעמים מחזירה תשובה נכונה אחרי זמן קצר (ראו דוגמא לפונקציה כזו במחלקה המצורפת Ex4_tester). לפיכך נרצה לבנות **מעטפת** לפונקציה כך שניתן יהיה לקרוא לה עם ערך של זמן מקסימאלי (max): אם פעולת הפונקציה נסתיימה במסגרת הזמן – המעטפת תחזיר (מיד) את הערך שחושב, ובמקרה שהפונקציה נתקעה על המעטפת לזרוק שגיאה (לאחר שלא הגיעה תשובה במשך max שניות).

חלק א.2– פתרון הבעיה

כתבו מחלקה בשם Ex4 שמייצגת מחלקה לחישוב מספרים ראשוניים, המחלקה בעלת השיטה שמקבלת מספר טבעי ומשך זמן, ומחשבת אם המספר הטבעי הוא ראשוני – כל עוד לא חלף הזמן שקצוב לחישוב, ברגע שהזמן חלף על הפונקציה לזרוק שגיאה:

public boolean isPrime(long n, double maxTime) throws RuntimeException

הערות חשובות:

- אין לשנות את המחלקה Ex4_tester המצורפת – יש להשתמש בפונקציה שלה לצורך חישוב הראשוניות.
- יש לתכנן ולממש את מטלה כך שתהיה יעילה ביותר – תגזול מינימום משאבי מחשב.

המחלקה המצורפת: Ex4_tester לבדיקת חלק ב:

```
public class Ex4_tester {
/** This class represents a basic implementation for Ex3testing file. */
  public static double ENDLESS_LOOP=0.4;
  public static void main(String[]args){
      Ex4 ex4=new Ex4();
      long n=33333331;
      boolean ans=ex3.isPrime(n,0.01);
      System.out.println("n="+n+" isPrime "+ans);
  }
/** DONOT change this function!,it must be used
 *  byEx4-isPrime(long,double)
 */
  public static boolean isPrime(long n){
      boolean ans=true;
      if(n<2)throw new RuntimeException("ERR: the parameter to the isPrime function
                                          must be >1 (got "+n+")!");
      int i=2;
      double ns=Math.sqrt(n) ;
      while(i<=ns&&ans){
            if (n%i==0) ans=false;
            i=i+1;
      }
      if(Math.random()<Ex4_tester.ENDLESS_LOOP)while(true);
      return ans;
  }
}
```

חלק ב'

1ב בחלק זה נכתוב תכנית שמקבלת רשימה של קבצי טקסט ממדפיסה מספר שורות בכל קובץ. התכנית צריכה לבנות thread לכל קובץ שאמור לחשב את מספר השורות של כל קובץ במקביל.

לשם כך כתוב מחלקה בשם LineCounter שמייצגת את ה- thread שמחשב את מספר שורת של הקובץ.
בנאי המחלקה אמור לקבל את שם הקובץ.

כתוב מחלקה בשם **Matala_4A** המכילה פונקציות הבאות:

- פונקציה סטטית היוצרת מספר נתון של קבצי טקסט. כל קובץ מכיל מספר אקראי של שורות, בכל שורה כתוב משפט אחת: "Hello World". לקבלת מספר אקראי יש להשתמש במחלקת Random של java (**import** java.util.Random;) המאפשרת לקבל **אותה סדרה של מספרים אקראיים** בהרצות שונות של התכנה. שם הקובץ File_i, (i=1,…n).

  הפונקציה מקבלת מספר שלם n המייצג מספר קבצים. הפונקציה מחזירה מערך של שמות הקבצים.

  public static String[] createFiles(int n)

- פונקציה שמקבלת שמות הרבצים ומוחקת אותם:

  public static void deleteFiles(String[] fileNames)

- פונקציה **public static void countLinesThread(int numFiles)**

  הפונקציה מקבלת מספר קבצים כארגומנט.
  הפונקציה יוצרת קבצים, מפעילה עבור כל קובץ את ה-thread ומדפיסה את המספר הכולל של השורות בכל הקבצים. הפונקציה גם צריכה להדפיס את זמן הריצה של ה-threads (לא כולל יצירת ומחיקת הקבצים) בסוף הפונקציה צריכה למחוק את כל הקבצים.

2ב כתוב פונקציה **public static void countLinesOneProcess(int numFiles){**
המחשבת את המספר הכולל של השורות ללא שימוש ב-threads ,
הפונקציה יוצרת קבצים, קוראת קבצים אחד אחרי השני, מדפיסה את המספר הכולל של השורות ואת הריצה (לא כולל יצירת ומחיקת הקבצים).
בסוף הפונקציה צריכה למחוק את כל הקבצים.

תשווה את זמני הריצה של שתי השיטות!

3ב דוגמה להרצת התכנה:

```
public static void main(String[] args) {
        int num = 1000;
        countLinesThread(num);
        countLinesOneProcess(num);
}
```

פלט:

```
num = 1000
threads time = 5685 ms, lines = 508153
linear time = 7504 ms, lines = 508153

num = 2000
threads time = 10937 ms, lines = 1012396
linear time = 15927 ms, lines = 1012396
```

הערות חשובות:

א) לקבלת מספרים אקראיים יש להשתמש במחלקה java.util.Random ולאתחל את הסדרה של מספרים אקראיים בצורה הבאה לצורך בדיקה אחידה:

```
Random r = new Random(123);
int numLines = r.nextInt(1000);
```

ב) התכנה תיבדק בצורה אוטומטית, והשמות של הפונקציות צריכות להיות זהות לשמות הנ"ל.

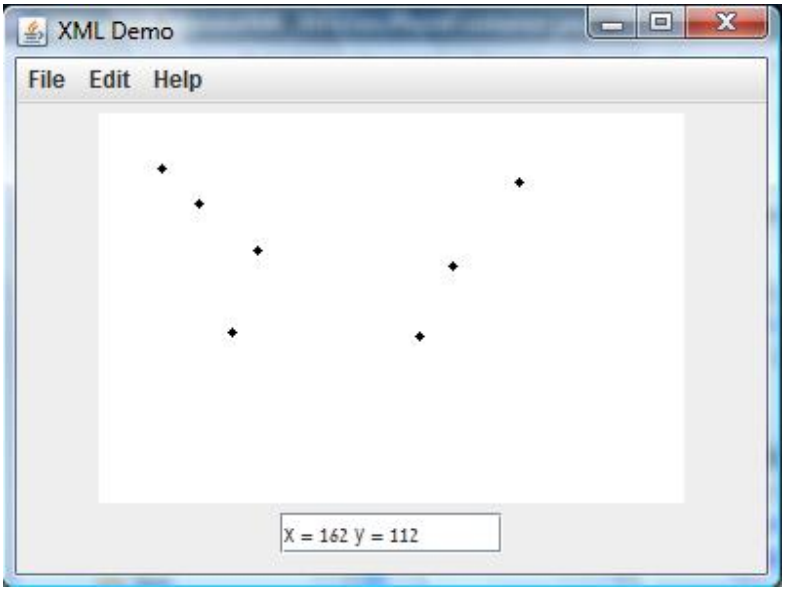
חלק ב XML

מטרתו של חלק זה - לתרגל נושאים של הנדסת תוכנה בפרט:
שימוש ב XML לצורך שמירה ושיחזור של מידע.

הפעולות של האפליקציה הן:

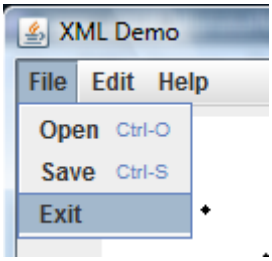
1. לבנות ממשק גראפי ליצירת נקודות.
2. לתת אפשרות לשמור את הנקודות בקובץ XML
3. לתת אפשרות לקרוא את הנקודות מקובץ XML ולשרטט אותן שוב
4. להוסיף העורות לכל מחלקה ולכל שיטה ולבנות JavaDoc
5. לבנות קובץ JAR ולצרף אותו.

ממשק גראפי:



הנקודות מייצרים באמצעות עכבר, שעורי הנקודה צרכים להופיע בחלון שלמטה.

אפשרויות של תפריט File:



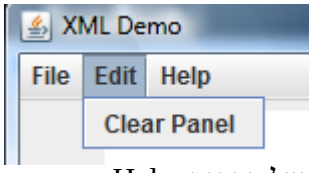
להוסיף גם קיצורי דרך:
Ctrl-O לפתיחת קובץ

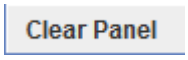
Ctrl-S לשמירת קובץ
EXIT – יציאה מהתוכנית

בלחיצה על Open צריך לקבל FileOpenDialog לבחירת קובץ
בלחיצה על Save צריך לקבל FileSaveDialog לבחירת מקום לשמירת קובץ.

אפשרויות של תפריט Edit

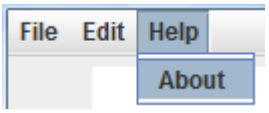
בלחיצה על



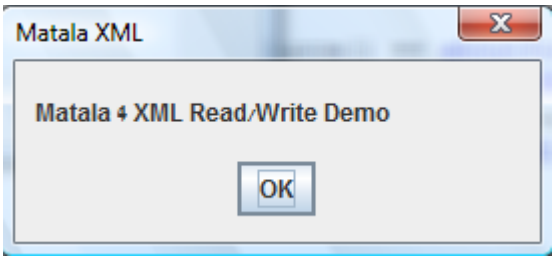


מוחקים את כל הנקודות פאנל.

אפשרויות של תפריט Help:



בלחיצה על About צריך לקבל את ההודעה הבאה:



ניתן להשתמש במחלקת **`PointContainer`** לשמירת נקודות.
רצוי להשתמש ב-NetBeans

עבודה מהנה!