

# תוכנה 1 – סתיו תשע"ג

## תרגיל מספר 3

### הנחיות כלליות:

- קראו בעיון את קובץ נוהלי הגשת התרגילים אשר נמצא באתר הקורס.
- הגשת התרגיל תעשה במערכת ה moodle בלבד (<http://moodle.tau.ac.il/>).
- יש להגיש קובץ zip יחיד הנושא את שם המשתמש ומספר התרגיל (לדוגמא, עבור המשתמש zvainer יקרא הקובץ zvainer\_hw3.zip). קובץ ה-zip יכיל:
  - א. קובץ פרטים אישיים בשם details.txt המכיל את שמכם ומספר ת.ז. הזהות שלכם.
  - ב. קבצי ה-java של התוכניות אותם התבקשתם לממש.
  - ג. קובץ טקסט אחד עם העתק של כל קבצי ה-java.
  - ד. קובץ טקסט בשם answers עם התשובות לשאלות.

### חשוב:

הנכם מתבקשים להקפיד על חתימה של מתודות לפי המפורט בתרגיל, אחרת ייתכן והשאלה לא תיבדק.

### חלק א':

בכל אחד מהסעיפים הבאים עליכם לממש מתודה המקבלת מערך של ציונים (המיוצגים כמספרים שלמים), מבצעת חישוב כלשהו על המערך ומחזירה את תוצאת החישוב. יש לממש את ארבע המתודות באותה מחלקה שתיקרא GradeAnalyzer, על פי הפירוט הבא:

(1) מתודה המקבלת מערך של ציונים ומחזירה את ממוצע הציונים.

```
public static float gradeAverage(int[] grades)
```

(2) מתודה המקבלת מערך של ציונים ומחזירה ממוצע של ציונים עוברים בלבד (ציון 60 ומעלה).

```
public static float passingGradeAverage(int[] grades)
```

(3) מתודה המקבלת מערך של ציונים ומחזירה את הציון הגבוה ביותר.

```
public static int maxGrade(int[] grades)
```

(4) מתודה המחזירה את מספר הציונים שערכם שווה לערך הציון המקסימלי. שימו לב: במימוש מתודה זו יש להשתמש בקריאה למתודה maxGrade שהוגדרה בסעיף הקודם.

```
public static int maxGradeCount(int grades[])
```

שימו לב שבשאלה זו עליכם לממש את המתודות המוגדרות לעיל בעצמכם, מבלי להשתמש בפונקציות הספרייה הקיימות בג'אווה להחזרת ממוצע או מקסימום.

מתודת ה-main של התוכנית תקבל אוסף ציונים כארגומנטים בשורת הפקודה, תמיר את הציונים ממערך של מחרוזות למערך של מספרים שלמים, תקרא לכל אחת מהפונקציות כך שתפעל על מערך הציונים, ותדפיס למסך את תוצאות החישובים על לפי הפורמט המובא בדוגמא הבאה:

```
83 1 75 93 100
```

פלט:

```
Grade Analyzer
```

```
Grade Average:          70.4
Passing Grade Average:  87.75
Max Grade:              100
Max Grade Count:        1
```

**חשוב:** את המתודות בחלקים ב'-ד' בתרגיל זה יש לממש באותה מחלקה בעלת השם Assignment03

### חלק ב':

כתבו את המתודה binaryAdder, המקבלת שתי מחרוזות (טיפוסים מסוג String) המייצגות מספרים בייצוג בינארי ([http://en.wikipedia.org/wiki/Binary\\_numeral\\_system#Addition](http://en.wikipedia.org/wiki/Binary_numeral_system#Addition)), ומחזירה מחרוזת של תוצאת החיבור בין מספרים אלה. ניתן להניח כי הקלט תקין (כלומר מחרוזות הקלט מכילות רק את התווים '0' ו-'1'). לא ניתן להניח חסם על אורך מחרוזות הקלט. להלן דוגמאות:

```
binaryAdder("0","0") -> "0"
binaryAdder("0","1") -> "1"
binaryAdder("1","0") -> "1"
binaryAdder("1","1") -> "10"
binaryAdder("10100101","111") -> "101011100"
binaryAdder("1010010100101","1110111") -> "10101000111100"
binaryAdder("11110111001101011001010000000010","110010") ->
"11110111001101011001010000110100"
```

ניתן להגדיר מבני עזר או שרותים חדשים לצורך המימוש.

```
public static String binaryAdder(String a, String b) {  
}
```

### חלק ג':

כתבו את המתודה `areAnagrams`, המקבלת שתי מחרוזות (טיפוסים מסוג `String`). ובודקת האם האחת מתקבלת משיכול אותיות השנייה. ניתן להניח שהקלט מורכב רק מתווים באנגלית, כאשר לצורך שאלה זו, רווחים אינם נחשבים כאות. ניתן להניח כי הקלט תקין, כלומר שהמחרוזות אינן `null`.

להלן מספר דוגמאות:

```
areAnagrams("Debit Card","Bad Credit") -> true  
areAnagrams("The eyes","They see") -> true  
areAnagrams("Conversation","Voices rant on") -> true  
areAnagrams("Radar","Tartar") -> false
```

ניתן להגדיר מבני עזר או שרותים חדשים לצורך המימוש.

```
public static boolean areAnagrams(String a, String b) {  
}
```

### חלק ד':

(1) ממשו את המתודה `minRunSeq` אשר בהינתן מחרוזת (`string`) מחזירה את רצף הריצה (run sequence) הקצר ביותר במחרוזת. **רצף ריצה** מוגדר בתור תת המחרוזת שבה מופיע אותו התו ברצף (המתודה תחזיר את רצף הריצה הקצר ביותר במחרוזת) במידה וקיימים מספר רצפי ריצה באותו אורך מינימלי, יש להחזיר את האחרון.

להלן כמה דוגמאות:

```
minRunSeq("hoopla") -> "a"  
minRunSeq("aaaabbbbCCCCC") -> "bbb"  
minRunSeq("bbbbbbbaa") -> "aa"  
minRunSeq("") -> ""
```

ניתן להגדיר מבני עזר או שרותים חדשים לצורך המימוש.

```
public static string minRunSeq (String str) {  
}
```

(2) ממשו את השרות `maxRunSeq` אשר בהינתן מחרוזת מחזיר את הריצה הארוכה ביותר במחרוזת. במידה וקיימים מספר רציפי ריצה בעלי אותו אורך מקסימלי, יש להחזיר את האחרון. להלן מספר דוגמאות:

```
maxRunSeq ("hoopla") -> "oo"
```

```
maxRunSeq ("aaaabbbCCCCC") -> "CCCCC"
```

```
maxRunSeq ("bbbbbbbaa") -> "bbbbbbb"
```

```
maxRunSeq ("") -> ""
```

ניתן להגדיר מבני עזר או שרותים חדשים לצורך המימוש.

```
public static string maxRunSeq (String str) {  
}
```

### חלק ה':

חלק זה נועד לתרגל כתיבת חוזים עבור שירותים קיימים. בכל סעיף נתונה פונקציה, עליכם כתוב את החוזה (תנאי קדם ואחר) עבור הפונקציה הנתונה. הניחו כי כל הפונקציות אינן בודקות את הקלט. ניתן להיעזר בדוגמא הראשונה לחוזים במצגת תרגול #3.

1. לוגריתם בבסיס 10

```
public static double log(double d) {  
    ...  
}
```

2. מחלק משותף מקסימלי

```
public static int gcd(int a, int b) {  
    ...  
}
```

3. שורש ריבועי

```
public static double squareRoot(double a) {  
    ...  
}
```

4. מיון

```
public static void sort(int[] arr) {  
    ...  
}
```