



תרגול

## דף תרגילים מחלקות

### ( עבודה מס' 1 ) נקודת התחלה

#### המחלקה Point

#### ממשק המחלקה Point

המחלקה Point שנבנה בתרגיל זה מגדירה נקודה בעלת שתי קואורדינטות: x ו-y.

Point (double x, double y)	הפעולה בונה נקודה חדשה על פי ערכי הפרמטרים D
double GetX()	הפעולה מחזירה את קואורדינטת ה-x של הנקודה
void SetX (double x)	מקבלת ערך מטיפוס double, וקובעת את קואורדינטת ה-x של הנקודה בהתאם D
double GetY()	הפעולה מחזירה את קואורדינטת ה-y של הנקודה
void SetY (double y)	מקבלת ערך מטיפוס double, וקובעת את קואורדינטת ה-y של הנקודה בהתאם D
string ToString()	הפעולה מחזירה מחרוזת המתארת את נתוני הנקודה על פי הצורה הבאה: ( <X> , <Y> )

#### מה עליכם לעשות?

- צרו מחלקה חדשה בשם Point (זכרו: שם המחלקה ושם הקובץ שבו היא נמצאת חייבים להיות זהים).
- יצגו את המחלקה Point (במיליD אחרות: קבעו מה יהיו התכונות של מופעי המחלקה).
- ציירו תרשימ UML המתאימ למחלקה.
- ממשו את פעולות המחלקה Point.

כדי לבדוק שהמחלקה שכתבתם עובדת כראוי, עליכם לכתוב תוכנית בדיקה, לפי ההנחיות האלה:



## המחלקה TestPoint

צרו מחלקה נוספת בשם TestPoint D (בקובץ TestPoint.cs).

בתוך המחלקה TestPoint כתבו פעולת Main(...) המבצעת את משימות האלה:

1. בונה נקודה חדשה לפי הקואורדינטות (7, 43).

2. בונה נקודה חדשה נוספת לפי הקואורדינטות (5, 5).

מדפיסה את שתי הנקודות בעזרת המחזורת המוחזרת מהפעולה ToString() כ y (אינן צורר

לכתוב את שם הפעולה במפורש):

```
Console.WriteLine(xxx);
```

4. מחליפה בין קואורדינטות ה-x של שתי הנקודות, תוך שימוש בפעולות השונות של המחלקה

נקודה.

5. מדפיסה שוב את הנקודות החדשות.

### חלקב:

נרחיב את המחלקה Point ונוסיף לה שתי פעולות:

<b>double</b> Distance (Point p)	הפעולה מקבלת נקודה ומחזירה את המרחק שבינה לביין הנקודה הנוכחית (ראו למטה תזכורת לחישוב המרחק)
Point Middle (Point p)	הפעולה מקבלת נקודה ומחזירה את הנקודה הנמצאת ביין הנקודה שהתקבלה כפרמטר וביין הנקודה הנוכחית באמצע

תזכורת: חישוב נקודת האמצע ביין שתי הנקודות  $(x1, y1)$  ו- $(x2, y2)$  הוא:

$$x_{middle} = \frac{x_1 + x_2}{2} \quad y_{middle} = \frac{y_1 + y_2}{2}$$



חישוב המרחק בין שתי נקודות  $(x_1, y_1)$  ו- $(x_2, y_2)$  הוא:

$$\sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$

פעולות החזקה והשורש קיימות במחלקה Math המופיעה במאגר המחלקות המוכנות של סירפ (ראו ב-MSDN). Math נמצאת בתוך החבילה System ויש "לייבא" אותה.

### ממשק חלקי של המחלקה Math

המחלקה מאגדת בתוכה פעולות מחלקה שונות המבצעות חישובי D מתמטיים נפוצים.

<b>double Pow (double x, double y)</b>	הפעולה מקבלת שני פרמטרים x ו-y, ומחזירה את הערך של x בחזקת y
<b>double Sqrt (double x)</b>	הפעולה מקבלת את הפרמטר x ומחזירה את השורש הריבועי שלו

פעלת הפעולות נעשית דרך D המחלקה. פעולות האלה הן פעולות מחלקה. להוציא שורש מ-9 יש לכתוב:

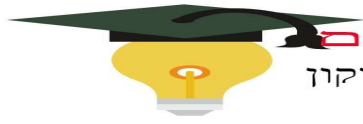
```
double root = Math.Sqrt(9);
```

1. הוסיפו לפעולה הראשית חישוב של המרחק בין שתי הנקודות המקוריות שיצרתם בחלק א,
2. והדפיסו אותו.

הוסיפו לפעולה הראשית חישוב של נקודת האמצע בין הנקודות שיצרתם בחלק א, לאחר החלפת ערכי ה-x. הדפיסו את הנקודה.

### שימו לב:

בפעולה המחשבת נקודת אמצע, ערך ההחזרה גם הוא עצם מסוג Point. כלומר עליכם ליצור את העצם החדש בתוך מימוש הפעולה, ואז להחזיר אותו כערך ההחזרה של הפעולה.



## ( עבודה מס' 2 ) משחק בקוביות



### ממשק המחלקה קובייה – Die

המחלקה Die (קובייה) מגדירה קובייה שלה 6 פאות. על הפאות מופיעים המספרים 1 עד 6. כאשר הקובייה נמצאת במנוחה, ונשאלת השאלה "מהו המספר שהקובייה מראה?" התשובה לכך היא: "המספר שנמצא על הפאה העליונה".

Die()	הפעולה בונה עצם מטיפוס Die. הקובייה שנוצרה מראה מספר אקראי בין 1 ל-6
void Roll()	הפעולה מדמה "הטלת קובייה". בתום הפעולה מתעדכן המספר שהקובייה מראה לאחר ההטלה
int GetNum()	הפעולה מחזירה את המספר שמראה הקובייה

### מה עליכם לעשות?

1. זשבו מהן התכונות הנחוצות למחלקה Die וציירו UML מתאימים למחלקה.
2. כתבו את המחלקה Die במלואה (כולל תיעוד).
3. רמז: כדי להטיל את הקוביות באופן אקראי השתמשו בפעולה Next (1, 7) של המחלקה Random. על אופן פעולתה ראו ב-MSDN.
4. כתבו מחלקה בשם DiceGame (משחק קוביות), ובה פעולה ראשית היוצרת שתי קוביות. בכל תור תטיל התוכנית את שתי הקוביות עד אשר יתקבל הציור: 6, 6. בכל תור יש למעשה שתי הטלות של שתי קוביות המשחק.
5. התוכנית תדפיס את תוצאות ההטלות בכל התורות.
6. כאשר יתקבל הציור 6, 6 תיעצר התוכנית ותדפיס כמה תורות התקיימו עד אשר קיבלנו 6, 6.

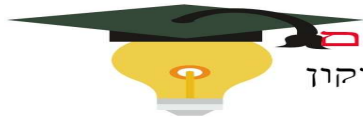


## ( עבודה מס' 3 ) f (Date) תארי

### ממשק המחלקה Date

המחלקה מגדירה את הטיפוס תארי, המורכב מיום, מחודש ומשנה.

<code>Date (int day, int month, int year)</code>	הפעולה בונה עצם מטיפוס Date על פי פרמטריס נתונים. הנחות: ערע הפרמטר day הוא מספר שלם בין 1 ל-31. ערע הפרמטר month הוא מספר שלם בין 1 ל-12. ערע הפרמטר year הוא מספר שלם אי-שלילי בן ארבע ספרות
<code>int GetYear()</code>	הפעולה מחזירה את השנה
<code>int GetMonth()</code>	הפעולה מחזירה את החודש
<code>int GetDay()</code>	הפעולה מחזירה את היום
<code>void SetYear (int yearToSet)</code>	הפעולה קובעת את ערע השנה על פי הפרמטר הנתון. הנחה: ערע הפרמטר הוא מספר שלם אי-שלילי בן ארבע ספרות
<code>void SetMonth (int monthToSet)</code>	הפעולה קובעת את ערע החודש על פי הפרמטר הנתון. הנחה: ערע הפרמטר הוא מספר שלם בין 1 ל-12
<code>void SetDay (int dayToSet)</code>	הפעולה קובעת את ערע היום על פי הפרמטר הנתון. הנחה: ערע הפרמטר הוא מספר שלם בין 1 ל-31
<code>int CompareTo (Date other)</code>	הפעולה מחזירה מספר חיובי א אם התארי הנוכחי מאוחר מהתארי other; 0 - אם התארי שווים; מספר שלילי - אם התארי הנוכחי קודם לתארי other
<code>string ToString()</code>	הפעולה מחזירה מחרוזת המתארת את התארי בצורה הבאה: <code>&lt;day&gt;.&lt;month&gt;.&lt;year&gt;</code>



מה עליכם לעשות?

## חלק א:

1. כתבו את כותרת המחלקה Date, בחרו ייצוג למחלקה וממשו את כל הפעולות הנזכרות בממשק. ניתן להניח תקינות של כל הקלטים ואינן צורף לבצע בדיקות תקינות לגביהם.
2. תעדו את המחלקה כראוי.
3. כתבו תוכנית בדיקה ב-TestDate, ובה בדקו את כל הפעולות שמימשת במחלקה Date, כלומר צרו לפחות שני מופעים של Date, שיפעילו את כל פעולות הממשק.

## חלק ב:

לפניכם תוכנית ראשית המשתמשת במחלקה Date:

```
public static void Main(string[] args)
{
    Date d1 = new Date(16, 7, 1963);
    Date d2 = d1;
    d1.SetDay(20);
    d2.SetYear(1980);
    Console.WriteLine(d1);
    Console.WriteLine(d2);
}
```

1. מה יודפס בתום הרצת התוכנית?
2. כמה עצמים מסוג Date נוצרו במחלקה הראשית? הסבירו.

## עבודה מס' 4 מספר רציונלי

### רקע

מספר רציונלי הוא מספר הניתן לכתיבה כמנה של שני מספרים שלמים: מונה ומכנה. למשל, 0.3 הוא מספר רציונלי כיוון שהוא ניתן לכתיבה כ- $\frac{3}{10}$ . טיפוס כזה כבר קיים בסישרפ כטיפוס פשוט (double). עתה נגדיר אותו בתור מחלקה.

ניתן להגדיר מספר רציונלי בעזרת מחלקה בעלת שתי תכונות: מונה ומכנה, ששניהם מספרים שלמים. הכפלת המונה והמכנה של מספר רציונלי באותו המספר מבטאת ייצוג אחר של אותו המספר. לדוגמה:

$$\frac{1}{3} = \frac{2}{6} = \frac{3}{9}$$

כלומר: יכולים להתקיים עצמים המייצגים אותו מספר רציונלי, אך שרכי תכונותיהם שונים. לא כל שני מספרים מייצגים מספר רציונלי חוקי. כאשר ערך התכונה המייצגת את המכנה הוא 0, המספר איננו חוקי.

### ממשק המחלקה Rational

המחלקה Rational מגדירה מספר רציונלי.

תזכורת: בפעולות רבות יש להתחשב במקרי קצה בעייתיים. כאשר מדובר על קלט לפעולה, נעדיף להתריע בתיעוד הפעולה על הבעיה ולקבוע עבור אילו ערכים תפעל הפעולה כראוי. כע נמנע מהמשתמש במחלקה להעביר ערכים לא רצויים לפעולה. בהמשך לימודיכם תלמדו על מנגנון החריגות שמאפשר להתמודד עם מקרים אלה ולהציע להם פתרונות. נצטרך לבדוק מהן הבעיות העוללות להתעורר תוך כדי הפעולה ולפתור אותן, תוך מימוש הפעולה.



Rational (int x, int y)	פעולה בונה המקבלת שני פרמטרים: x עבור המונה ו-y עבור המכנה. מכנה אינו יכול להיות שווה 0, כלומר כל מספר שייווצר הוא מספר רציונלי חוקי
int GetNumerator()	הפעולה מחזירה את המונה של המספר הנוכחי
int GetDenom()	הפעולה מחזירה את המכנה של המספר הנוכחי
bool IsEqual (Rational num)*	הפעולה מקבלת כפרמטר מספר רציונלי num r, ובודקת האם שני המספרים הרציונליים שווים זה לזה
Rational Multiply (Rational num)	הפעולה מקבלת כפרמטר מספר רציונלי num r, ומחזירה מספר רציונלי שהוא מכפלת הפרמטר במספר הנוכחי
Rational Divide (Rational num)**	הפעולה מקבלת כפרמטר מספר רציונלי num r, ומחזירה מספר רציונלי שהוא המנה של המספר הנוכחי ב-num. יש לבדוק שהמחלק אינו בעל מונה שווה ל-0, DA המחלק אכן לא תקיף תחזיר הפעולה null
string ToString()	הפעולה מחזירה מחרוזת המתארת את המספר הרציונלי בצורה הבאה: <x> / <y>

\* השוואה של מספרים רציונליים תיעשה בעזרת מכפלת המונה של האחד במכנה של השני

$$a \cdot d = c \cdot b$$

והשוואת המכפלות. לדוגמה:

$$\frac{a}{d} = \frac{c}{b}$$

\*\* החלוקה של שבר בשבר נעשית על ידי כפל השבר הראשון בהופכי של השבר השני. לדוגמה:

$$\frac{2}{3} \div \frac{4}{6} = \frac{2}{3} \cdot \frac{6}{4}$$

מה עליכם לעשות?

## חלק א:

1. כתבו את המחלקה Rational. הקפידו לתעד כראוי את הפעולה הבונה תוך ציון הדרישה שהמכנה לא יהיה שווה 0.

2. כתבו תוכנית בדיקה הבודקת את המימוש שכתבתם עבור פעולות הממשק.





חלקב:

קבלת D את המחלקה Rational כמחלקה קיימת, ואינכד יכולי D לשנותה. את D מעונייני D בשתי פעולות נוספות: בפעולת חיבור ובפעולת חיסור של שני מספרי D רציונליי D.

1. איך תוכלו להגדיר ולממש פעולות אלה, והיכמן?
2. כתבו את כותרות הפעולות המתאימות וכמן תיעוד מלא לפעולות אלה.
3. ממשו את פעולת החיבור של שני מספרי D רציונליי D.
4. כתבו תוכנית בדיקה שתוכיח את נכונות הפעולות שמימשת D.



## עבודה מס' 5 הרשאות גישה

מה עליכם לעשות?

בטבלה שלפניכם מופיע קוד חלקי של שתי מחלקות: Alice ו-Bob.

<pre>public class Alice {     public int a1;     private int a2;     public Alice()     {         this.a1 = 1;         this.a2 = 2;     }     public void IncreaseBoth()     {         this.a1++;         this.a2++;     }     private static int FindA2 (Alice a)     {         return a.a2;     } }</pre>	<pre>public class Bob {     private Alice ally;      public Bob()     {         this.ally = new Alice();     }      public void ChangeAlice()     {         Console.WriteLine ((this.ally).a1);         (this.ally).IncreaseBoth();          Console.WriteLine ((this.ally).a2);         (this.ally).FindA2 (this.ally);     } }</pre>
---	--

ענועל השאלות הבאות:

1. האם המחלקה Alice, על פי הקטע המופיע לעיל, תעבור הידור? אם לא, נמקו והדגימו.

2. המחלקה Bob אינה עוברת הידור. נבדוק מדוע:

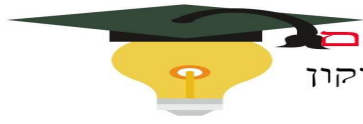
א. האם הפנייה לתכונה a1 בשורה הבאה היא פנייה חוקית או לא? הסבירו:

```
Console.WriteLine ((this.ally).a1);
```

ב. האם בשורה הבאה מותר לזמן פעולה של המחלקה Alice?  
`(this.ally).IncreaseBoth();`

ג. האם השורה הבאה תקינה? הסבירו ונמקו:  
`Console.WriteLine ((this.ally).a2);`

ד. האם מותר לזמן בתוך מחלקה Bob את הפעולה הבאה? הסבירו:  
`(this.ally).FindA2 (this.ally);`



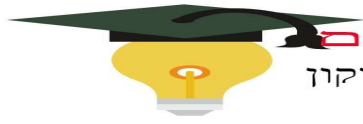
## ( עבודה מס' 6 ) – counter מונה

### רקע

אנו מעוניינים שכל אחת מהנקודות הנוצרות על ידי המחלקה Point תקבל מספר סידורי המציינ איזה מופע היא של המחלקה. כלומר, הנקודה הראשונה שתיווצר מהמחלקה תקבל את המספר הסידורי 1, הנקודה שאחריה תקבל את המספר 2 והנקודה ה-n תקבל את המספר הסידורי n.

### מה עליכם לעשות?

1. הוסיפו למחלקה Point את התכונה או התכונות הנדרשות לצורך מתן מספר סידורי לכל מופע של המחלקה. הסבירו מה עשיתם.D.  
רמז: המספר הסידורי של כל נקודה הוא תכונה המאפיינת את העצם עצמו. המספר הכולל של הנקודות שכבר נוצרו מן המחלקה אינו מאפיינ עצם מסוים.D. היכן יישמר מספר זה?
2. שנו את מימוש הפעולה הבונה כך שכל מופע של נקודה יכיל את מספרו הסידורי מרגע יצירתו.
3. שנו את הפעולה ToString() כך שבראשית המחרוזת המתארת את הנקודה יודפס מספרה הסידורי, ורק בהמשך יופיעו שאר מאפייני הנקודה.
4. הוסיפו למחלקה פעולה המאפשרת לקבל ברגע נתון את מספר המופעים שנוצרו מטיפוס המחלקה עד כה.



## ( עבודה מס' 7 פעולת מחלקה

### מה עליכם לעשות?

ברצוננו להגדיר פעולה השופכת כמות נתונה של מיס מדלי אחד לדלי אחר, להבדיל מהפעולה שבמשק הדלי, השופכת את הכמות המקסימלית האפשרית מדלי למשנהו. איננו רוצים להוסיף פעולה זו לממשק המחלקה דלי (א) שאנו הגדרנו את המחלקה הזו, ולכן זה בהחלט אפשרי), אלא לכתוב אותה כפעולה חיצונית.

ענו על הסעיפים הבאים:

1. כותרת הפעולה: אילו פרמטרים תקבל הפעולה ומאיזה טיפוס? 2.
- מיקום הפעולה: היכן תוגדר הפעולה? איך אינכם בטוחים חזרו וקראו את הסעיפים המתאימים בפרק.
3. מימוש הפעולה: האם במימוש הפעולה ניתן לפנות לתכונות הדלי ישירות? כיצד תממשו את הפעולה, האם תשתמשו בפעולות ממשק כלשהן? אילו? 4.
- מקרי קצה: הגדירו מקרי קצה שיש להתחשב בהם או ציינו בתיעוד מה תעשה הפעולה במקרים אלה.
5. מימוש והפעלה: כתבו את הפעולה במחלקה המכילה את הפעולה הראשית ותעדו אותה. הפעילו את הפעולה כדי להוכיח את נכונות המימוש.



## עבודה מס' 8 כיתה מוזיקלית

### מה עליכם לעשות?

בבית ספר קיימת כיתה שתלמידיה אוהבים מוזיקה. לתלמידי יש ספרייה מוזיקלית משותפת. ה-D אוספים תקליטורים ומשמיעים אותם בהפסקות. בתחילת השנה, כל תלמיד מתחייב להביא מספר מסוים של תקליטורים שונים (INITIAL\_AMOUNT\_OF\_CDS), מכאן ואילך הוא יכול להוסיף תקליטורים D רצונו בכך (בסך r השנה הילד שהביא הכי הרבה תקליטורים יקבל פרס מיוחד מבני כיתתו). רג, חובב המוזיקה והמחשבים, רוצה לערוך מעקב אחר כמות התקליטורים שמביאים התלמידים. כדי לעשות זאת כתב מחלקה המגדירה תלמיד בכיתה.

לפניכם מחלקה שכל מופע שלה מייצג תלמיד בכיתה זו:

```
public class Student
{
    private string name;
    private int myAmount;
    private static int classCdBx = 0;
    public const int INITIAL_AMOUNT_OF_CDS = 3;

    public Student(string name)
    {
        this.name = name;
        this.myAmount = INITIAL_AMOUNT_OF_CDS;
        Student.classCdBx += INITIAL_AMOUNT_OF_CDS;
    }

    public static int GetClassCdBx()
    {
        return Student.classCdBx;
    }

    public int GetStudentAmount()
    {
        return this.myAmount;
    }

    public string GetName()
    {
        return this.name;
    }

    public void EnterCds(int amount)
    {
        Student.classCdBx += amount;
        this.myAmount += amount;
    }
}
```

1. במחלקה אחרת נכתבה הפעולה הראשית. מה תהיה תוצאת ההדפסה?

```
public class Test
{
    public static void Main(string[] args)
    {
        Student[] members = new Student[3];
        members[0] = new Student ("Moshe");
        members[1] = new Student ("Dvir");
        members[2] = new Student ("Michal");

        for (int i = 0; i < members.Length; i++)
        {
            members[i].EnterCds(i);
        }
        Console.WriteLine(members[2].GetStudentAmount());
        Console.WriteLine(Student.GetClassCdBox());
    }
}
```

2. מדוע הוגדר המשתנה classCdBox כ-static?



## עבודה מס' 10

### מחלקת קבצים

לפניכם ממשק של מחלקת **File** (תכונות ופעולות) מוצג ע"י תרשים UML :

#### File

String name

int size

File(String name, int size)

פעולה בונה שיוצרת קובץ חדש על סמך הערכים שהועברו לה

File(File other)

פעולה בונה מעתיקה

boolean isSameFile(File other)

פעולה שמחזירה האם הקבצים זהים

String getName()

int getSize()

void setName(String name)

void setSize(int size)

String ToString()

ממש את המחלקת **File** .

א. לפניכם ממשק של מחלקת **File** (פעולות בלבד) מוצג ע"י תרשים UML :

#### עליכם לממש את המחלקת Folder

##### הנחות והנחיות

- בתיקייה לכל היותר 100 קבצים
- הקבצים בתיקייה יהיו מסודרים כך שהם ממלאים אותה **ברצף** החל מתא 0 (אסור שבסיום של פעולה יהיו "חורים" ברצף הקבצים)
- בתיקייה כל הקבצים שונים זה מזה, כלומר לכל 2 קבצים בתיקייה (a ו-b) הביטוי `a.isSameFile(b)` יחזיר `false`
- יש להגדיר את המינימום ההכרחי של תכונות במחלקת **Folder**, פתרונות שישלבו תכונות שאינן הכרחיות (תכונות שניתן לחשבם על סמך תכונות אחרות) יקבלו ניקוד שאינו מלא

- מומלץ לממש את הפעולות על-פי הסדר בו הן מופיעות בטבלה, בהרבה מקרים מאוד מקל אם נעזרים בפעולות שהוגדרו כבר.

### Folder

Folder()	פעולה בונה שיוצרת תיקייה חדשה <u>ריקה</u>
Folder(Folder other)	פעולה בונה מעתיקה
int getNumOfFiles()	פעולה המחזירה כמה קבצים בתיקייה
File getFile(int i)	פעולה המחזירה את הקובץ (עצמו, לא העתק) הנמצא באינדקס ה-i (אינדקסים מתחילים מ-0)
int calculateSize()	פעולה המחזירה את הגודל הכולל של התיקייה (סכום כל הגדלים)
int contain(File f)	פעולה הבודקת אם התיקייה מכילה את הקובץ f ומחזירה את האינדקס בו הוא נמצא (האינדקס מתחיל מ-0). אם הקובץ לא נמצא בתיקייה, הפעולה תחזיר -1
boolean add(File f)	פעולה המוסיפה את הקובץ f לתיקייה (אם אינו קיים כבר) הפעולה מחזירה true אם הקובץ הוסף, ו- false אחרת.

#### הנחה מקלה:

	אף פעם לא תתבקשו להוסיף קובץ לתיקייה שאין בה מקום פנוי
	פעולה המוחקת קובץ f מהתיקייה (אם קיים בה) ומחזירה true אם מחקה או false אחרת
boolean delete(File f)	שימו לב, יש לדאוג שבסיום הפעולה לא ייווצרו "חורים" ברצף הקבצים שבתיקייה
Folder intersection(Folder f)	פעולה המחזירה תיקייה חדשה ובה כל הקבצים שנמצאים גם בתיקייה this וגם בתיקייה f (חיתוך)

#### שימו לב:

	לפעולה אסור לשנות את התיקייה this
Folder subtract(Folder f)	פעולה המבצעת <b>חיסור</b> בין תיקיות כלומר: על הפעולה להחזיר תיקייה <u>חדשה</u> שבה יש את כל הקבצים בתיקייה this <u>שאינם</u> נמצאים בתיקייה f (כלומר הקבצים בתיקייה this "פחות" הקבצים בתיקייה f)

#### המלצה:





היעזרו בפעולות קודמות

**שימו לב:**

לפעולה אסור לשנות את התיקיה `this`

Folder smallestK(int k)

פעולה המקבלת מספר `k` ומחזירה תיקיה חדשה ובה `k`

הקבצים התיקיה `this` ביותר

**הנחות מקלות:**

- `k` לכל היותר מספר הקבצים שיש כרגע בתיקיה

- כל הקבצים שונים זה מזה בגודל

- אין דבר כזה קובץ שהגודל שלו 0

**שימו לב:**

לפעולה אסור לשנות את התיקיה `this`

String toString()

פעולת תיאור העצם - מחזירה מחרוזת ובה לכל קובץ

בתיקיה יש שורה המתארת אותו,

בנוסף, תכיל המחרוזת בשורה אחרונה "סיכום של

התיקיה":

- כמה קבצים יש בתיקיה ומה גודלם הכולל

**ב. הוסף מחלקה ראשית (תוכנית) עם פעולה ראשית,**

על הפעולה לבצע את הדברים הבאים:

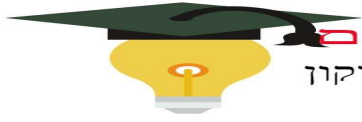
**(1) ליצור 2 תיקיות:**

- את התיקיה הראשונה יש למלא ב-100 קבצים אשר את פרטיהם קלטנו מהמשתמש (שימוש בפקודות מהצורה `console.ReadLine...`).
- את התיקיה השנייה יש "למלא" ב-50 קבצים אשר את פרטיהם קלטנו מהמשתמש.

**המלצה:** על מנת ליצור את 2 התיקות הנ"ל, צרו במחלקה הראשית פעולת עזר (סטטית) אשר

מקבלת מספר `num` ומחזירה תיקיה שיש בה `num` קבצים שנקלטו מהמשתמש.

- על הפעולה ליצור תיקיה חדשה שבה יש את כל הקבצים מתיקיה הראשונה אשר הינם גדולים מ-2500 וגם מופיעים בתיקיה השנייה.
- בסוף הפעולה יש להדפיס את התיקיה שנוצרה.



(2) ליצור תיקייה שבה 4 קבצים.

**שימו לב:** אין לקלוט את פרטי הקבצים מהמשתמש (כלומר לא להשתמש ב- input.next...), על הפרטים להופיע ישירות בתוך קוד הפעולה.

יש להדפיס את 2 הקבצים הכי גדולים שבתיקייה ע"י שימוש בפעולות `smallestK` ו-`subtract`