שם התלמיד/ה: _____ כיתה יא-1 תאריך: 20.5.2026

שם המורה: צבי מלמד

קרא/י בעיון את השאלות שלפניך, וענה/י עליהן כנדרש. שים/י לב לכל חלקי השאלה. **שמור/י על כללי ההתנהגות הנאותים** בשעת מבחן בעיקרם: **<u>שקט ויושר אישי.</u>**

בחינת מתכונת במדעי המחשב כיתה יא'-1 – מועד ב'

<u>משך הבחינה</u> (ללא תוספת זמן) **3:30 שעות**

בבחינה נתונות 6 שאלות, מתוכן 3 שאלות בפרק על מבנה נתונים ו- 3 שאלות בפרק על מונחה עצמים.
עליכם לענות על 4 שאלות מתוכן.
משקל כל השאלות זהה – 27 נקודות לכל שאלה (סה"כ 108 נקודות. בכל מקרה, הציון המקסימלי הוא 100).

הנחיות

- ⇦ כל חומר עזר, למעט מחשב, מותר לשימוש
- ⇦ אין בשום מקרה אפשרות להעביר חומר עזר בין התלמידים
- ⇦ יש לענות על הבחינה בעת כחול או שחור או בעיפרון, לעונים בעיפרון לא תינתן זכות הערעור
- ⇦ ענו על כל שאלה בדף נפרד. רשמו בראש הדף את מספר השאלה. אם זה המשך התשובה – סמנו בהתאם, כך שיהיה ברור לבודק.
- ⇦ **תשובה בכתב יד לא ברור, או מרובת קשקושים ומחיקות – לא תיבדק ותקבל אפס נקודות !!**
- ⇦ **במידת הצורך – העתיקו את התשובה לדף חדש ורישמו באופן ברור X על דף תשובה שלא צריך להיבדק.**

הערות כלליות תכנותיות:

- ⇦ מותר לשתף פונקציות בין סעיפים שונים של אותה שאלה. לדוגמא, אפשר לקרוא בסעיף ב' לפונקציה שכתבנו בסעיף א'. (במקרים שזה רלבנטי).
- ⇦ הקפידו על ההוראות ושימו לב לניסוחים של השאלות.
- ⇦ סמנו V בטבלה להלן – לאיזה שאלות עניתם.

<u>שאלה</u>	**<u>סימון – שאלה שבחרתי לענות</u>**	**<u>ציון</u>**	**<u>הערות / משוב</u>**
1			
2			
3			
4			
5			
6			
סה"כ			

פרק א' – מבנה נתונים

שאלה #1 (27 נק')

נתונה המחלקה Task המתארת משימה במערכת הפעלה. לכל אובייקט מטיפוס Task שתי תכונות:

- name – מחרוזת המתארת את שם המשימה.
- priority – מספר שלם המתאר את עדיפות המשימה. ערך גבוה יותר מציין עדיפות גבוהה יותר.

הניחו שלמחלקה קיימות פעולות Get ו־Set לכל התכונות, ואין צורך לממשן.

סעיף א'

כתבו פעולה חיצונית בשם PriorityRemove המקבלת תור q של אובייקטים מטיפוס Task.
הפעולה תוציא מהתור את האובייקט בעל העדיפות הגבוהה ביותר בתור, ותחזיר אותו.

אם קיימים כמה אובייקטים בעלי אותה עדיפות מקסימלית, יש להחזיר את הראשון מביניהם בתור.
יש לשמור על סדר שאר האיברים בתור.

מגבלות:

- חובה לשמור על מבנה התור פרט לאיבר שהוצא ממנו.
- אין להשתמש במערך או ברשימה מקושרת.
- מותר להשתמש בתורי עזר.
- שימו לב, בסעיף זה התור הוא "תור רגיל" שאיננו מסודר לפי עדיפויות.

סעיף ב'

נתון תור q של אובייקטים מטיפוס Task, המסודר לפי עדיפות בסדר יורד (כלומר זהו תור עדיפויות).
כמו כן נתונים שני מספרים שלמים:

- K – עדיפות מבוקשת.
- delta – מספר חיובי.

כתבו פעולה חיצונית המקבלת את התור ושני המספרים.

הפעולה תוסיף את הערך delta לעדיפות של **כל המשימות שנמצאות בתור q ואשר העדיפות שלהן היא K**, ותמקם אותן מחדש במקום המתאים בתור, כך **שהתור יישאר מסודר לפי עדיפויות** בסדר יורד.
כאשר ממקמים את המשימות עם העדיפות המעודכנת בתור – אין חשיבות לסדר בינן לבין משימות אחרות שהיו בתור באותה עדיפות לפני הפעולה.

מגבלות:

- אותן מגבלות כמו בסעיף א'.

הערה:

- אין קשר בין סעיף א' לסעיף ב' – כלומר אין ציפייה שתשתמשו בפעולה של סעיף א'.

סעיף ג'

מה הסיבוכיות של הפעולה בסעיף ב' ? חובה לנמק באופן ברור ומשכנע.

שאלה 2 (27 נק')

רצף עולה ברשימה הוא רצף של איברים עוקבים, שבו כל איבר גדול מהאיבר שלפניו.
לדוגמא, ברשימה:

```
3 → 5 → 8 → 2 → 1 → 7 → 9 → 11 → 4 → 10
```

קיימים שלושה רצפים עולים:

```
3 → 5 → 8
1 → 7 → 9 → 11
4 → 10
```

נגדיר "**רצף עולה באורך לפחות-K**", עבור $K \geq 2$ כרצף עולה שאורכו הוא לפחות K.
לדוגמא, אם $K = 3$, אז שני הרצפים הראשונים הם רצפים עולים באורך 3 לפחות. אם $K = 4$, אז רק הרצף: $1 \rightarrow 7 \rightarrow 9 \rightarrow 11$ נחשב. ואם $K = 2$ אז כל הרצפים נחשבים, ונוכל להגיד שהרשימה מכיל 3 רצפים עולים באורך 2-לפחות.

אפשר להניח ש $K \geq 2$.

סעיף א'

כתבו פעולה **רקורסיבית** חיצונית בשם `CountIncreasingSequences` המקבלת רשימה של מספרים שלמים ומספר שלם K.
הפעולה תחזיר כמה רצפים עולים ברשימה הם באורך של לפחות K.

לדוגמא, עבור הרשימה בדוגמא למעלה, אם $K = 2$ הפעולה תחזיר 3, ואם $K = 4$ הפעולה תחזיר 1.

סעיף ב'

כתבו פעולה חיצונית בשם `RemoveIncreasingSequences` המקבלת רשימה של מספרים שלמים ומספר שלם K.

```
Node<int>[] RemoveIncreasingSequences(Node<int> lst, int K)
```

הפעולה תחזיר מערך בגודל 2:

- `arr[0]` – הרשימה הכוללת את האיברים שלא הוסרו
- `arr[1]` – הרשימה הכוללת את האיברים שהוסרו

לדוגמא, עבור הרשימה הנ"ל והערך $K = 3$:

```
arr[0] : 2 → 11 → 4 → 10
arr[0] : 3 → 5 → 8 → 1 → 7 → 9 → 11
```

מגבלות:

- יש לשמור על הסדר המקורי של האיברים בכל אחת מרשימות המערך.
- אין לייצר איברים חדשים אלא להשתמש באיברים הקיימים.

סעיף ג'

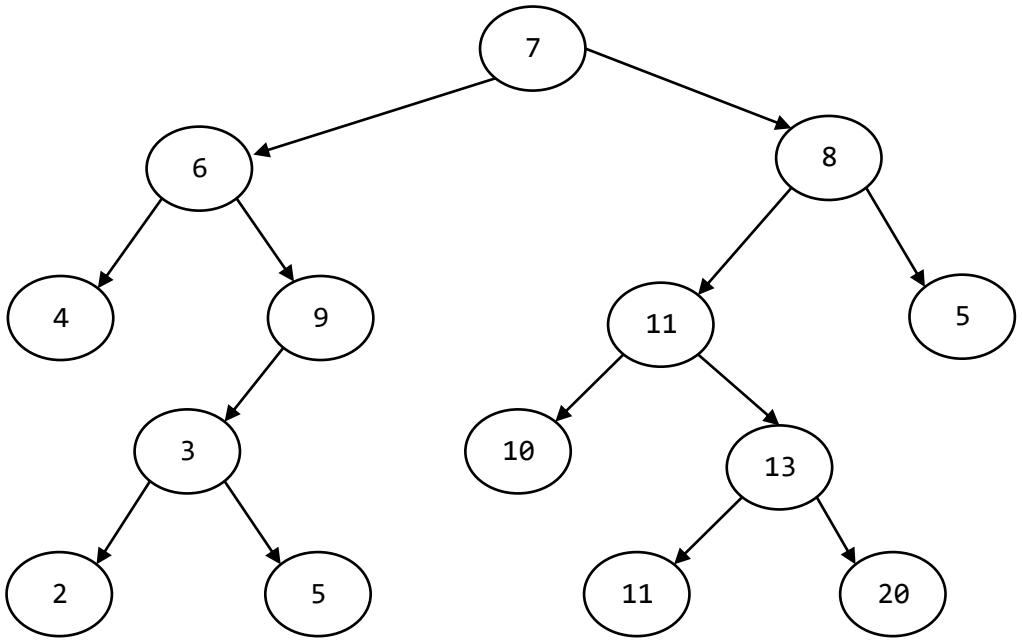
מה הסיבוכיות של הפעולה שכתבת בסעיף ב'? חובה לנמק באופן ברור ומשכנע.

שאלה 3 (27 נק')

נתון עץ בינארי של מספרים שלמים.

נגדיר צומת יציב כצומת המקיים את התנאי הבא:

- לצומת יש בן שמאלי ובן ימני.
- ערך הצומת שווה לממוצע החשבוני של ערכי שני בניו (ממוצע כפי שמוגדר בפעולת חילוק של שלמים).



לדוגמא, בעץ הזה:

- שורש העץ 7 הוא צומת יציב כי יש לו שני בנים ומתקיים (6+8)/2 == 7
- הצומת 6 הוא יציב, כי יש לו שני בנים, ומתקיים (4+9)/2 == 6
- הצומת 3 הוא יציב, כי יש לו שני בנים, ומתקיים (2+5)/2 == 3
- הצומת 9 אינו יציב – כי יש לו רק בן אחד.
- כל העלים אינם נחשבים לצמתים יציבים.
- הצומת 13 איננו יציב. אמנם יש לו שני בנים אבל: (11+20)/2 = 10 != 13

סעיף א'

כתבו פעולה רקורסיבית חיצונית בשם `CountStableNodes` המקבלת עץ בינארי של מספרים שלמים ומחזירה כמה צמתים יציבים יש בעץ.

סעיף ב'

נגדיר "מסלול-יציב" בעץ, כרצף של צמתים יציבים שמתחילים **משורש העץ**.

כתבו פעולה רקורסיבית חיצונית בשם `LongestStablePath` המקבלת עץ בינארי של מספרים שלמים.
הפעולה תחזיר את אורך המסלול הארוך ביותר של צמתים יציבים, המתחיל בשורש העץ.
אם השורש אינו צומת יציב, הפעולה תחזיר 0.

סעיף ג'

כתבו פעולה רקורסיבית חיצונית בשם `ExistsStablePath` המקבלת עץ בינארי של מספרים שלמים.
הפעולה תחזיר `true` אם קיים לפחות מסלול אחד מהשורש לעלה שבו כל הצמתים במסלול הם צמתים יציבים.
אחרת, הפעולה תחזיר `false`.

פרק ב' – תכנות מונחה עצמים

<u>שאלה 4 (27 נק')</u>

בחברה עירונית לגביית ארנונה פותחה מערכת ממוחשבת לניהול נכסי נדל"ן בתחומי העיר. המערכת כוללת את המחלקות: Owner (בעל נכס), Property (נכס), Residential (בית למגורים), Commercial (נכס מסחרי), Shop (חנות) ו-Office (משרד).

להלן פירוט תכונות המחלקות:

- למחלקה Owner שתי תכונות: name (שם בעל הנכס – מחרוזת), id (מספר זהות – מחרוזת).
- למחלקה Property שלוש תכונות: owner (בעל הנכס – מטיפוס Owner), address (כתובת – מחרוזת), area (שטח בנוי במ"ר – מספר ממשי).
- למחלקה Residential תכונה אחת: numResidents (מספר הדיירים בבית – מספר שלם).
- למחלקה Commercial תכונה אחת: license (מספר רישיון עסק – מחרוזת).
- למחלקה Shop תכונה אחת: streetFront (האם לחנות יש חזית לרחוב – בוליאני).
- למחלקה Office תכונה אחת: floor (מספר הקומה – מספר שלם).

הניחו שלכל התכונות קיימות פעולות **Get** ו-**Set** ואין צורך לממשן.

<u>סעיף א'</u>

1. סרטטו תרשים היררכיה המתאר את הקשר בין המחלקות בהתאם לתיאור הנתון. יש לסמן ירושה באמצעות חץ והכלה באמצעות הסימן המתאים (קו עם מעוין בבסיס של המחלקה המכילה).
2. כתבו את כותרות המחלקות ואת התכונות שלהן בשפת C#.

<u>סעיף ב'</u>

נתונה הפעולה הבונה של המחלקה **Owner**: (אין צורך לממש פעולה זו).

```
public Owner(string name, string id)
```

ממשו את הפעולה הבונה של המחלקה **Property**. הפעולה מקבלת בעל נכס, כתובת ושטח בנוי:

```
public Property(Owner owner, string address, double area)
```

ממשו את הפעולה הבונה של המחלקה **Shop**. הפעולה מקבלת בעל נכס, כתובת, שטח בנוי, מספר רישיון עסק וערך בוליאני המציין האם לחנות יש חזית לרחוב:

```
public Shop(Owner owner, string address, double area, string license, bool
streetFront )
```

<u>סעיף ג' ← בעמוד הבא</u>

סעיף ג'

העירייה גובה ארנונה עבור הנכסים לפי הכללים הבאים:

- למחלקה **Property (נכס רגיל):** 20 שקלים לכל מ"ר בנוי.
- למחלקה **Residential (בית מגורים):** תעריף בסיסי של נכס רגיל. אם מספר הדיירים בבית גדול מ־4, תינתן הנחה של 100 שקלים.
- למחלקה **Commercial (נכס מסחרי):** 30 שקלים לכל מ"ר בנוי.
- למחלקה **Shop (חנות):** תעריף בסיסי של נכס מסחרי. אם לחנות יש חזית לרחוב, תתווסף תוספת של 500 שקלים.
- למחלקה **Office (משרד):** תעריף בסיסי של נכס מסחרי. אם המשרד נמצא בקומה גבוהה מ־10, יש תוספת של 300 שקלים.

כתבו פעולה בשם **CityTax** בכל מחלקה שבה הדבר נדרש. הפעולה תחזיר מספר ממשי המייצג את סכום הארנונה עבור הנכס.

יש להקפיד על עקרונות תכנות מונחה עצמים (ירושה ופולימורפיזם).

שימו לב: אין לשנות את תכונות המחלקות, ואין להשתמש באופרטורים **is** או **as**.

שאלה 5 (27 נק')

לפניכם המחלקות A ו- B:

```
public class A
{
    protected int x;

    public A(int x)
    {
        this.x = x * 2;
    }

    public virtual void Foo()
    {
        this.x = this.x + 1;


    public void Bar()
    {
        Foo();
    }

    public override string ToString()
    {
        return "x = " + this.x;
    }
}

    public override string ToString()
    {
        return base.ToString() +
               " y = " + this.y;
    }
}
```

```
public class B : A
{
    private int y;

    public B(int x, int y) : base(x)
    {
        this.y = y * this.x;
    }

    public override void Foo()
    {
        base.Foo();
        this.y = this.y + 2;
    }

    public void Goo()
    {
        this.x = this.x * 2;
    }
```

סעיף א'

נתון קטע הקוד הבא:

```
A a1 = new A(3);
A a2 = new B(2, 5);
B b1 = new B(4, 1);
Object obj = a2;

a1.Bar();
a2.Bar();
b1.Goo();
((B)obj).Foo();
```

1. העתיקו את הטבלה הבאה לדף התשובות ומלאו אותה:

שורת קוד	פעולות שהופעלו ציינו שם מחלקה+פעולה, למשל: A.foo() או B.foo()	אובייקט או אובייקטים שהשתנו והערכים שלהם
`a1.Bar();`		
`a2.Bar();`		
`b1.Goo();`		
`((B)obj).Foo();`		

2. ציינו מה יודפס כתוצאה מהרצת קטע הקוד הבא (בהמשך לקטע הקוד הקודם)

```
Console.WriteLine(a1);
Console.WriteLine(a2);
Console.WriteLine(b1);
Console.WriteLine(obj);
```

<u>סעיף ב'</u>

לכל אחד מקטעי הקוד הבאים:

- קבעו אם הקוד עובר קומפילציה.
- אם הקוד אינו עובר קומפילציה – הסבירו את השגיאה.
- אם הקוד עובר קומפילציה, ציינו האם תתרחש שגיאת זמן ריצה או שהקוד ירוץ באופן תקין.
- אם הקוד ירוץ באופן תקין – כתבו את הפלט.

סעיף ב-#1	`Object obj = new A(4);` `((B)obj).Foo();`

סעיף ב-#2	`A a = new B(3, 2);`
	`a.Foo();`
	`Console.WriteLine(a);`

סעיף ב-#3	`Object obj = new B(1, 7);` `Console.WriteLine(obj);`
סעיף ב-#4	`A a = new B(2, 3);` `a.Goo();`

סעיף ב-#5	בתוך המחלקה A:
	`public A(int x, int k)`
	`{`
	`  this.x = x + k;`
	`}`
	בתוך המחלקה B
	`public B(int a, int b, int c) : base(a, b)`
	`{`
	`  y = c * x;`
	`}`
	ולאחר מכן בתוך ה main()
	`B b2 = new B(2, 3, 4);`
	`Console.WriteLine(b2);`
	`Console.WriteLine((A)b2);`

שאלה 6 (27 נק')

לפניכם המחלקות XX ו- YY:

```
public class XX
{
    protected int x;

    public XX(int x) { this.x = x; }
    public virtual void Foo() { this.x++ };
    public override string ToString()
    {
        return "x = " + this.x;
    }
}

public class YY : XX
{
    private int y;
    public YY(int x, int y) : base(x) { this.y = y; }
    public override void Foo()
    {
        base.Foo();
        this.y = this.y + 2;
    }
    public void Goo() { this.x = this.x * 2; }
    public override string ToString()
    {
        return base.ToString() + " y = " + this.y;
    }
}
```

סעיף א'

לכל אחד מקטעי הקוד הבאים:

- קבעו אם הקוד עובר קומפילציה.
- אם הקוד אינו עובר קומפילציה – הסבירו את השגיאה.
- אם הקוד עובר קומפילציה, ציינו האם תתרחש שגיאת זמן ריצה או שהקוד ירוץ באופן תקין.
- אם הקוד ירוץ באופן תקין – כתבו את הפלט.

סעיף א-#1	`XX a = new YY(2, 5);`
	`a.Foo();`
	`Console.WriteLine(a);`

סעיף א-#2	`Object obj = new YY(3, 4);`
	`((YY)obj).Goo();`
	`Console.WriteLine(obj);`

סעיף א-#3	`Object obj = new XX(5);`
	`YY y = (YY)obj;`
	`y.Foo();`

סעיף א-#4	`XX a = new YY(1, 2);` `a.Goo();`
סעיף א-#5	`YY y = new YY(2, 3);` `Console.WriteLine((Object)y.ToString());`

<u>סעיף ב'</u>

1. נתון קטע הקוד הבא:

```
XX[] arr = new XX[3];

arr[0] = new XX(2);
arr[1] = new YY(1, 4);
arr[2] = new YY(3, 2);

for (int i = 0; i < arr.Length; i++)
    Console.WriteLine(arr[i]);
```

כתבו מה הקוד מדפיס.

2. נתונה הפעולה הבאה:

```
public static void F(XX obj)
{
    obj.Foo();
    Console.WriteLine(obj);
}
```

כתבו מה יודפס כתוצאה מהרצת קטע הקוד הבא:

```
F(new XX(3));
F(new YY(2, 5));
```

3. כתבו קטע קוד קצר **שאינו עובר קומפילציה** משום שהפעולה המבוקשת אינה קיימת בטיפוס הסטטי של המשתנה. (טיפוס סטטי: הטיפוס שעליו אנחנו מכריזים)