

לתרגול אדפטיבי, בדיקת תשובות אוטומטית, ועוזר בינה מלאכותית חינם, היכנסו ל-www.TIRGUL.ai

```
Player p1 = new Player("Dana", 40);  
Player p2 = new Player("Omer", 25);  
p1.AddPoints(10);  
Console.WriteLine(p1.GetScore());  
Console.WriteLine(p1);
```

הפלט:

```
50  
Dana: 50
```

פעולות Get ו-Set

Get מחזירה את ערך התכונה, ו-Set מקבלת ערך חדש ומכניסה אותו לתכונה. פעולה שרק משנה תכונה היא void.

```
public int GetScore()  
{  
    return this.score;  
}  
  
public void SetScore(int score)  
{  
    this.score = score;  
}
```

פעולות פנימיות

פעולה פנימית משנה תכונה, או מחשבת ומחזירה ערך. פעולה בוליאנית מחזירה true או false ב-1 if, ו-0 else.

מחלקה ותכונות

מחלקה היא תבנית לעצמים: היא קובעת אילו תכונות יש לכל עצם ואילו פעולות הוא יודע לבצע. התכונות נכתבות בראש המחלקה, תמיד private, ולכן רק הפעולות של המחלקה ניגשות אליהן ישירות.

```
public class Player  
{  
    private string name;  
    private int score;  
    private bool active;  
}
```

פעולה בונה ו-this

הפעולה הבונה נקראת בשם המחלקה, בלי טיפוס מוחזר, ורצה פעם אחת כשיוצרים עצם ב-new. this.name היא התכונה של העצם הנוכחי, ו-name לבדו הוא הפרמטר. כשלשניהם אותו שם, this חובה. תכונה יכולה לקבל ערך קבוע, ואפשר לבדוק תקינות לפני ההשמה.

```
public Player(string name, int score)  
{  
    this.name = name;  
    if (score < 0)  
    {  
        this.score = 0;  
    }  
    else  
    {  
        this.score = score;  
    }  
    this.active = true;  
}
```

יצירת עצם ושימוש בו

בתוכנית הראשית יוצרים עצם ב-new עם הארגומנטים לפעולה הבונה, ומפעילים פעולות בעזרת נקודה אחרי שם העצם. לתכונה private אי אפשר לגשת מבחוץ (p1.score היא שגיאה), רק דרך הפעולות.

```
Player a = new Player("Noa", 10);
Player b = a;
b.AddPoints(5);
Console.WriteLine(a.GetScore());
Player c = null;
if (c == null)
{
    Console.WriteLine("No player");
}
```

הפלט:

```
15
No player
```

פעולה חיצונית שמקבלת עצמים

פעולה חיצונית (static, מחוץ למחלקה) מקבלת עצם כפרמטר ומשתמשת בפעולות שלו. אין בה `this`. היא יכולה לשנות את העצם שקיבלה, להשוות שני עצמים, או להחזיר אחד מהם. בקריאה שולחים את שם העצם: `Player top = Better(p1, p2);`

```
static Player Better(Player a, Player b)
{
    if (a.GetScore() > b.GetScore())
    {
        return a;
    }
    else
    {
        return b;
    }
}
```

פעולה פנימית עם עצם אחר (other)

פעולה פנימית יכולה לקבל עצם נוסף מאותה מחלקה, בדרך כלל בשם `other`. בקריאה `p1.Beats(p2)`, `this` הוא `p1`, העצם שעליו הופעלה הפעולה, ו-`other` הוא `p2`, העצם שנשלח בסוגריים. בתוך המחלקה מותר לגשת ישירות גם לתכונות של `other`.

```
public bool Beats(Player other)
{
    if (this.score > other.score)
    {
        return true;
    }
    else
    {
        return false;
    }
}
```

```
public void AddPoints(int points)
{
    this.score = this.score + points;
}

public bool IsHigh()
{
    if (this.score >= 50)
    {
        return true;
    }
    else
    {
        return false;
    }
}
```

פעולה שנעזרת בפעולה אחרת

פעולה יכולה להפעיל פעולה אחרת של אותו עצם. גם לפני שם הפעולה כותבים `this`, וכך לא חוזרים על החישוב שכבר כתוב ב-`IsHigh`.

```
public string Level()
{
    if (this.IsHigh() == true)
    {
        return "Pro";
    }
    else
    {
        return "Rookie";
    }
}
```

ToString והדפסת עצם

`ToString` מחזירה מחרוזת שמתארת את העצם. הכותרת שלה קבועה, כולל `override`. כשמדפיסים עצם, `Console.WriteLine(p1)`, מודפס מה ש-`ToString` מחזירה.

```
public override string ToString()
{
    return this.name + ": " + this.score;
}
```

שני שמות לאותו עצם, ו-null

ההשמה `b = a` אינה יוצרת עצם חדש: שני השמות מצביעים על אותו עצם, ושינוי דרך `b` נראה גם דרך `a`. משתנה מטיפוס מחלקה שלא הוכנס אליו עצם מכיל `null`, ואי אפשר להפעיל עליו פעולה.

מערך של עצמים

`new Player[3]` יוצר שלושה תאים ריקים, ובכל אחד `null`, עד שמכניסים לתא עצם ב-`new`. על תא `arr[i].GetScore()` מפעילים פעולה כמו על עצם: `arr.Length` עוברת על כל התאים.

```
Player[] arr = new Player[3];
arr[0] = new Player("Dana", 40);
arr[1] = new Player("Omer", 25);
arr[2] = new Player("Noa", 60);
for (int i = 0; i < arr.Length; i++)
{
    Console.WriteLine(arr[i]);
}
```

פעולות על מערך של עצמים

הצורות מנושא 12 עובדות גם כאן: סכום, ספירה לפי תנאי, "האם קיים" (מחזירים `true` בתוך הלולאה ו-`false` אחריה), וקיצוני. משווים את מה שהפעולות של `arr[i]` מחזירות, ואת העצם הקיצוני שומרים במשתנה מטיפוס המחלקה שמתחיל מ-`arr[0]`. פעולה יכולה גם לבנות ולהחזיר מערך חדש.

```
static Player Best(Player[] arr)
{
    Player best = arr[0];
    for (int i = 1; i < arr.Length; i++)
    {
        int s = arr[i].GetScore();
        if (s > best.GetScore())
        {
            best = arr[i];
        }
    }
    return best;
}
```

הכלה: תכונה שהיא מערך

תכונה יכולה להיות מערך. הפעולה הבונה מקבלת את המערך ושומרת אותו, והפעולות הפנימיות עוברות עליו עם `this.points[i]` ו-`this.points.Length`.

```
public class Quiz
{
    private string topic;
    private int[] points;

    public Quiz(string topic, int[] points)
    {
        this.topic = topic;
        this.points = points;
    }

    public double Average()
    {
        int sum = 0;
        int n = this.points.Length;
        for (int i = 0; i < n; i++)
        {
            sum += this.points[i];
        }
        return (double)sum / n;
    }
}
```

הכלה: תכונה שהיא עצם

תכונה יכולה להיות עצם ממחלקה אחרת. הפעולה הבונה מקבלת עצם מוכן, או שיוצרים אותו בתוך הקריאה, `new` בתוך `new`. לתכונות של העצם הפנימי ניגשים דרך הפעולות שלו, ואפשר לשרשר קריאות: `c.GetOwner().GetName()` ב-`c.ToString()`, הצמדת `this.owner` למחרוזת מפעילה את ה-`ToString` של הבעלים.

```
public class Car
{
    private string model;
    private Owner owner;

    public Car(string model, Owner owner)
    {
        this.model = model;
        this.owner = owner;
    }

    public Owner GetOwner()
    {
        return this.owner;
    }

    public override string ToString()
    {
        return this.model + ", " + this.owner;
    }
}
```

```
private Player[] squad;

public int Total()
{
    int sum = 0;
    int n = this.squad.Length;
    for (int i = 0; i < n; i++)
    {
        sum += this.squad[i].GetScore();
    }
    return sum;
}
```

```
Owner o = new Owner("Dana", 30);
Car c = new Car("Mazda", o);
Car d = new Car("Kia", new Owner("Tom", 45));
Console.WriteLine(c.GetOwner().GetName());
```

מחלקה שמחזיקה מערך של עצמים

השילוב של השניים: תכונה שהיא מערך של עצמים, ופעולות שעוברות עליו ומפעילות את הפעולות של העצמים שבתוכו. הפעולה הבונה מקבלת את המערך ושומרת אותו, כמו ב- Quiz.