מבוא למדעי המחשב – כתה יא-1

תאריך: 5.1.2026

דף תקציר לתכנות גנרי + Node<T>

(עדכון - גרסה -#1 - 8.1.2026)

הגדרת המחלקה הגנרית(The Blueprint)

זוהי תבנית של מחלקה. בשפת C# זה נקרא "טיפוס פתוח" (Open Type).

חשוב להבין ולשים לב למבנה המדויק שמופיע בדרך כלל בבחינות.

בדרך כלל לא נצטרך לכתוב את המחלקה הזאת אלא להשתמש בה.

```
public class Node<T>
{
    private T value;
    private Node<T> next;

    // constructor without reference to next
    public Node(T value)
    {
        this.value = value;
        next = null;
    }

    // constructor with reference to next
    public Node(T value, Node<T> next)
    {
        this.value = value;
        this.next = next;
    }

    public T GetValue()
    {
        return this.value;
    }

    public Node<T> GetNext()
    {
        return this.next;
    }

    public void SetValue(T value)
    {
        this.value = value;
    }

    public void SetNext(Node<T> next)
    {
        this.next = next;
    }

    public bool HasNext()
    {
        return this.next != null;
    }
}
```

החומר נועד לשימוש אישי של תלמידי תיכון קריית שרת בלבד.
אין להעתיק, לשכפל או להפיץ ללא רשות © צבי מלמד

כתיבת פעולות סטטיות גנריות על מערכים

בדיוק כמו בחוליות (Nodes), גם במערכים ניתן לכתוב פעולה אחת שתתאים לכל סוגי המערכים (מספרים, מחרוזות וכו'). כדי שהפעולה תהיה גנרית, עלינו להוסיף את הסימון <T> מיד לאחר שם הפעולה, ולהשתמש ב T-כסוג הנתונים של המערך בפרמטרים. זכרו: בתוך פעולה גנרית נשתמש ב- ()Equals להשוואת ערכים.

- חובה להוסיף <T> אחרי שם הפעולה.

דוגמה 1: הדפסת מערך מכל סוג(void)

```
public static void PrintArray<T>(T[] arr)
{
    for (int i = 0; i < arr.Length; i++)
    {
        Console.Write(arr[i] + " ");
    }
    Console.WriteLine();
}
```

דוגמה 2: חיפוש ערך במערך (החזרת אמת/שקר)

```
public static bool Contains<T>(T[] arr, T valueToFind)
{
    for (int i = 0; i < arr.Length; i++)
    {
        // שימוש ב
        // Equals
        // כי אנחנו לא יודעים איזה טיפוס אנחנו בודקים
        if (arr[i].Equals(valueToFind))
        {
            return true;
        }
    }
    return false;
}
```

דוגמא 3: החזרת הערך המקסימלי במערך של int או double

הערה: לא נדגים על טיפוס כלשהו T, כי אז האופרטור < (גדול מ..) וכו' לא בהכרח מוגדר. במקרה כזה צריך להשתמש בפונקציית השוואה שנקראת CompareTo אבל זה מסובך קצת, כי לא תמיד היא מוגדרת (להבדיל מ Equals – בקיצור – זה לא נדרש לבגרות). אבל עבור double למשל זה אפשרי:

שימו לב: כאן צריך להשתמש בטיפוס T גם כארגומנט (טיפוס המערך), גם בתור משתנה לוקלי בתוך הפונקציה, וגם בהכרזת הפונקציה – הערך המוחזר.

```
// Generic Max
public static T GetMax<T>(T[] arr) where T : IComparable<T>
{
    T max = arr[0];
    foreach (T item in arr)
    {
        if (item.CompareTo(max) > 0)
            max = item;
    }
    return max;
}
```

החומר נועד לשימוש אישי של תלמידי תיכון קריית שרת בלבד.
אין להעתיק, לשכפל או להפיץ ללא רשות

© צבי מלמד

כתיבת פעולות סטטיות גנריות על רשימות מהטיפוס Node<T>

פעולה שמכניסה איבר בתחילה הרשימה

שימו לב שהפעולה מקבלת ערך, ויוצרת את החוליה!

הערך שהיא מחזירה זה ראש הרשימה החדשה.

```
// הפעולה יוצרת חוליה חדשה ומציבה אותה בראש השרשרת
public static Node<T> InsertAtBeginning<T>(Node<T> chain, T
newValue)
{
    // יצירת חוליה חדשה שמצביעה על תחילת השרשרת הקיימת
    Node<T> newNode = new Node<T>(newValue, chain);

    // החזרת החוליה החדשה כדי שתהיה ה"ראש" החדש ב-Main
    return newNode;
}
```

פעולה שמכניסה איבר בסוף הרשימה

דגש: שימו לב לשימוש בפעולה HasNext() של המחלקה Node<T> !!

```
public static Node<T> AddToEnd<T>(Node<T> chain, T value)
{
    // מקרה קצה: הרשימה ריקה – החוליה החדשה היא הראש
    if (chain == null)
    {
        return new Node<T>(value);
    }

    // מקרה כללי: סריקה עד החוליה האחרונה
    Node<T> pos = chain;
    while (pos.HasNext())
    {
        pos = pos.GetNext();
    }

    // הוספת החוליה החדשה אחרי האחרונה
    pos.SetNext(new Node<T>(value));

    // מחזירים את ראש השרשרת המקורי (שלא השתנה במקרה זה)
    return chain;
}
```

תרגיל:

א. כתוב פעולה גנרית GetLastNode שמחזירה את האיבר האחרון ברשימה.

ב. כתוב מחדש את הפונקציה הקודמת (הכנסת איבר בסוף הרשימה) תוך שימוש בפעולה GetLastNode.

החומר נועד לשימוש אישי של תלמידי תיכון קריית שרת בלבד.
אין להעתיק, לשכפל או להפיץ ללא רשות

© צבי מלמד

פעולה הסופרת כמה חוליות יש בשרשרת:

```
public static int CountNodes<T>(Node<T> chain)
{
    int count = 0;
    Node<T> pos = chain; // יצירת "מצביע" זמני לסריקה

    while (pos != null)
    {
        count++;
        pos = pos.GetNext(); // קידום המצביע לחוליה הבאה
    }
    return count;
}
```

פעולה שבודקת אם שתי חוליות הן שוות, כלומר, ה value שלהם הוא אותם ערכים

```
public static bool AreEqualNodes<T>(Node<T> n1, Node<T> n2)
{
    // כדי למנוע קריסה בדיקה אם אחת החוליות היא null
    if (n1 == null || n2 == null)
        return n1 == n2; // מחזיר true רק אם שניהם null

    // להשוואת הערכים הגנריים Equals-שימוש ב
    return n1.GetValue().Equals(n2.GetValue());
}
```

דגש: אנחנו לא יודעים איזה טיפוס הוא ה value ולכן לא בטוח שהפעולה != or == מוגדרת. ולכן, משתמשים בפעולה Equals() שהשפה דואגת לכך שהיא תמיד תהיה מוגדרת.

החומר נועד לשימוש אישי של תלמידי תיכון קריית שרת בלבד.
אין להעתיק, לשכפל או להפיץ ללא רשות

© צבי מלמד

פעולה שבודקת אם שתי רשימות הן שוות, כלומר, יש להן חוליות עם אותן האיברים בדיוק, לפי אותו הסדר

דגש: אנחנו משתמשים בפעולה הקודמת !

```
public static bool AreListsEqual<T>(Node<T> list1, Node<T> list2)
{
    Node<T> p1 = list1;
    Node<T> p2 = list2;

    // סריקה במקביל של שתי הרשימות
    while (p1 != null && p2 != null)
    {
        // שימוש בפעולה שכתבנו למעלה להשוואת התוכן
        if (!AreEqualNodes(p1, p2))
            return false;

        p1 = p1.GetNext();
        p2 = p2.GetNext();
    }

    // הן שוות באורכן (שתיהן null) אם שתי הרשימות הסתיימו באותו זמן
    return p1 == null && p2 == null;
}
```

תבנית – דוגמא לפעולה רקורסיבית עם רשימה גנרית

```
public static int SimpleRecursive<T>(Node<T> chain)
{
    // 1. תנאי עצירה (רשימה ריקה)
    if (chain == null) return 0;

    // 2. חישוב על האיבר הנוכחי + קריאה לבאה בתור
    return 1 + SimpleRecursive(chain.GetNext());
}
```

תרגיל – מחיקת כל איבר שני ברשימה,

כתבו פונקציה שמוחקת את כל האיברים במיקום הזוגי מרשימה גנרית.

(אם הרשימה ריקה או מכילה איבר אחד – הפונקציה לא עושה דבר.

אם הרשימה מכילה 2 או 3 איברים – היא מוחקת את האיבר השני.

אם הרשימה מכילה 4 או 5 איברים – היא מוחקת את האיבר השני וגם הרביעי.)

א. כתוב את הפונקציה הזאת באמצעות לולאה
ב. כתוב את הפונקציה הזאת במימוש רקורסיבי.

החומר נועד לשימוש אישי של תלמידי תיכון קריית שרת בלבד.
אין להעתיק, לשכפל או להפיץ ללא רשות

© צבי מלמד