מבוא לתכנות

הרצאה 7 – מערכים

סמסטר 1

מערך

- מערך הוא ***סידרה*** של ערכים:

'a'	'b'	'c'	'd'	'e'	'f'	'g'	'h'

- כל ערך במערך נקרא ***איבר***
- האיברים יכולים להיות מכל סוג. אבל כל האיברים במערך אחד חייבים להיות מאותו הסוג.

יצירת מערך

- כדי להגדיר מערך, אנחנו פשוט מוסיפים [] לסוג המשתנה:

```
int[] grades;
string[] names;
```

- students ו grades הם שני מערכים ריקים ללא איברים כלל
- כדי להגדיר איברים למערך נשתמש במילת המפתח *new* וב *[num]* כאשר *num* הוא מספר האיברים שנרצה במערך

```
grades = new int[3];
names = new string[4];
```

- כעת ל grades יש 3 איברים מסוג int שערכם 0 , students בעל 4 איברים שערכם ""

– כל סוג יוגדר עם ערך ברירת המחדל: double, 0 ← int, 0.0 ← null ← string (כאשר נדפיס למסך נקבל "") , false ← bool...

גישה לאיברים

- איברים במערך מסודרים ומאונדקסים כמו מחרוזות
  - האינדקס של האיבר הראשון הינו 0
  - האינדקס של האיבר האחרון במערך בגודל *n* הינו (*n*-1)
- כדי לעדכן איבר במקום ספציפי, נשתמש ב:

```
names[1] = "Bob"; //write to the second element
names[0] = "Alice"; //write to the first element
names[3] = "Dora"; //write to the last element
```

- או, שנוכל להגדיר ולעדכן את כל האיברים בבת אחת כך:

```
names = new string[]{"Alice", "Bob", "Charlie", "Dora" };
```

גישה לאיברים

- כדי לקרוא איבר נשתמש באותו סינטקס:

```
names = new string[]{"Alice", "Bob", "Charlie",
"Dora" };
string brown = names[2];
Console.WriteLine(brown); //prints Charlie
```

מעבר על המערך

- כדי לשלוף איברים מהמערך ניעזר באינדקס:

```
names = new string[]{"Alice", "Bob", "Charlie",
"Dora" };
for (int i = 0; i < names.Length; i++)
{
    names[i] = "*" + names[i] + "*";
    Console.WriteLine(names[i]);
}
```

```
*Alice*
*Bob*
*Charlie*
*Dora*
```

– `Length` זוהי תכונה שמחזירה את כמות האיברים במערך הנתון

תרגיל 1

- כתבו את הפונקציה `int Max(int[] values)`. בהינתן מערך של מספרים, הפונקציה תחזיר את המספר הגדול ביותר
- ראשית, בואו נגדיר את האלגוריתם במילים.

תרגיל 2

- יש לנו 2 רשימות:
  - `symbols` – רשימת סמלי NASDAQ עבור חברות.
    דוגמה : [“MSFT”, “INTC”, “CSCO”, “GOOG”]
  - `monthly_gain` - רשימת רווח/הפסד של החודש שעבר, באחוז, של כל חברה דוגמה : [-0.72, 1.05, 0.76, -1.49]
- אנו נרצה למצוא את הסמל של החברה עם הרווח החודשי הגבוה ביותר

סוגי נתונים

- יש שני סוגי של נתונים:

הפניה	ערך
string	int
array	float\double
	boolean
	char

- סוגי "ערך" מכילים ערך יחיד ופשוט.
- סוגי "הפניה" מחזיקים נתונים מורכבים:
  - מחרוזת: רצף תווים
  - מערך: רצף של ערכים

סוגי נתונים

- כאשר שמים ערך במשתנה, המשתנה פשוט מחזיק את הערך:

```
x = 17
```

- אבל כאשר שמים הפניה למשתנה, המשתנה מכיל הפניה לאובייקט:

```
int[] numbers = new int[] {1,2,3};
```

Variable	Value
x	17
numbers	

הפניה

Value
1
2
3

הפניות מרובות

- שימו לב לקוד הבא:

```
int[] a = new int[]{1, 2, 3, 4, 5}
int[] b = a
b[2] = 7
```

  - מהם הערכים בa?
  - מהם הערכים בb?
- זה נראה כאילו a וb הם מערכים נפרדים, אך הם לא.
- למעשה, a וb הם *הפניות* לאותו המערך!
- אם נשנה משהו בb, נשנה גם את a.
  מדוע? בגלל שa וb מפנים לאותו מערך.
- כמו כן גם הפוך, שינויים בb ישנו את a.
- הפניות מרובות היא טכניקת תכנות חשובה שנשתמש בה בהמשך הקורס

סוג הפניה

- כשמשתמשים בסוג הפניה (למעט מחרוזת) כארגומנט, אנו שולחים את ההפניה ולכן הערכים המקוריים של המערך ישתנו מתוך הפונקציה
- מאחורי הקלעים...

```
Int[] x = new int[]{17};
…func(x);

… func(int[] num){…num[0]++…}
```

memory

address	Symbol	Value
#100	x	#103
#101	b	3
#102		
#103		18
#104		
#105	num	#103
…		

ref

ref

תרגיל 3

- כתבו את הפונקציה. `int[] FilterEvens(int[] numbers)`
בהינתן מערך מספרים, הפונקציה מחזירה מערך חדש שמכיל רק את המספרים הזוגיים
- דוגמה : `FilterEvens(new int[]{1,2,3,4,5,6})` תחזיר `{2,4,6}`
- ראשית, בואו נגדיר את האלגוריתם במילים.

Bubble Sort

- אחד האלגוריתמים המפורסמים (והפשוטים) למיין מערך נקרא "מיון בועות". בואו נממש אותו.
- “repeatedly stepping through the list to be sorted, comparing each pair of adjacent items and swapping them if they are in the wrong order. The pass through the list is repeated until no swaps are needed, which indicates that the list is sorted.” (wikipedia)

העתקת מערך

- מה אם אנחנו רוצים ליצור עותק של המערך?
- אופציה 1: להעתיק איבר איבר

```
a = new int[] {70,80,90,100,95};
int[] b = new int[a.Length+2];
for (int i = 0; i < a.Length; i++)
{
    b[i] = a[i];
}
a[0] = 100;
```

  - מה הערכים בa?
  - מה הערכים בb?

70
80
90
100
95
0
0

- כעת a וb מצביעים על 2 רשימות שונות!
- כל שינוי בb לא ישפיע על a
- וגם הפוך, כל שינוי בa לא ישפיע על b

העתקת מערך

- אופציה 2: נשתמש בפונקציית `CopyTo(array, index)`

```
int[] b2 = new int[a.Length];
a.CopyTo(b2, 0);
```

70
80
90
100
95

  - `Array` הוא מערך היעד
  - `Index` זהו האינדקס שממנו מתחילים את ההעתקה

- אופציה 3: נשתמש בפונקציית `Array.Copy(src, dest, length)`

```
int[] b3 = new int[a.Length];
Array.Copy(a,b3, a.Length);
```

70
80
90
100
95

  - `src` הוא מערך המקור
  - `dest` הוא מערך היעד
  - `Length` זהו מספר האיברים שיש להעתיק

העתקת מערך

- אופציה : 4 נשתמש בפונקציית `Clone()` – הפתרון הטוב ביותר.

```
int[] b4 = (int[]) a.Clone();
```

70
80
90
100
95

– נצטרך להמיר את מה שמוחזר ל – `(int[])`

- מדוע?? נדע זאת בעתיד .

מיון מערך

- שיטה שימושית אחרת למערך היא: `Array.Sort()`

```
a = new int[] { 1, 2, 3, 4, 5, 8, 9, 30, 27, 17, 3 };
Array.Sort(a);
```

- השיטה ממיינת את האיברים במערך בהתאם לסוג שלהם ולסדר
- היא יכולה למיין מחרוזות גם כן:

```
names = new string[] { "Bob", "Dora", "Alice",
"Charlie" };
Array.Sort(names);
```

ניקיון מערך

- פונקציית `Clear(array, index, length)` שמה ערכים דיפולטיביים למספר ערכים
  - Index – היכן להתחיל
  - Length – how many elements to clear

```
a = new int[] { 1, 2, 3, 4, 5, 8, 9, 30, 27, 17, 3 };
Array.Clear(a,3,3);
```

1,2,3,0,0,0,9,30,27,17,3,

Foreach

foreach(type varName in ArrName)

{

Use varName here as the current value

}

1. לא ניתן לשנות את תוכן המערך בלולאה זו
2. לא אפשרי לדעת את האינדקס הנוכחי
3. אבל הרבה יותר נוח לכתיבה

```
int[] a = new int[]{ 1, 2, 3, 4, 5, 8, 9, 30, 27, 17, 3};
foreach (int  number in a)
{
    Console.WriteLine(number + 2);
}
```