תאריך: 7.11.25

# מבוא למדעי המחשב – כתה יא-1

## דף עבודה בנושא: מערכים של אובייקטים ו־References – משחק "מוקשים" שלב #1

### מטרת התרגיל

להבין שאובייקט הוא הפניה, ומערך של משתנים מסוג מחלקה מכיל הפניות (References), ולתרגל עבודה עם אובייקטים במערך.

התרגיל הזה יתפתח בהמשך (בשבועות הקרובים) לשלבים נוספים כך שהמשחק יהיה מלא וכיפי, והוא יתרגל נושאים שונים שנלמד.

### מבנה הפרויקט

```
MineSweeper_V1/
├── Program_MSW.cs          מנהל את המשחק כולו.
├── Cell.cs
├── Utils.cs                ← פונקציות עזר כלליות:
├── Lab_MSW_Student_V1.cs   ← הקובץ שבו אתם עובדים!
```

### הוראות עבודה

1. פתחו פרויקט חדש ב־Visual Studio או ב־VS Code (Console App).
2. הוסיפו לתוך הפרויקט את הקבצים שהוזכרו למעלה (מבנה הפרוייקט).

   הקבצים האלו נמצאים במודל או ב GDRIVE. תוכלו להגיע אליהם בקלות דרך הקישור הבא: https://tzvimelamed.com/lab/
3. הריצו את התוכנית.

   אתם אמורים לראות פלט שמודיע אילו חלקים עדיין לא ממומשים.
4. אל תשנו את הקובץ Utils.cs. הוא מדמה "ספרייה קיימת".

### רקע:

המשחק שלנו מדמה שדה מוקשים חד־ממדי. כל תא בשדה מיוצג על־ידי אובייקט מסוג Cell.
כל תא יודע האם יש בו מוקש (hasMine) והאם נחשף (revealed).

---

### שלב א – יצירת הלוח

1. כתבו פונקציה בשם `public static Cell[] CreateBoard(int size)`

   הפונקציה יוצרת ומחזירה מערך באורך size ומכניסה לכל תא אובייקט חדש מסוג Cell.
   כל תא נוצר עם `hasMine = false` ו־`revealed = false`.
2. כתבו פונקציה בשם

```
public static void PlaceMines(Cell[] board, int numMines)
```

הפונקציה ממקמת numMines מוקשים אקראיים במערך.

השתמשו בפונקציה העזר שמגרילה "מיקום" של מוקש:

```
int pos = Utils.RandInRange(0, board.Length - 1);
```

**שאלה:** מה לדעתכם יקרה אם לא נבדוק האם התא כבר מכיל מוקש?

האם יהיו יותר מוקשים או פחות מוקשים?

**הנחיה**: כתבו לולאת while שממשיכה להגריל מיקום עד שנמצא מיקום ריק.

## שלב ב – הדפסה

כתבו פונקציה בשם

```
public static void PrintBoard(Cell[] board)
```

הפונקציה מדפיסה את המערך באופן הבא:

כל תא מיוצג על־ידי הסימן " ? " אם הוא לא נחשף.

אם נחשף והוא מכיל מוקש אז נשתמש במחרוזת: " 💣 " → . אם הוא בטוח (ללא מוקש) נשתמש במחרוזת " 🟩 " → .

בין כל תא לתא מופיע תו | .

בשורה הבאה מתחת להדפסה עלינו להדפיס את האינדקסים של התאים (0, 1, 2, ...).

הנה דוגמה של הדפסת הלוח:

```
=== Mine Sweeper V1 ===

Enter board size: 7
Enter number of mines: 3

=== Mine Field ===
 ? | ? | ? | ? | ? | ? | ? |
 0   1   2   3   4   5   6

Choose a cell (0 - 6), 7 for hint, or -1 to exit: 7

=== FULL BOARD REVEALED ===
🟩 | 💣 | 🟩 | 💣 | 🟩 | 🟩 | 💣 |
 0   1   2   3   4   5   6
```

שימו לב להדפסה של המערך ולאחריו שורת האינדקסים

שימו לב להדפסה של המערך ולאחריו שורת האינדקסים

## שלב ג – חשיפה ובדיקה

כתבו פונקציה בשם `public static bool RevealCell(Cell[] board, int index)`

הפונקציה משנה את התא board[index] כך ש־ revealed = true.

אם התא מכיל מוקש, (hasMine == true) החזירו true והמשחק נגמר.

אחרת, החזירו false.

## שלב ד – בדיקת ניצחון

כתבו פונקציה בשם `public static bool IsGameWon(Cell[] board)`.

הפונקציה בודקת אם כל התאים הבטוחים (כלומר !hasMine) נחשפו.

אם כן → החזירו true.

**שלב ה – ריצה ובדיקה**

6. הריצו את התכנית.
   - בכל סיבוב מוצג הלוח.
   - המשתמש מקיש אינדקס . האינדקס צ"ל ערך חוקי. אם הוא 1- - סיום המשחק. אם הוא בגודל המערך – הצגת המערך והמשך המשחק (כאילו רמז).
   - אם פגע במוקש → הדפיסו 💥 והפסיקו את הלולאה.
   - אם לא → המשיכו לסיבוב הבא.
   - אם כל התאים הבטוחים נחשפו → הדפיסו הודעת ניצחון🎉

---

**למחשבה**

- מדוע אין צורך להחזיר את המערך מתוך הפונקציה `?RevealCell`
- מה היה קורה אילו היינו מעבירים למערך עותקים של תאים במקום הפניות?
- באיזה נושאים חדשים התנסינו בתרגיל הזה?
- מה דעתך על איך שהפונקציה `()main` כתובה? האם יש בה לולאות? האם יש בה משפטי תנאי?