**שיעור 2**

קישור לגיט:<https://github.com/AhuvaElkarif/OOP_Course/blob/main/Lesson2/Lesson2/Program.cs>

תרגיל 1

התוצאה שהתקבלה: 0.

המסקנה: תוצאת ה-0 שמתקבלת נובעת מכך שמבנים (struct) מאוחסנים על ה-stack ולא על ה. heap ה-Garbage Collector (GC) ב-C# עוקב אחרי הזיכרון שהוקצה על ה-heap ולכן כאשר יוצרים אובייקט מסוג struct, לא נמדד שינוי בזיכרון שהוקצה. מכיוון שמבנים הם טיפוסים ערכיים ואינם אובייקטים המנוהלים על ידי ה-GC לא נרשם שינוי בזיכרון שהוקצה ב .heap אם היינו משתמשים במחלקה (class) היינו רואים שינוי בזיכרון, שכן אובייקטים מסוג class מאוחסנים ב-heap והGC עוקב אחריהם.

תרגיל 2

כן, הפקודה new string() מבצעת הקצאת זיכרון בheap מכיוון שמחרוזות בc# מאוחסנות תמיד בheap (מחרוזות הן אובייקטים). לכן ככל שגודל המחרוזת היתה גדולה כך נעשה יותר הקצאה בזיכרון.

תרגיל 3

הקצאת הזיכרון לstring הייתה גדולה הרבה יותר מהקצאת הזיכרון של המחרוזת מסוג StringBuilder .

מסקנה: השימוש ב-StringBuilder יעיל יותר מבחינת צריכת זיכרון לעומת חיבור מחרוזות. בעוד ש-StringBuilder מבצע הקצאה פנימית חכמה ומונע יצירה של אובייקטים חדשים בכל חיבור, חיבור מחרוזות באמצעות += יוצר אובייקט string חדש בכל פעם, מה שמוביל לצריכת זיכרון גבוהה יותר במיוחד כשיש הרבה חיבורים. לכן, עבור יצירת מחרוזות בצורה איטרטיבית, StringBuilder היא הבחירה היעילה יותר.